МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРИЗОВАНИХ СИСТЕМ ЗАХИСТУ ІНФОРМАЦІЇ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ С.В. Казмірчук

«_____»_____ 2020 р.

На правах рукопису
УДК 004.056.53(043.2)

МАГІСТЕРСЬКА АТЕСТАЦІЙНА РОБОТА

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

Тема: Програмний модуль автентифікації з використанням нейромережевого перетворення біометричних ознак в криптографічний ключ

Автор: Ю.Р. Данилюк

Науковий керівник: доцент кафедри КСЗІ А.В. Ільєнко

Нормоконтролер: доцент кафедри КСЗІ А.В. Ільєнко

Київ 2020

ВСТУП

Актуальність. Автентифікація є обов'язковою процедурою перевірки достовірності даних, без якої захищена інформація може виявитися під загрозою.

Одним із недоліків криптографічних систем є необхідність забезпечення надійного зберігання секретних криптографічних ключів. Перспективним напрямом підвищення ефективності систем автентифікації є інтеграція біометричних і криптографічних методів. Біометричні методи в криптографічних системах на початкових етапах застосовувались для захисту від несанкціонованого доступу зловмисників до секретних ключів. На сьогоднішній день їх основним застосуванням є перетворення біометричних параметрів у криптографічний ключ.

Метою дипломної роботи є програмна реалізація модуля автентифікації з використанням нейромережевого перетворення біометричних ознак в криптографічний ключ.

Виходячи з мети, завданням даної дипломної роботи є:

1) дослідити існуючі біометричні криптографічні системи з використанням нейронних мереж та сформувати вимого до власного програмного модулю;
2) розробити алгоритм автентифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ;
3) розробити програмну реалізацію модуля автентифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ за допомогою мови програмування Python;
4) провести тестування та доцільність використання розробленого програмного модулю.

Галузь застосування. Розроблений алгоритм та програмний модуль може бути використаний для будь-яких додатків чи систем, що передбачають наявність автентифікації та мають доступ до камери.

Об'єкт дослідження. Автентифікація з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ.

Предмет дослідження: існуючі методи та засоби біометричної автентифікації з використанням нейронних мереж.

Методи дослідження. Проведені дослідження базуються на технологіях нейромережевої генерації біометричних ознак та їх співставленням з криптографічним ключем шляхом застосування бібліотек Python.

Наукова новизна. В дипломній роботі запропоновано підхід, основою якого є інтеграція біометричної системи і криптографічного модуля, що дозволяє використовувати в якості виходу нейронної мережі згенерований на основі біометричного зразку секретний криптографічний ключ. Цей підхід вирішує проблему надійного зберігання секретного ключа.

Практичне значення полягає у створенні удосконаленого алгоритму та програмного модулю біометричної автентифікації з використанням нейронних мереж, які дозволяють з високою точністю визначити рівень співпадіння вхідного зразку з еталонним та зменшити рівень помилкових спрацювань шляхом застосування бібліотек Python.

Апробація результатів. Робота була апробована на наступних конференціях:

- Данилюк Ю.Р. Аналіз систем біометричної автентифікації з використанням нейромережевого перетворення біометричних ознак в криптографічний ключ // Materiály XVI Mezinárodní vědecko - praktická konference «Aplikované vědecké novinky», Volume 2 : Praha. Publishing House «Education and Science». – P. 9-13.
- Данилюк Ю.Р. Механізм біометричної автентифікації користувача з використанням нейронних мереж / Данилюк Ю.Р., Ільєнко А.В. // Materials of the XVI International scientific and practical Conference Scientific horizons - 2020, September 30 - October 7, 2020: Sheffield. Science and education LTD – P. 57-60

Розділ 1. АНАЛІЗ ІСНУЮЧИХ БІОМЕТРИЧНИХ КРИПТОГРАФІЧНИХ СИСТЕМ

1.1 Основні поняття біометричної автентифікації

Біометричні дані дозволяють автоматично ідентифікувати особу на основі її унікальної характеристики. До часто використовуваних в цілях автентифікації біометричних даних відносять відбитки пальців, сітківку та райдужку ока, геометрію кисті, обличчя, підпис, відбитки долонь тощо.

Головною перевагою біометричних технологій є найвища надійність. І дійсно, усі знають, що двох людей з однаковими відбитками пальців у природі просто не існує. Правда, сьогодні вже відомо кілька способів обману дактилоскопічних сканерів. Наприклад, потрібні відбитки пальців можуть бути перенесені на плівку або до пристрою може бути прикладена велика фотографія пальця зареєстрованого користувача. Втім, треба зізнатися, що сучасні пристрої вже не попадаються на такі прості виверти. Так що зловмисникам доводиться видумувати все нові й нові способи обману біометричних сканерів.

Основним недоліком біометричної ідентифікації є вартість устаткування. Адже для кожного комп'ютера, що входять до цієї системи, необхідно придбати власний сканер. Варто також відзначити, що подібні дешеві сканери недовговічні. Крім того, у них досить високий відсоток помилок другого роду (відмова в доступі зареєстрованому користувачеві). Тому користувачеві доводиться вибирати, який пристрій придбати – дорожчий й кращий або дешевший й гірший.

Нижче наведені основні методи біометричної автентифікації.

Автентифікація за відбитками пальців. Ця біометрична технологія, цілком імовірно, в майбутньому використовуватиметься найширше. Переваги засобів доступу по відбитку пальця - простота використання, зручність і надійність. Весь процес ідентифікації здійснюється досить швидко і не вимагає особливих зусиль від користувачів. Ймовірність помилки при ідентифікації користувача набагато менша порівняно з іншими біометричними методами.

Використання геометрії руки. Цей метод сьогодні застосовується в більш ніж 8000 організацій, включаючи Колумбійський законодавчий орган, Міжнародний Аеропорт Сан-Франциско, лікарні і імміграційні служби. Переваги ідентифікації по геометрії долоні порівнянні з автентифікацією по відбитку пальця в питаннях надійності, хоча пристрій для зчитування відбитків долонь займає більше місця. Найбільш досконалий пристрій, Handkey, сканує як внутрішню, так і бічну сторону руки.

Автентифікація за райдужною оболонкою ока. Перевага сканування райдужної оболонки полягає в тому, що зразок плям на оболонці знаходиться на поверхні ока, і від користувача не вимагається спеціальних зусиль. Фактично відеозображення ока може бути відсканованим на відстані метра, що робить можливим використання таких сканерів в банкоматах. Ідентифікуючі параметри можуть скануватися і кодуватися, зокрема, і у людей з ослабленим зором, але непошкодженою райдужною оболонкою.

Автентифікація за сітківкою ока. Сканування сітківки відбувається з використанням інфрачервоного світла низької інтенсивності, направленого через зіницю до кровоносних судин на задній стінці ока. Сканери для сітківки ока набули великого поширення в надсекретних системах контролю доступу, оскільки ці засоби автентифікації характеризуються одним з найнижчих відсотків відмови в доступі зареєстрованим користувачам і майже нульовим відсотком помилкового доступу.

Автентифікація за рисами особи (за геометрією особи) - один з напрямів, що швидко розвиваються в біометричній індустрії. Розвиток цього напряму пов’язаний з швидким зростанням мультимедійних відео-технологій. Проте більшість розробників поки зазнають труднощі в досягненні високого рівня виконання таких пристроїв. Проте можна чекати появу в найближчому майбутньому спеціальних пристроїв ідентифікації особи за рисами обличчя в залах аеропортів для захисту від терористів і т. ін.

Існує два основних процеси біометричного розпізнавання: реєстрація, під час якої система виконує індивідуальне порівняння біометричних даних, та власне розпізнавання, тобто співставлення отриманих зразків із шаблонними даними.

Структурна схема, представлена на рис. 1.1, ілюструє покрокову процедуру проведення біометричної автентифікації [1].

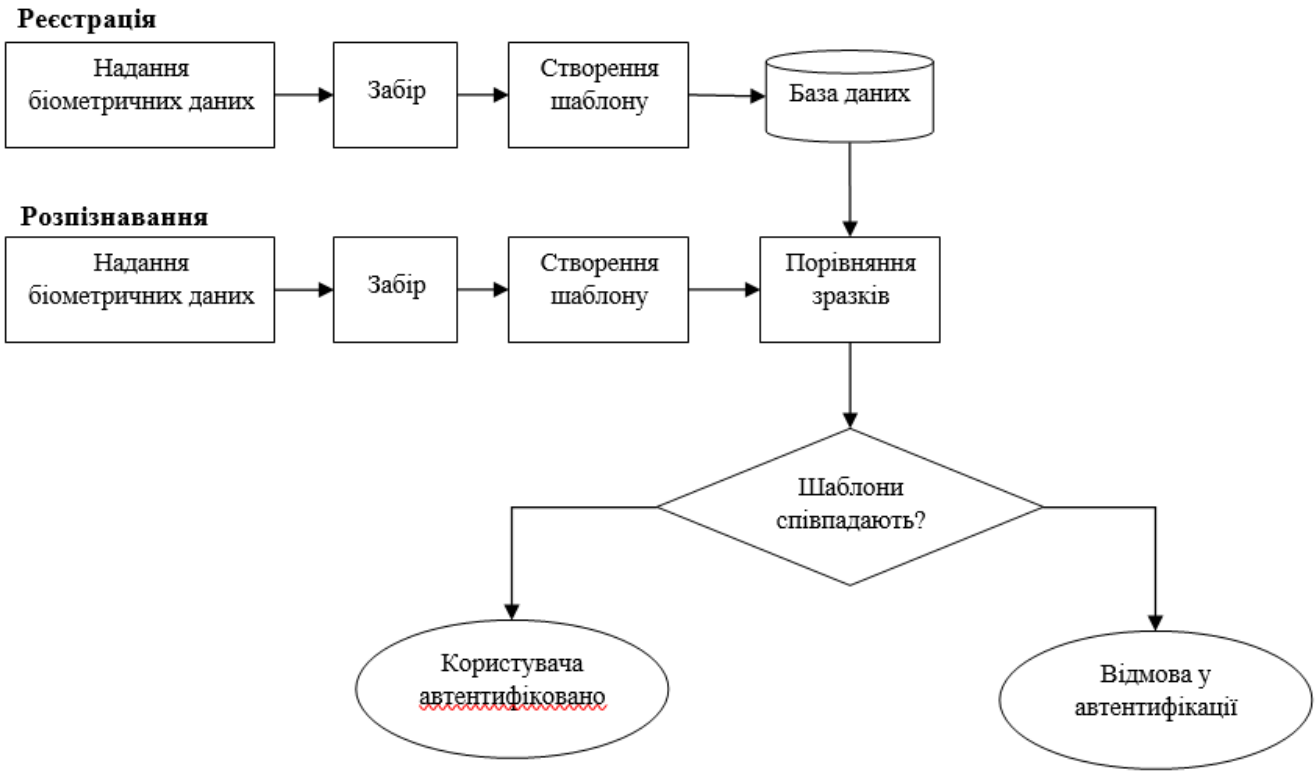

Рис 1.1 Процеси біометричного розпізнавання

Біометрична автентифікація конкретного шаблону може бути представлена у вигляді 4 послідовних кроків [1]:

1) отримання зразків (перший базовий крок отримання біометричних зразків людини і подальшого занесення їх в базу даних);

2) попередня обробка зразків (включає в себе перетворення зображення з кольорового режиму в чорно-білий, локалізацію, оцінку якості, прибирання шуму та ін.);

3) витягання зразків (отримання найбільш релевантної інформації з вхідних даних і представлення цієї інформації у більш компактному вигляді);

4) співставлення / класифікація (найбільш важливий крок, метою якого є розпізнавання).

Нині існує два основних напрями застосування біометричних методів: вирішення задачі автентифікації користувача і їх інтеграція з криптографічними системами [2-3].

Одним із недоліків криптографічних систем є необхідність забезпечення надійного зберігання секретних криптографічних ключів. Перспективним напрямом підвищення ефективності систем автентифікації є інтеграція біометричних і криптографічних методів. Біометричні методи в криптографічних системах на початкових етапах застосовувались для захисту від несанкціонованого доступу зловмисників до секретних ключів. На сьогоднішній день їх основним застосуванням є перетворення біометричних параметрів у криптографічний ключ.

Вихідні біометричні ознаки користувача відтворюються недостатньо чітко, що ускладнює їх використання в криптографічних перетвореннях, де вимагається точне значення ключа [2].

Для ефективного розпізнання біометричних зразків часто використовуються штучні нейронні мережі, які дозволяють з високою точністю визначити рівень співпадіння вхідного зразку з еталонним та зменшити рівень помилкових спрацювань.

Передумовою використання нейронних мереж в якості математичної основи при створенні нових методів криптографічного захисту інформації може служити їх здатність до відновлення спотворених сигналів і розпізнавання об'єктів, що мають характеристики відмінні від еталонних. Додатковою перевагою є апаратна реалізація нейромережевих алгоритмів, що дозволяє збільшити швидкість шифрування і дешифрування даних. Однією з основних проблем, що заважають просуванню нейромережевих методів шифрування, є погана вивченість питань їх криптостійкості, що робить актуальним завдання дослідження характерних особливостей і пошуку вразливостей нейромережевих криптографічних алгоритмів.

Біометричні системи автентифікації на основі застосування технологій штучного інтелекту у більшості випадків апроксимують нелінійне функціональне відображення, дозволяють віднести біометричний образ, що розпізнається, до

одного з попередньо визначених класів. Такі системі вразливі до атаки на «останній біт» [4].

Переваги використання нейронних мереж:

- можливість отримувати смислове значення зі складних даних;
- адаптивне навчання – здатність навчитися виконувати завдання на основі даних, наданих для навчання;
- самоорганізація – штучна нейронна мережа може створювати організацію або представлення інформації, яку вона отримує під час навчання;
- робота в реальному часі – обчислення нейронної мережі можуть розбиватись на паралельні потоки;
- відмовостійкість до несправностей через надмірне кодування інформації.

Системи біометричної автентифікації, що використовують нейромережеві технології, мають ряд недоліків:

1) необхідність повторно навчати нейронну мережу у разі додавання нового користувача (може бути вирішено за допомогою модульної структури системи);
2) такі системи вразливі до помилки другого роду, що полягає у ймовірності допуску в систему несанкціонованого користувача.

Таким чином, біометричні системи необхідно вдосконалювати за рахунок створення комплексних систем, що враховують декілька джерел біометричних образів. А також, варто використовувати методи, що дозволяють однозначно ідентифікувати пару «біометричний образ» - «криптографічний ключ».

1.2 Helper data в біометричних системах

Біометричні криптосистеми вимагають зберігання публічної інформації, яка залежить від біометрії. Ця інформація застосовується для отримання або генерації ключів і називається helper data (допоміжні дані)[8]. Через біометричну дисперсію неможливо безпосередньо витягти ключі з біометричних характеристик.

Допоміжні дані, які не розкривають суттєвої інформації про оригінальні біометричні шаблони, використовуються для реконструкції криптографічних

ключів. Біометричні порівняння проводяться шляхом перевірки дійсності ключів, а результатом процесу автентифікації є або ключ, або повідомлення про помилку [9].

Біометричні криптосистеми вимагають генерації або отримання стовідсотково правильних ключів, тоді як звичайні біометричні системи просто відповідають "Так" або "Ні". Крім того, більшість біометричних криптосистем забезпечують вищий ступінь квантування при вилученні біометричних ознак порівняно зі звичайними біометричними системами, які здатні встановлювати більш точні пороги для регулювання швидкості розпізнавання.

Існує три види біометричних криптосистем, залежно від способу отримання допоміжних даних:

- криптосистеми з випуском ключа (key release cryptosystems);
- криптосистеми зі зв’язуванням ключа (key binding cryptosystems);
- криптосистеми з генерацією ключа (key generation cryptosystems).

1.3 Біометрична система з випуском ключа

У системах з випуском ключів криптографічний ключ розглядається як секретний, а біометричні дані використовуються як ключ для захисту випадково сформованого криптографічного ключа. Криптографічний ключ видається лише тоді, коли справжня біометрія представлена в криптографічній конструкції.

Існує два способи пов’язання криптографічного ключа з біометричним, а саме схема нечіткого сховища (fuzzy vault scheme) [10] та схема нечітких зобов’язань (fuzzy commitment scheme) [11].

Нечітке сховище – це схема шифрування, запропонована Джуелсом та Суданом [15], яка використовує деякі концепції кодів, що виправляють помилки, для кодування інформації таким чином, що її важко отримати без "ключа", використовуваного для її кодування, навіть якщо методи, що використовуються для кодування, є загальновідомими. Хоча такий підхід може працювати з будь-яким кодом, схема нечіткого сховища часто використовується з кодами Рід-Соломона.

Коди Ріда-Соломона працюють, кодуючи значення у повідомленні як коефіцієнти багаточлена, а потім обчислюючи цей поліном у наборі точок, щоб

отримати кодове слово для повідомлення. Використовуючи кількість оцінювальних балів, більшу за ступінь полінома, можна буде отримати поліном (а отже і повідомлення) шляхом інтерполяції навіть за наявності відсутніх або помилкових значень. Однак, якщо помилок занадто багато, не буде унікального інтерполюючого полінома належного ступеня. Ці властивості використані у схемі нечіткого сховища.

Секрет кодується за допомогою набору значень ("ключ"), а потім може бути розблокований за допомогою іншого набору значень, лише якщо він суттєво збігається з набором, який використовується для його блокування. Цей підхід пропонує незмінність порядку, що означає, що набори, які використовуються для блокування та розблокування секрету, є впорядкованими. Завдяки цій властивості схема нечіткого сховища була застосована до біометричного шифрування [14], а саме до автентифікації за відбитками пальців.

Нечітке сховище є схемою, яка доповнює традиційні криптографічні системи безпеки, поєднуючи їх з біометричною автентифікацією, щоб подолати вразливість системи безпеки, властиву сховищу криптографічних ключів. Біометричні системи шифрування, засновані на схемі нечіткого сховища, підходять для окремих пристроїв безпеки та автентифікації у вигляді системи на мікросхемі.

Схема нечітких зобов'язань була запропонована Джулсом та Ваттенбергом у 1999 році. Це найпростіша, але найбільш досліджена серед усіх схема, яка вважається найбільш придатною для використання в біометриці, де шаблони мають вигляд впорядкованого рядка або вектора об’єктів. Метод є продовженням криптографічного зобов'язання, але допускає певну варіативність у значеннях за рахунок корегуючих кодів [16].

Нечітка схема зобов’язань складається з функції F, яка використовується для введення кодового слова $c \in C$ та підтвердження $x \in \{0,1\}n$. Набір C являє собою набір виправлених помилок кодових слів c довжини n, x являє собою бітовий потік довжини n, що називається підтвердженням (біометричні дані). Вектор різниці c та x, $\delta \in \{0,1\}$ n, де $x = c + \delta$, та хеш-значення $h(c)$ зберігаються як зобов’язання, яке називається $F(c, x)$ (допоміжні дані). Кожен x’, який є достатньо «близьким» до x, відповідно до певної метрики, повинен мати можливість реконструювати c за

допомогою вектора різниці δ для перекладу x' у напрямку x. Хеш результату перевіряється на $h(c)$. Що стосується біометричної прив'язки ключів, система отримує підтвердження x при зарахуванні, вибирає кодове слово $c \in C$, обчислює та зберігає зобов'язання (c), (δ та $h(c)$). Під час автентифікації залучається свідок x, і система перевіряє, чи дає x успішне скасування зобов'язання.

Схема нечітких зобов'язань складається з етапів реєстрації та власне автентифікації, які зображені на рис. 1.2 та 1.3 відповідно.

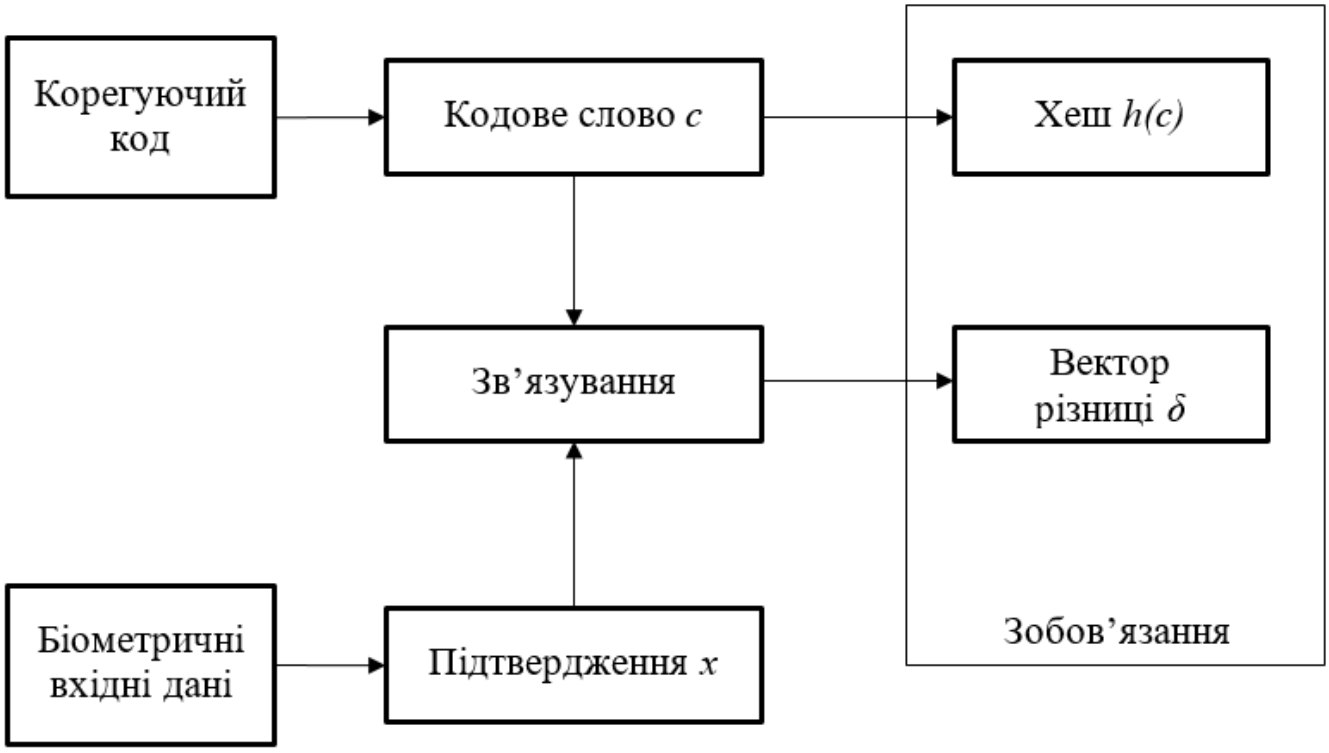

Рис. 1.2. Схема нечіткого зобов'язання (етап реєстрації)

Фенг Хао та колеги запропонували схему нечітких зобов'язань на основі біометрії, де випадково сформований криптографічний ключ K захищений кодом райдужної оболонки. Для випуску секретного ключа, що відбувається під час регенерації ключа, створюється новий запит до тієї ж райдужки, щоб відновити або випустити той самий криптографічний ключ [12].

У існуючій роботі з випуску ключів дослідники також намагаються забезпечити криптографічний захистит, використовуючи схему нечіткого сховища

на основі відбитків пальців [13]. У роботі [13] координати спеціальних точок відбитку використовуються для приховування секретного ключа (K) за допомогою криптографічної конструкції (наприклад, нечіткого сховища). Ключ вивільняється зі сховища, коли запит з тим самим відбитком пальця подається до сховища як вхід.

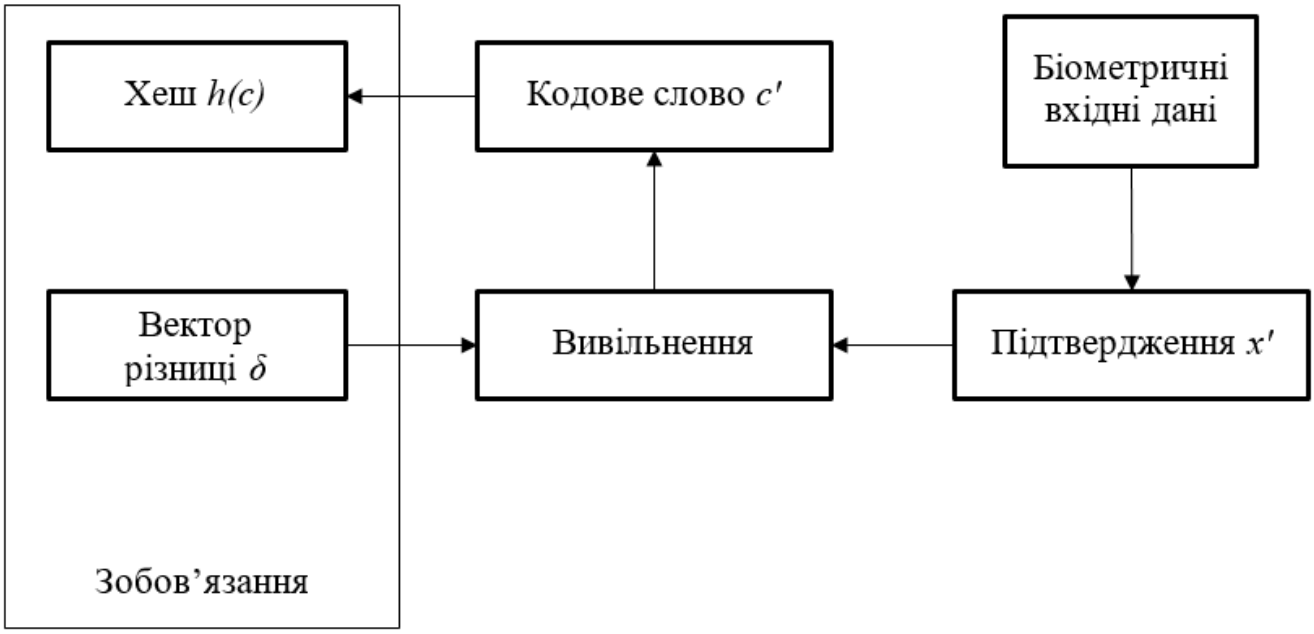

Рис. 1.3. Схема нечіткого зобов'язання (етап автентифікації)

Подібним чином Нандакумар та співавтори також запропонували підхід на основі схеми нечіткого сховища з відбитками пальців, де значення координат, а також орієнтація спеціальних точок відбитку розглядаються як дані для захисту випадково сформованого ключа [14].

1.4 Біометрична система зі зв'язуванням ключа

У системах зі зв'язуванням ключа допоміжні дані отримуються шляхом прив'язки вибраного ключа до біометричного шаблону. В результаті процесу зв'язування злиття секретного ключа та біометричного шаблону зберігається як допоміжні дані. Застосовуючи відповідний алгоритм пошуку ключів, ключі отримують із допоміжних даних при автентифікації [17].

Оскільки криптографічні ключі не залежать від біометричних функцій, їх можна скасувати, тоді як оновлення ключа, як правило, вимагає повторної реєстрації

для створення нових допоміжних даних. До основних підходів біометричного зв'язування ключів відносять Mytec1 та Mytec2, схему нечітких зобов'язань та нечітке сховище.

Перший серйозний підхід до біометричного зв'язування ключів на основі відбитків пальців був запропонований у [18-20]. Представлена система отримала назву Mytec2, і стала наступником Mytec1 [21], яка була першою біометричною криптосистемою, але виявилася непрактичною з точки зору точності та безпеки. В основі алгоритму Mytec2 (і Mytec1) лежить механізм кореляції. Алгоритм, що лежить в основі Mytec2, був узагальнений у патенті [22], який включає пояснення щодо застосування алгоритму до інших біометричних характеристик, таких як райдужна оболонка ока. Однак у публікаціях не повідомляється про вимірювання продуктивності.

Основні відомості щодо схем нечіткого зобов’язання та нечіткого сховища представлені у п. 1.3.

1.5 Біометрична система з генерацією ключа

У системах з генерацією ключів допоміжні дані отримуються лише з біометричного шаблону. Ключі генеруються безпосередньо з допоміжних та даного біометричного зразка. Хоча зберігання допоміжних даних не є обов'язковим, більшість запропонованих схем генерації ключів зберігають їх (якщо схеми генерації ключів витягають ключі без використання будь-яких допоміжних даних, вони не можуть оновлюватися у випадку компромісу).

Попередня ідея генерації ключів безпосередньо з біометричних шаблонів була представлена в патенті Бодо [23]. Реалізації цієї схеми не існує, і очікується, що більшість біометричних характеристик не забезпечують достатньо інформації для надійного вилучення досить довгого та оновлюваного ключа без використання будь-яких допоміжних даних.

Етапи реєстрації та автентифікації криптосистем з генерацією ключа зображені на рис. 1.4 та 1.5 відповідно.

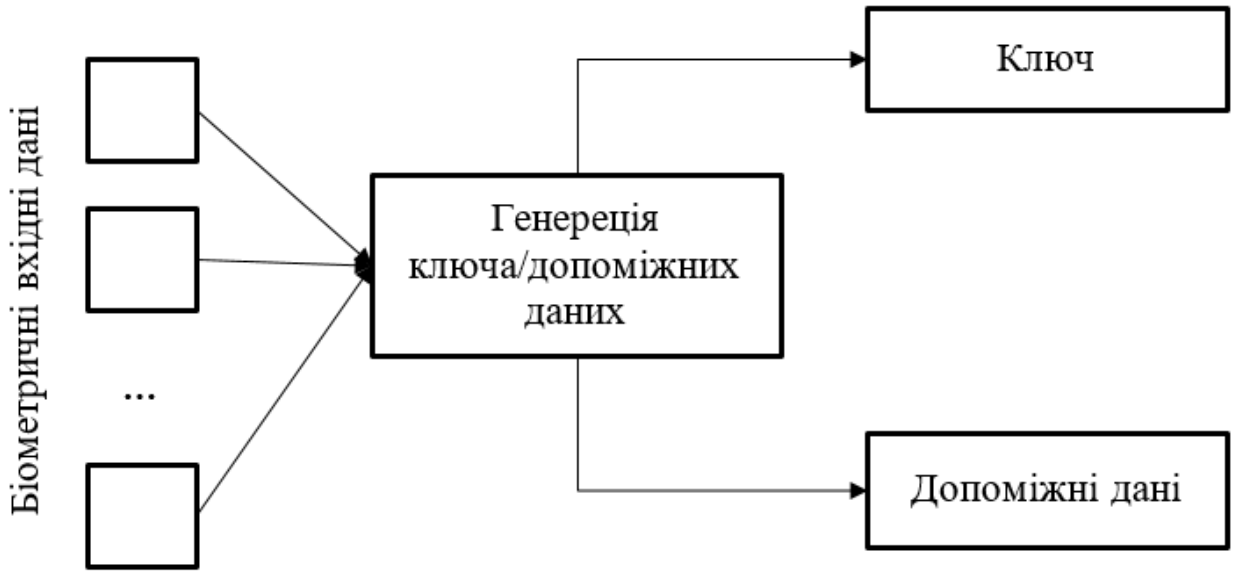

Рис. 1.4. Криптосистема з генерацією ключа (етап реєстрації)

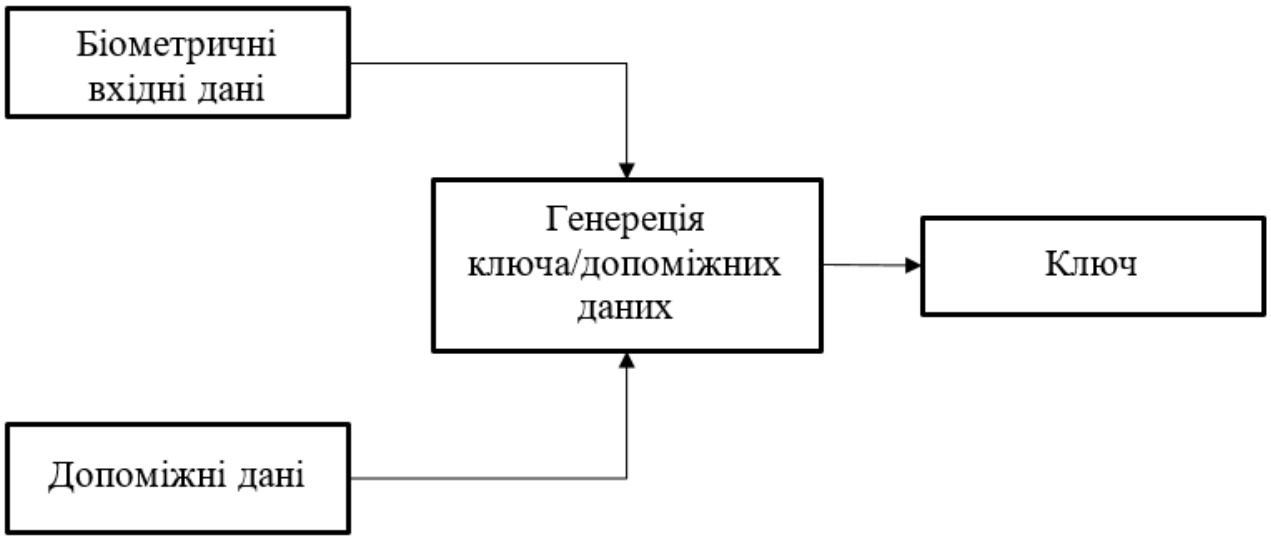

Рис. 1.5. Криптосистема з генерацією ключа (етап автентифікації)

Схеми генерації ключів, що базуються на допоміжних даних, також відносяться до "нечітких екстракторів" або "захищених ескізів". Нечіткий екстрактор генерує надійний випадковий рядок із біометричного входу, тоді як збережені допоміжні дані допомагають відтворенню. На відміну від цього, у

захищеному ескізі допоміжні дані застосовуються для відновлення вихідного біометричного шаблону.

Є дві схеми, в яких використовуються допоміжні дані. Схема приватного шаблону, заснована на райдужній оболонці, була запропонована Давідою та колегами [24-25]. В цій схемі сам біометричний шаблон (або його хеш-значення) служить секретним ключем. Зберігання допоміжних даних, які є бітами перевірки виправлення помилок, потрібно для виправлення несправних бітів заданих кодів райдужки. В іншій групі схем, званих схемами квантування, допоміжні дані будуються таким чином, щоб допомагати в квантуванні біометричних ознак для отримання стабільних ключів.

1.6 Порівняльний аналіз існуючих систем

Порівняльна характеристика існуючих криптографічних систем надана в табл. 1 [2-6].

Для генерації ключів на основі біометричних образів використовуються 2 основних інструмента, що задовольняють вимогам сучасної криптографії та мають низьку ймовірність помилки другого роду: нейромережевий перетворювач «біометрія – код» та нечіткі екстрактори.

Нейромережевий перетворювач «біометрія – код» заснований на використанні нейронної мережі з невеликою кількістю прихованих шарів або взагалі без їхнього використання [4, 5]. Розмірність простору ознак і вихідного вектору мережі, навпаки, велика, що дозволяє реалізувати перетворення нечіткого вхідного вектору біометричних признаків («свій») у відповідний криптографічний ключ, а випадковий вхідний вектори («чужий») – у випадковий вихідний образ [2].

Нечіткі екстрактори (fuzzy extractors) [3, 6] дозволяють однозначно відновити секретний ключ із нечіткого біометричного образу за рахунок використання допоміжних даних [7]. «Нечіткий» в даному контексті говорить про те, що фіксовані значення, необхідні для криптографії, будуть вилучені з значень, близьких, але не ідентичних вихідному ключу. В основі роботи нечіткого екстрактора лежить

можливість вилучення випадкової рівномірно розподіленої послідовності та коректне її відновлення з розмитого біометричного образу.

Табл. 1.1.

Порівняльний аналіз існуючих криптографічних біометричних систем

	З випуском ключа «key release cryptosystems»	**Зі зв'язуванням ключа «key binding cryptosystems»**	**З генерацією ключа «key generation cryptosystems»**
Особливості	Біометричний еталон та ключ зберігаються окремо	Криптографічний ключ і біометричний еталон пов'язані за допомогою алгоритму заміщення невеликої кількості секретних біт криптографічним ключом; використовуються коригуючі коди; нечіткий контейнер (fuzzy vault) – найбільш розповсюджена схема	Криптографічний ключ береться з біометричних даних і його не потрібно зберігати в базі даних; великі штучні нейронні мережі; нечіткі екстрактори (fuzzy extractors)
Переваги	Простота реалізації	Безпечність методу заснована на секретності алгоритмів закриття та відновлення ключа	Криптографічний ключ не зберігається в базі даних
Недоліки	Біометричні еталони зберігаються локально; необхідний доступ до біометричних еталонів, що	Детерміновані алгоритми закриття ключа можуть бути компрометованими; алгоритми складні в реалізації через мінливість біометричних ознак	Висока складність реалізації системи; біометричні дані відтворюються неточно, що ускладнює їх використання у якості основи стійкої

	зберігаються локально і є незахищеними та незашифрованими		генерації ключа
Вразливості та атаки	Підміна зловмисником модуля порівняння образів за допомогою шкідливого програмного забезпечення	Кореляційна атака (correlation attacks, attacks via record multiplicity – ARM); атака з інверсією ключа (surreptitious key inversion attacks – SKI); атака підстановки зі змішуванням (blended substitution attacks)	

1.7 Висновки до розділу 1

Автентифікація є обов'язковою процедурою перевірки достовірності даних, без якої захищена інформація може виявитися під загрозою.

Перспективним напрямком розвитку систем автентифікації є біометричні криптографічні системи. Основні зусилля в цьому напрямку спрямовані на розвиток і вдосконалення апаратно-програмних засобів, які дозволили б досягти одночасного і значного зменшення рівня помилок першого та другого роду, а також були б захищені від спуфінг-загроз. Масовий випуск таких систем дозволить значно зменшити їх вартість.

Більшість біометричних криптосистем націлені на зв’язування або генерацію ключів, достатньо довгих для застосування в загальній криптографічній системі (наприклад, принаймні 128-бітна довжина ключів для алгоритму AES). Щоб запобігти підбору біометричних ключів або перебору brute force, вони повинні мати достатній розмір та ентропію.

Ефективність системи біометричних криптосистем здебільшого розцінюється з точки зору коефіцієнта помилкового відхилення та помилкового прийняття.

Оскільки як метрики, так і ключова ентропія залежать від допустимих рівнів при порівнянні, ці три величини дуже взаємопов'язані.

Другим фактором, який впливає на безпеку біометричних криптосистем, є витік конфіденційності. В ідеалі, витоки конфіденційності повинні бути мінімізовані (для заданої довжини ключа), щоб уникнути шахрайства з особистими даними. Вимоги щодо розміру ключів та витоку конфіденційності визначають фундаментальний компроміс у рамках підходів до біометричних криптосистем, який рідко оцінюється (цей компроміс вивчається в інформаційно-теоретичній перспективі).

Розділ 2. СТРУКТУРА ТА АЛГОРИТМ ФУНКЦІОНУВАННЯ СИСТЕМИ АВТЕНТИФІКАЦІЇ

2.1 Структура нейромережевої системи біометричної автентифікації

Структура неромережевої системи (НС) біометричної автентифікації наведена на рис. 2.1, 2.2.

При розробці структури системи автентифікації з нейромережевим перетворенням вхідних біометричних ознак в криптграфічний закритий ключ алгоритм було розділено на п’ять умовних етапів:

1) попередня обробка ознак;
2) генерація ознак;
3) створення бази біометричних зразків;
4) нейромережеве співставлення образів і ключів;
5) відновлення ключа.

Задача блоку попередньої обробки ознак полягає в уніфікації зображення обличчя людини і побудові первинного вектору ознак.

Далі вектори ознак подаються до блоку генерації ознак, де відбувається їх перетворення в компактні вектори ознак, які містять мінімально достатній об’єм інформації.

У випадку, коли потрібно додати користувача в систему, компактний вектор ознак подається на вхід блоку створення біометричних образів, де виконується нейромережеве співставлення закритого ключа і вектора, що подається на вхід.

Блок відновлення ключа необхідний для трансформації компактного вектора ознак в необхідний ключ користувача. Слід відзначити, що при спробі несанкціонованої авторизації система видає випадковий ключ, який в системі не використовується.

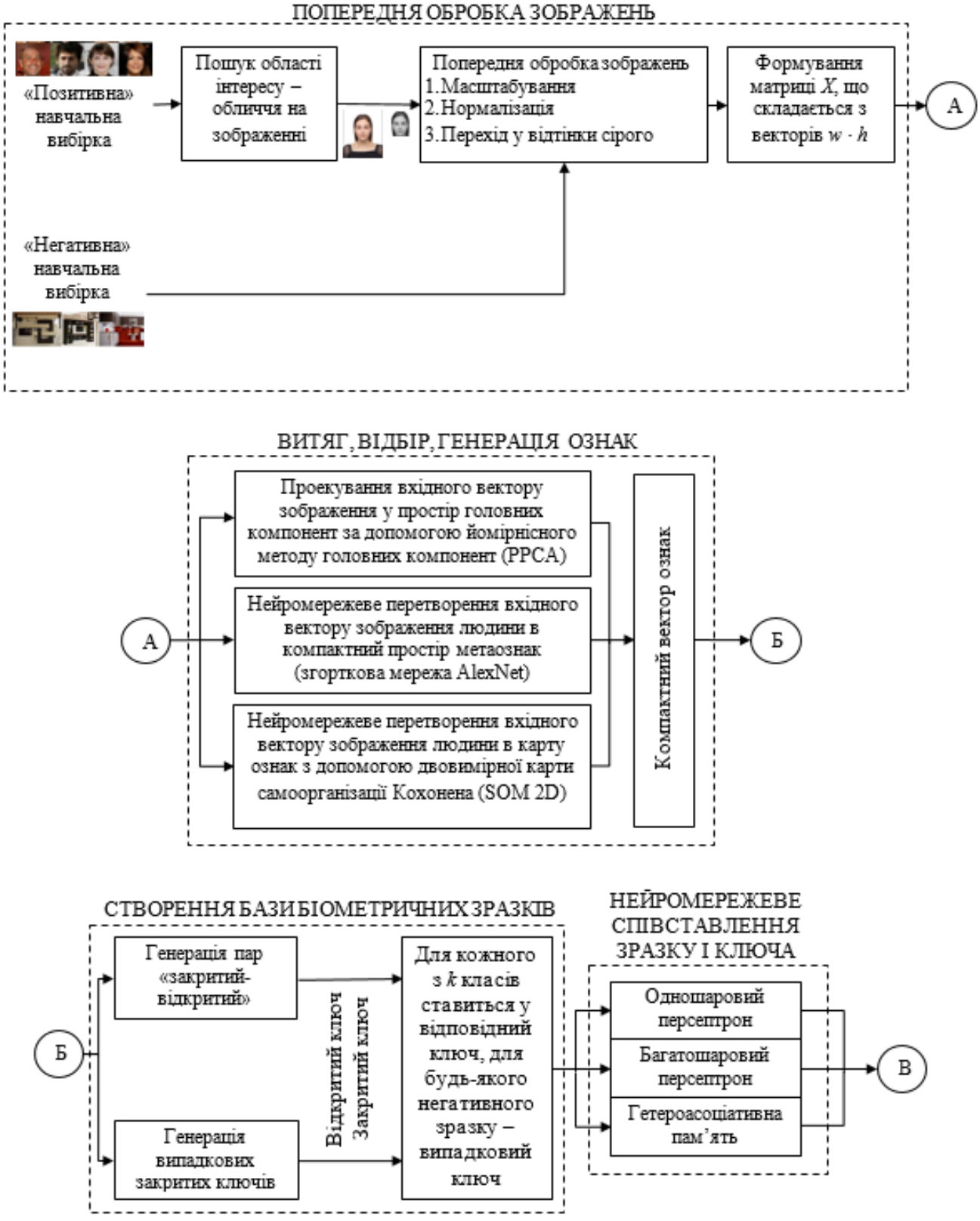

Рис. 2.1. Структура нейромережевої системи біометричної автентифікації

Рис. 2.2. Структура нейромережевої системи біометричної автентифікації (продовження)

2.2 Підходи до генерації біометричних ознак

Генерація ознак полягає в проекуванні первинного вектору на новий простір ознак і формуванні компактного вектору ознак кожного образу для подальшої нейромережевої обробки. Цей вектор подається в нейромережевий блок у формі двійкового вектору або ж у формі цілочисельного вектору, який, в свою чергу, генерує закритий ключ, що подається на вхід модуля криптографічної системи.

Для генерації ознак пропонується використати наступні підходи:

1) двовимірна карта самоорганізації Кохонена (self-organizing map, SOM) [26];
2) ймовірнісний метод головних компонент (ЙМГК, probabilistic principal component analysis, PPCA) [27];
3) згорткова нейронна мережа (convolutional neural network, CNN) AlexNet [28].

2.2.1 Самоорганізована двовимірна карта Кохонена

Самоорганізована карта Кохонена (Самоорганізована карта, SOM) - це обчислювальний метод, призначений для завдань, насамперед, кластеризації та візуалізації, а також аналізу даних із просторів високої розмірності (тобто,

багатовимірних даних), отриманих експериментально. Метод був запропоновані Туево Кохоненом (1982). Родоначальниками моделі самоорганізованої мережі Кохонена були ранні моделі нейронних мереж (зокрема, модель асоціативної пам'яті та адаптивна модель навчання).

Метою застосування самоорганізованої карти Кохонена є пошук прихованих закономірностей в даних на основі зменшення розмірності вихідного простору в простір нижчого виміру (на практиці найчастіше використовують двовимірний, зокрема, завдяки зручній візуалізації). При цьому топологія вихідного простору залишається незмінною. В результаті тренування цієї моделі генерується решітка, що складається з тренованих нейронів, яку також називають «картою» вихідного простору.

Самоорганізована карта Кохонена має наступну архітектуру: є два шари - вхідний шар нейронів (розподілений) і вихідний шар нейронів (шар Кохонена), при цьому нейрони другого шару розташовані у вигляді двовимірної решітки (зазвичай сітка буває квадратною або шестикутною) так, що кожен нейрон першого шару з'єднаний з кожним нейроном другого шару (рис. 2.3).

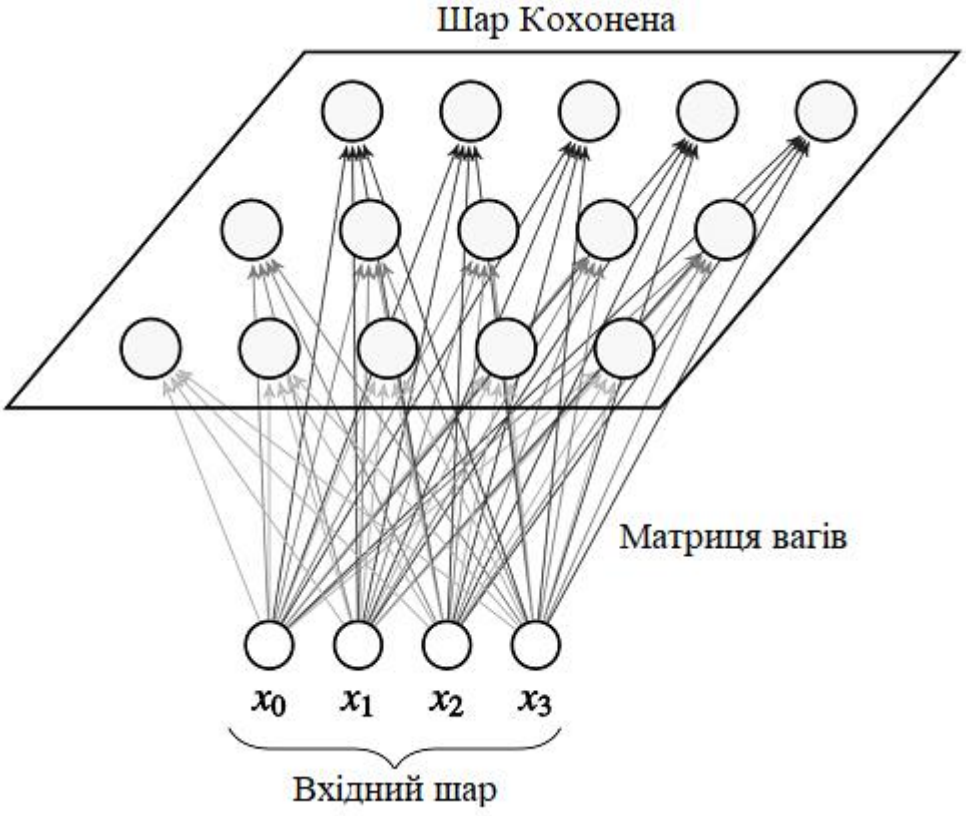

Рис. 2.3. Загальна структура карти Кохонена

Кількість нейронів у вхідному шарі рівна розмірності вхідного простору. Найчастіше нейрони шару Кохонена зазвичай називають кластерними елементами. Точніше, їх кількість визначає кількість кластерів, на які карта може розділити дані, що подаються на рівень розподілу. Чим більше елементів кластера, тим більш детальна кластеризація.

Як уже зазначалося, топологію шару Кохонена можна представити або як чотирикутну сітку, або як гексагональну сітку. Другий варіант є більш привабливим завдяки тому, що відстань для кожного нейрона до будь-якого сусіднього йому нейронає однаковою (рис. 2.4).

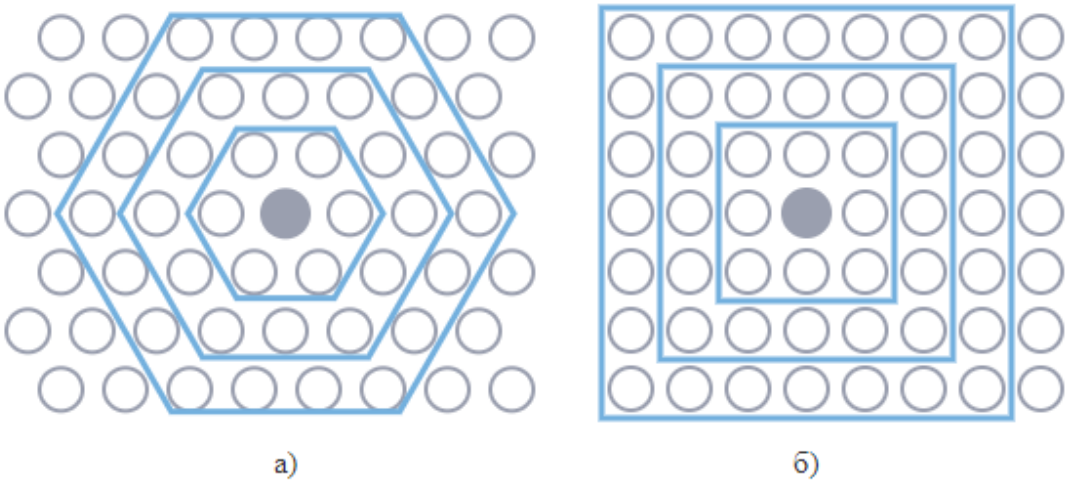

Рис. 2.4. Відстань між нейронами на карті гексагональної (а) та чотирикутної (б) сіток

Кожен нейрон вихідного шару визначається вектором вагів (розмірність вектора є різницею вхідного простору) та упорядкованою парою: (x, y), яка визначає положення нейрона на карті Кохонена.

Мережа має наступний принцип навчання: для кожного вхідного вектора визначається нейрон-переможець із шару Кохонена, після чого коригуються ваги нейрона-переможця та ряду його сусідів. Процедура повторюється для кожного вхідного вектора з навчальної вибірки.

Мережа із запропонованою архітектурою навчається на тренувальному наборі. Навчання полягає у регулюванні вагів нейронів у шарі Кохонена. Після навчання мережа починає свою роботу, присвоюючи кожен нейрон з тестового зразка тому чи іншому кластеру.

Загальні властивості алгоритму Кохонена:

- апроксимація вхідного простору (карта ознак, представлена множиною векторів синоптичних ваг, у вихідному просторі реалізує хорошу апроксимацію вхідного простору);
- топологічний порядок (карта ознак, отримана за алгоритмом SOM, топологічно впорядкована таким чином, що просторове положення нейронів у решітці відповідає конкретній області або ознаці вхідного зображення);
- вибір ознак (для даних із вхідного простору з нелінійним розподілом самоорганізована карта для апроксимації досліджуваного розподілу здатна виділити набір найкращих ознак).

Переваги алгоритму:

- алгоритм стійкий, в тому числі до даних з завадами;
- реалізує швидке та неконтрольоване навчання (процес самоорганізовується);
- може візуалізувати багатовимірні вхідні дані (спрощення багатовимірних вхідних даних за допомогою візуалізації);
- може апроксимувати будь-яку безперервну функцію, що дозволяє досліднику заздалегідь не висувати жодних гіпотез щодо моделі.

Недоліки алгоритму:

- працює лише з дійсними числовими векторами;
- може бути використаний для кластерного аналізу, лише якщо кількість кластерів відома заздалегідь;
- наявність спотворень (близькі об'єкти вихідного простору можуть перетворюватись у віддалені точки на карті);

- кінцевий результат роботи нейронних мереж сильно залежить від початкових налаштувань мережі (початковий розподіл ваг, властивості потенційної функції) [29].

2.2.2 Ймовірнісний метод головних компонент

Метод головних компонент (розклад Карунена-Лоєва, principal component analysis, PCA) - найпростіший метод зменшення розмірності даних, основні ідеї якого були сформовані ще в 19 столітті [30, 31].

Ідея методу полягає в пошуку у вхідному просторі гіперплощини заданої розмірності з подальшим проектуванням виборки на дану гіперплощину. При цьому обирається така гіперплощина, помилка проектування даних на яку є мінімальною у відношенні суми квадратів відхилень.

Для методу головних компонент можна сформулювати йомовірнісну модель (probabilistic PCA, PPCA). Переформулювання методу у ймовірнісних термінах дає цілий ряд переваг, а саме:

- можливість використовувати EM-алгоритм (Expectation-maximization algorithm) для пошуку рішення;
- правильна обробка пропущених значень (пропущені значення додаються до переліку прихованих змінних ймовірнісної моделі, до яких потім застосовується відповідний алгоритм EM);
- можливість переходу до моделі змішаного розподілу ймовірностей, яка значно розширює сферу застосування методу;
- можливість використання Байєсовського підходу до вирішення задач вибору моделі;
- можливість генерувати нові об’єкти на основі ймовірнісної моделі;
- для завдань класифікації можливість моделювати розподіли окремих класів об'єктів для подальшого використання їх в різних схемах класифікації;
- значення функції правдоподібності є універсальним критерієм, що дозволяє порівнювати різні ймовірнісні моделі між собою (зокрема, за

допомогою значення правдоподібності можна легко визначити відхилення в даних).

Як і класична модель PCA, ймовірнісна модель PCA є інваріантною щодо вибору базису в гіперплощині.

На відміну від класичної моделі PCA, в якій відновлюється тільки гіперплощина, що найкращим чином пояснює дані, ймовірнісна модель відновлює всю модель мінливості даних. Зокрема, вона описує дисперсію даних по усіх напрямках. Тому рішення включає в себе не тільки напрямні базисні вектори гіперплощини, що задаються власними векторами матриці коваріаций, але також і довжини цих базисних векторів [32].

2.2.3 Згорткова нейронна мережа AlexNet

Згорткова нейронна мережа (англ. Convolutional neural network, CNN) - спеціальна архітектура штучних нейронних мереж, запропонована Яном Лекуном в 1988 році і націлена на ефективне розпізнавання образів, входить до складу технологій глибокого навчання (англ. Deep learning).

Згорткова нейронна мережа використовує деякі особливості зорової кори, в якій були відкриті так звані прості клітини, що реагують на прямі лінії під різними кутами, і складні клітини, реакція яких пов'язана з активацією певного набору простих клітин. Таким чином, ідея згорткових нейронних мереж полягає в чергуванні згорткових шарів (англ. Convolution layers) і субдискретизуючих шарів (англ. Subsampling layers або англ. Pooling layers, шарів підвибірки).

Структура мережі - односпрямована (без зворотних зв'язків), принципово багатошарова. Для навчання використовуються стандартні методи, найчастіше метод зворотного поширення помилки. Функція активації нейронів (передавальна функція) - будь-яка, за вибором дослідника.

Назву архітектура мережі отримала через наявність операції згортки, суть якої полягає в тому, що кожен фрагмент зображення множиться на матрицю (ядро) згортки поелементно, а результат підсумовується і записується в аналогічну позицію вихідного зображення.

Робота згорткової нейронної мережі зазвичай інтерпретується як перехід від конкретних особливостей зображення до більш абстрактних деталей, і далі до ще більш абстрактних деталей аж до виділення понять високого рівня. При цьому мережа самоналаштовується і сама виробляє необхідну ієрархію абстрактних ознак (послідовності карт ознак), фільтруючи незначні деталі і виділяючи істотне.

У звичайному перцептроні, який представляє собою повнозв'язну нейронну мережу, кожен нейрон пов'язаний з усіма нейронами попереднього шару, причому кожний зв'язок має свій персональний ваговий коефіцієнт.

У згорткової нейронної мережі в операції згортки використовується лише обмежена матриця ваг невеликого розміру, яку «рухають» по всьому оброблюваного шару (на самому початку - безпосередньо по вхідному зображенню), формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічною позицією. Тобто для різних нейронів вихідного шару використовується одна і та ж матриця ваг, яку також називають ядром згортки. Її інтерпретують як графічне кодування якої-небудь ознаки, наприклад, наявність похилої лінії під певним кутом. Тоді наступний шар, отриманий в результаті операції згортки такою матрицею ваг, показує наявність даної ознаки в оброблюваному шарі і її координати, формуючи так звану карту ознак (англ. Feature map).

Згорткова нейронна мереже має не один набір ваг, а цілу гама, що кодує елементи зображення (наприклад лінії і дуги під різними кутами). При цьому такі ядра згортки не закладаються дослідником заздалегідь, а формуються самостійно шляхом навчання мережі класичним методом зворотного поширення помилки.

Операція субдискретизації (англ. Subsampling, англ. Pooling, також перекладається як «операція підвибірки» або операція об'єднання), виконує зменшення розмірності сформованих карт ознак. У даній архітектурі мережі вважається, що інформація про факт наявності шуканої ознаки важливіше точного знання його координат, тому з кількох сусідніх нейронів карти ознак вибирається максимальний і приймається за один нейрон ущільненої карти ознак меншої розмірності. За рахунок цієї операції, крім прискорення подальших обчислень, мережа стає більш інваріантною до масштабу вхідного зображення.

Структура згорткової нейронної мережі зображена на рис. 2.5.

Мережа складається з великої кількості шарів. Після початкового шару (вхідного зображення) сигнал проходить серію згорткових шарів, в яких чергується власне згортка і субдискретизація (пулінг). На кожному наступному шарі карта зменшується в розмірі, але збільшується кількість каналів. На практиці це означає здатність розпізнавання складних ієрархій ознак. Зазвичай після проходження декількох шарів карта ознак вироджується в вектор або навіть скаляр, але таких карт ознак утворюються сотні. На виході згорткових шарів мережі додатково встановлюють кілька шарів повнозв’язної нейронної мережі (перцептрон), на вхід якого подаються кінцеві карти ознак.

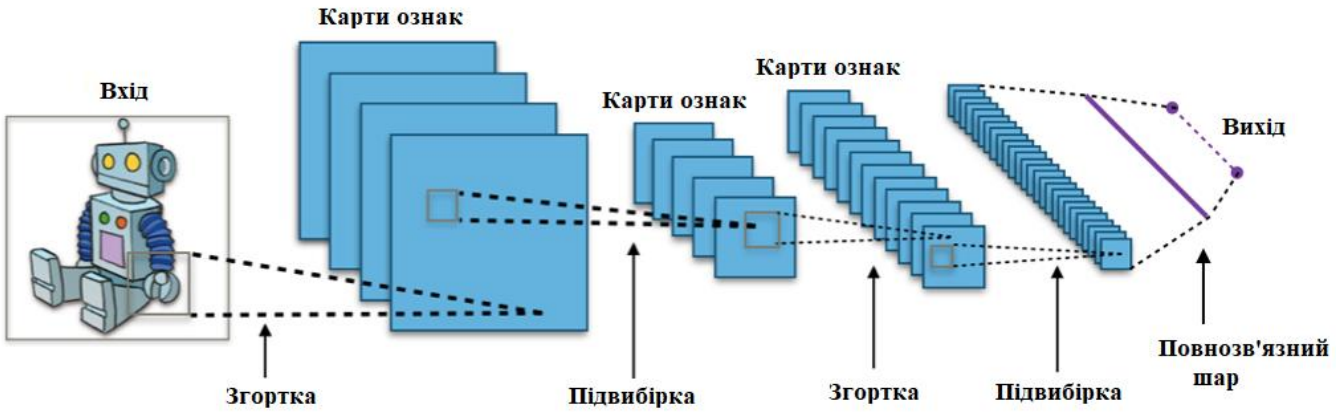

Рис. 2.5. Типова структура згорткової нейронної мережі

Переваги згорткової нейронної мережі:

- один з найкращих алгоритмів розпізнавання і класифікації зображень;
- в порівнянні з повнозв’язною нейронною мережею (по типу перцептрона) має набагато меншу кількість налаштовуваних ваг, оскільки одне ядро ваг може використовуватись для всього зображення, а не для кожного пікселя окремо;
- зручне розпаралелювання обчислень, а в результаті – можливість реалізації алгоритмів роботи і навчання мережі на графічних процесорах;
- відносна стійкість до повороту та зсуву зображення, що розпізнається;

- навчання за допомогою класичного методу зворотного поширення помилки.

Основним неделіком згорткової нейронної мережі є занадто велика кількість варіативних параметрів (кількість шарів, розмірність ядра згортки, кількість ядер для кожного з шарів, крок зсуву ядра при обробці шару, необхідність шарів субдискретизації, ступінь зменшення розмірності, функція зменшення розмірності, передаточна фунція нейронів, наявність і параметри вихідної повнозв’язної нейронної мережі тощо). Всі ці параметри суттєво впливають на результат, але обираються дослідниками емпіричним шляхом [33].

AlexNet - згорткова нейронна мережа, яка здійснила великий вплив на розвиток машинного навчання, особливо - на алгоритми комп'ютерного зору. Мережа з великим відривом виграла конкурс з розпізнавання зображень ImageNet LSVRC-2012 у 2012 році (з кількістю помилок 15,3% проти 26,2% у другого місця) [34].

Архітектура AlexNet схожа на створену Yann LeCum мережу LeNet. Однак у AlexNet більше фільтрів на шарі і вкладених згорткових шарів. Мережа включає в себе згортки, максимальне об'єднання, дропаут, аугментацію даних, функції активацій ReLU і стохастичний градієнтний спуск.

Особливості AlexNet:

1) У якості функції активації використовується Relu замість арктангенса для додавання в модель нелінійності. За рахунок цього при однаковій точності методу швидкість стає в 6 разів вищою.
2) Використання дропаутів замість регуляризації вирішує проблему перенавчання. Однак час навчання подвоюється з показником дропаута 0,5.
3) Проводиться перекриття об'єднань для зменшення розміру мережі. За рахунок цього рівень помилок першого і п'ятого рівнів знижуються до 0,4% і 0,3%, відповідно.

Архітектура мережі AlexNet наведена на рис. 2.6.

AlexNet містить вісім шарів з ваговими коефіцієнтами. Перші п'ять з них згорткові, а решта три - повнозв'язні. Вихідні дані пропускаються через функцію

втрат softmax, яка формує розподіл 1000 міток класів. Мережа максимізує багатолінійну логістичну регресію, що еквівалентно максимізації середнього по всім навчальним випадкам логарифма ймовірності правильного маркування з розподілу очікування. Ядра другого, четвертого і п'ятого згортальних шарів пов'язані тільки з тими картами ядра в попередньому шарі, які знаходяться на одному і тому ж графічному процесорі. Ядра третього згорткового шару пов'язане з усіма картами ядер другого шару. Нейрони в повнозв'язних шарах пов'язані з усіма нейронами попереднього шару.

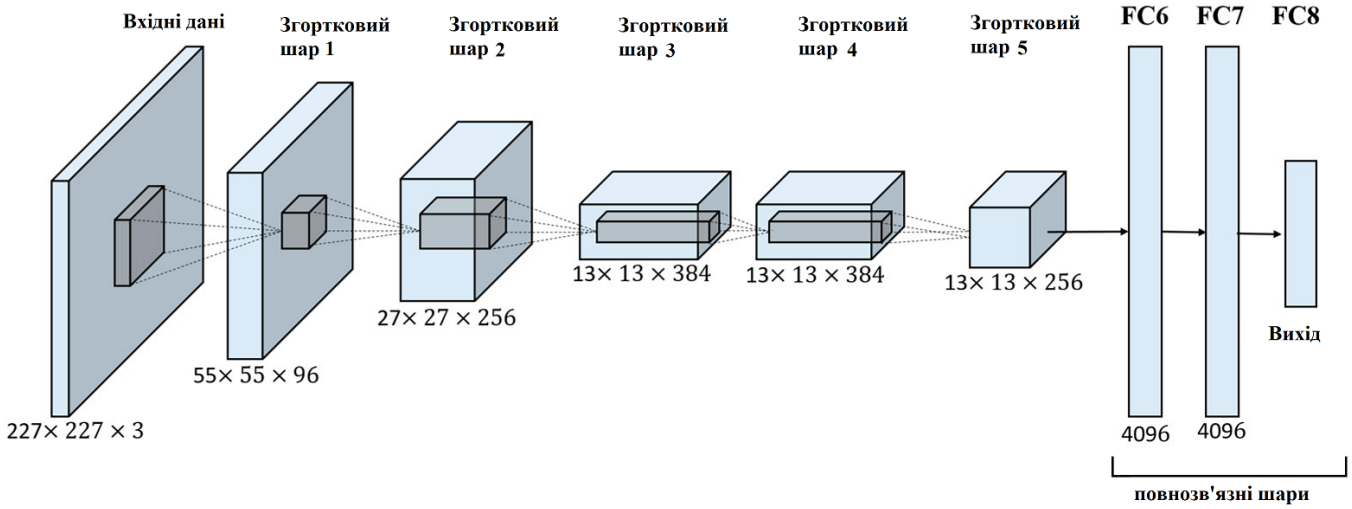

Рис. 2.6. Архітектура згорткової мережі AlexNet

Таким чином, AlexNet містить 5 згорткових шарів і 3 повнозв'язних шари. Relu застосовується після кожного згорткового і повнозв'язного шару. Дропаут застосовується перед першим і другим повнозв'язними шарами. Мережа містить 62,3 мільйона параметрів і витрачає 1,1 мільярда обчислень при прямому проході. Згорткові шари, на які припадає 6% всіх параметрів, здійснюють 95% обчислень.

Результати показують, що більша, глибока сверточное нейронна мережа здатна досягати рекордних результатів на дуже складних наборах даних, використовуючи тільки навчання з учителем [34].

2.3 Методи співставлення біометричних образів і криптографічних ключів

Співставлення компактних векторів біометричних образів і криптографічних ключів реалізовано за допомогою наступних методів:

1) побудова і навчання двонаправленої асоціативної пам’яті на основі нейронної мережі Б. Коско (bidirectional associative memory, BAM) [35];
2) побудова і навчання одношарових і багатошарових нейронних мереж прямого розповсюдження на основі перцептронів (БШП).

2.3.1 Двонаправлена асоціативна пам’ять

Штучна нейронна мережа зі зворотним зв'язком формує асоціативну пам'ять.

Автоасоціативна пам’ять - це пам’ять, яка може доповнити або виправити зображення, але не може пов’язати отримане зображення з іншим зображенням. Цей факт є результатом однорівневої структури асоціативної пам’яті, в якій вектор з’являється на виході тих самих нейронів, які приймають вхідний вектор. Такі мережі нестабільні. Для стабільної мережі послідовні ітерації призводять до все менших змін випуску, поки з часом вихід не стає постійним. Для багатьох мереж процес ніколи не закінчується. Нестабільні мережі мають цікаві властивості і вивчались як приклад хаотичних систем. У певному сенсі цього можна досягти без зворотного зв’язку, наприклад, перцептроно для випадків, коли стабільність важливіша за вивчення хаотичних систем.

Гетероасоціативна пам’ять - це пам’ять, в якій при надходженні подразника до одного набору нейронів реакція зворотного зв’язку з’являється на іншому наборі нейронів.

Перша модель автоасоціативної пам'яті була розроблена Хопфілдом - Нейронна мережа Хопфілда. Щоб отримати стійкість, довелось вагові коефіцієнти підбирати таким чином, щоб сформувати енергетичні мінімуми в необхідних вершинах одиничного гіперкуба.

В результаті Коско розвив ідеї Хопфілди та розробив модель гетероассоціативної пам'яті - двонаправлену асоціативну пам'ять (ДАП) [36].

Нейронна мережа Коско, двонаправлена асоціативна пам’ять (ДАП) - це нейронна мережа зі зворотним зв'язком, заснована на двох ідеях: теорії адаптивного резонансу Стефана Гросберга та автоасоціативній пам'яті Хопфілда.

ДАП э гетероасоціативною: вхідний вектор подається до одного набору нейронів, а відповідний вихідний вектор генерується іншим набором нейронів. Як і мережа Хопфілда, двонаправлена асоціативна пам’ять здатна до узагальнення шляхом отримання правильних реакцій попри спотворені вхідні дані. Крім того, можуть бути реалізовані адаптивні версії ДАП, які відокремлюють еталонний образ із екземплрів з завадами. Такі можливості сильно нагадують процес мислення людини і дозволяють штучним нейронним мережам зробити крок в напрямку до моделювання мозку людини [37].

Архітектура двонаправленої асоціативної пам’яті зображена на рис. 2.7.

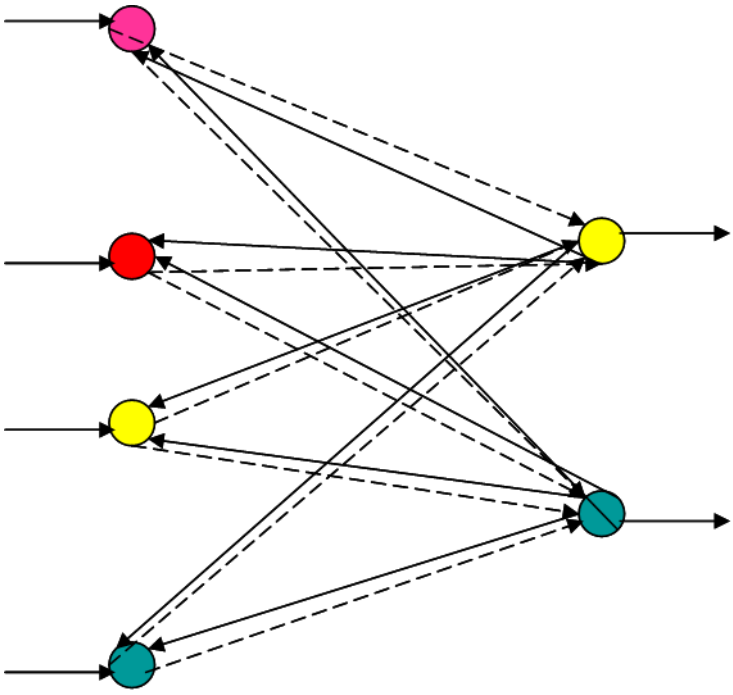

Рис. 2.7. Архітектура нейронної мережі Коско

Можна виділити наступні види двонаправленої асоціативної пам’яті:

1) синхронна ДАП;
2) неперервна ДАП;
3) адаптивна ДАП;
4) конкуруюча ДАП.

Синхронна ДАП. Мережа складається з двох шарів елементів, між якими існують двосторонні зв’язки, пов’язані з елементами за принципом всі до всіх. Тому для опису мережі може використовуватись матриця ваг. У випадку, коли ця матриця є квадратною та симетричною, ДАП перетворюється на авто-асоціативну мережу Хопфілда. Коли на вхід мережі подається сигнал з шумом, передача сигналів всередині ДАП відбувається доти, поки мережа не знайде найближчий еталон (асоціацію), котрому її раніше навчали. Цей процес можна трактувати як спогади та стабілізацію пам’яті.

Неперевна ДАП. Неперервні асинхронні ДАП відхиляють синхронність і розривність, але функціонують в основному аналогічно дискретним версіям. У синхронній ДАП формальні нейрони в шарах 1 і 2 синхронні, тобто кожен нейрон володіє пам'яттю, причому всі нейрони змінюють стан одночасно. У асинхронній системі будь-який нейрон здатен змінити стан у будь-який час, коли його вхідні дані це передбачають.

Адаптивна ДАП. Адаптивна ДАП змінює свої ваги в процесі функціонування. Це означає, що подача даних на вході в мережі навчального набору вхідних векторів забороняє зменшувати енергетичний стан до отримання резонансу. Поступова короткочасна пам'ять перетворюється в довгострокову пам'ять, налаштовуючи мережу в результаті функціонування. У процесі навчання вектори подаються на шар А, а асоціативні вектори на шар В. Один з них або обидва можуть бути зашумленими версіями еталона; мережа навчається обучається вхідним вектором, вільним від шуму. У цьому випадку вона освоює сутність асоціацій, навчаючись еталонам, хоча «бачила» лише зашумлені аппроксимації.

Конкуруюча ДАП. У багатьох конкуруючих нейронних системах спостерігаються деякі види конкуренції між нейронами. У ДАП конкуренція реалізується взаємним з'єднанням нейронів всередині кожного шару за допомогою додаткових зв'язків. Ваги цих зв’язків формують іншу матрицю вагів з позитивними значеннями елементів головної діагоналі та негативними значеннями всіх інших елементів.

2.3.2 Нейронні мережі прямого розповсюдження на основі перцептронів

Нейронна мережа прямого розповсюдження (feedforward neural network) - це класичний тип нейронної мережі, де зв’язки між нейронами не утворюють замкнених циклів. Нейронні мережі прямого розповсюдження - це перший і найпростіший тип класичного проектування нейронних мереж. У цій мережі інформація поширюється лише в одному напрямку від входів через приховані шари (якщо такі є) до виходів. У той же час у мережі відсутні цикли.

Структура нейронної мережі прямого розповсюдження представлена на рис. 2.8.

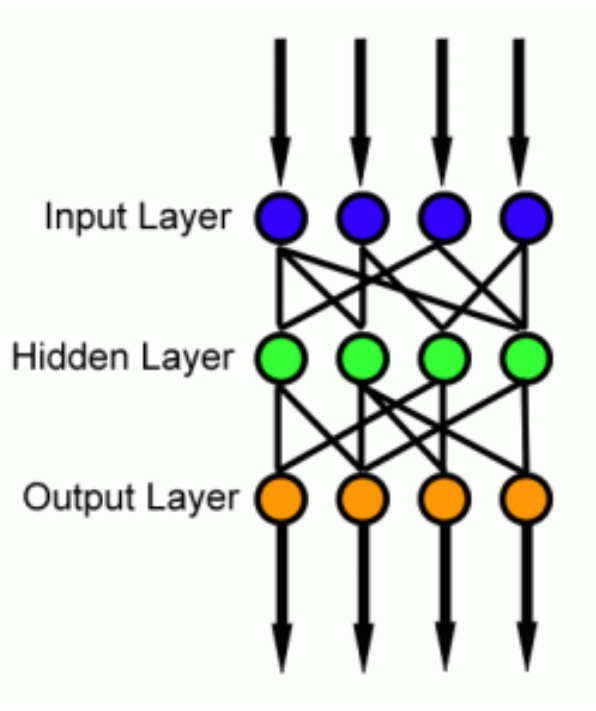

Рис. 2.8. Структура нейронної мережі прямого розповсюдження

Одношаровий перцептрон. Найпростіший вид нейронної мережі прямого розповсюдження - це одношарова мережа перцептронів, яка складається з одного шару вихідних вузлів; входи подаються безпосередньо на виходи через ряд ваг. Сума добутків ваг та входів обчислюється в кожному вузлі, і якщо значення перевищує деякий поріг (зазвичай 0), нейрон спрацьовує і приймає активоване значення (як правило, 1); інакше приймає деактивоване значення (зазвичай -1). Нейрони з таким видом активаційної функції також називають штучними

нейронами або лінійними пороговими одиницями. Подібний нейрон описали Уоррен Маккалок і Уолтер Піттс в 1940-х роках.

Персептрон можна створити, використовуючи будь-які значення для активованого та деактивованого станів, якщо порогове значення лежить між ними.

Тренувати перцептрони можна за допомогою простого алгоритму навчання, який зазвичай називають правилом дельта. Цей алгоритм обчислює помилки між обчисленими вихідними даними та вихідними даними зразку, і використовує їх для коригування ваг, реалізуючи таким чином форму градієнтного спуску.

Одношарові перцептрони здатні навчатися лише за лінійно відокремлюваними шаблонами; у 1969 р. у відомій монографії під назвою "Перцептрони" Марвін Мінський та Сеймур Паперт показали, що одношарову перцептронну мережу неможливо навчити функції XOR (тим не менш було відомо, що багатошарові перцептрони здатні відтворити будь-яку можливу булеву функцію).

Хоча одна порогова одиниця є досить обмеженою у своїй обчислювальній потужності, було показано, що мережі паралельних порогових одиниць можуть апроксимувати будь-яку безперервну функцію з компактного інтервалу дійсних чисел в інтервал [-1,1]. Цей результат можна знайти у Пітера Ауера, Харальда Бургштайнера та Вольфганга Маасса "Правило навчання для дуже простих універсальних апроксиматорів, що складаються з одного шару персептронів" [38].

Багатошаровий перцептрон. Цей клас мереж складається з декількох шарів обчислювальних одиниць, як правило, взаємопов’язаних у зворотному напрямку. Кожен нейрон в одному шарі має спрямовані зв’язки з нейронами наступного шару. У багатьох додатках одиниці цих мереж використовують сигмоїдальну функцію у якості функції активації.

Теорема універсального наближення для нейронних мереж стверджує, що кожна безперервна функція, яка відображає інтервали дійсних чисел на деякий вихідний інтервал дійсних чисел, може бути довільно апроксимована багатошаровим перцептроном лише з одним прихованим шаром. Цей результат

справедливий для широкого кола функцій активації, наприклад для сигмоїдальних функцій.

Багатошарові мережі використовують різноманітні методи навчання, найпопулярнішим є зворотне розповсюдження. Тут вихідні значення порівнюються з правильною відповіддю для обчислення значення деякої заздалегідь визначеної функції помилки. Потім різними методами помилка передається назад через мережу. Використовуючи цю інформацію, алгоритм регулює ваги кожного з'єднання, щоб зменшити значення функції помилки на деякий невеликий розмір. Після повторення цього процесу протягом досить великої кількості навчальних циклів мережа, як правило, сходиться до певного стану, де похибка розрахунків невелика. У цьому випадку можна сказати, що мережа вивчила певну цільову функцію. Для правильного регулювання ваг застосовується загальний метод нелінійної оптимізації, який називається градієнтним спуском. Для цього мережа обчислює похідну функції помилки відносно вагових коефіцієнтів мережі і змінює вагові коефіцієнти таким чином, що помилка зменшується (таким чином, йде вниз по поверхні функції помилки). З цієї причини зворотне поширення може застосовуватися лише в мережах з різними функціями активації.

Загалом, проблема навчання мережі коректно працювати, навіть на зразках, які не використовувались як навчальні, є досить вагомою проблемою, яка вимагає додаткових методів. Це особливо важливо для випадків, коли доступна лише дуже обмежена кількість навчальних зразків [39]. Небезпека полягає в тому, що мережа переорієнтує навчальні дані і не зможе вловити справжній статистичний процес, що генерує дані. Теорія обчислювального навчання стосується підготовки класифікаторів на обмеженій кількості даних. У контексті нейронних мереж проста евристика, яка називається ранньою зупинкою, часто гарантує, що мережа буде добре узагальнюватись до зразків, які не входять до навчального набору.

Іншими типовими проблемами алгоритму зворотного розповсюдження є швидкість збіжності та можливість опинитися в локальному мінімумі функції помилки. Сьогодні існують практичні методи, які роблять зворотне розповсюдження

в багатошарових перцептронах інструментом вибору для багатьох завдань машинного навчання.

2.4 Висновки до розділу 2

Структура системи автетнифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ включає в себе 5 основних етапів:

1) попередня обробка ознак;
2) генерація ознак;
3) створення бази біометричних зразків;
4) нейромережеве співставлення образів і ключів;
5) відновлення ключа.

Задача блоку попередньої обробки ознак полягає в уніфікації зображення обличчя людини і побудові первинного вертору ознак.

Генерація ознак полягає в проекуванні первинного вектору на новий простір ознак і формуванні компактного вектору ознак кожного образу для подальшої нейромережевої обробки. Цей вектор подається в нейромережевий блок у формі двійкового вектору або ж у формі цілочисельного вектору, який, в свою чергу, генерує закритий ключ, що подається на вхід модуля криптографічної системи.

Для генерації ознак пропонується використати наступні підходи:

- двовимірна карта самоорганізації Кохонена (self-organizing map, SOM);
- ймовірнісний метод головних компонент (ЙМГК, probabilistic principal component analysis, PPCA);
- згорткова нейронна мережа (convolutional neural network, CNN) AlexNet.

У випадку, коли потрібно додати користувача в систему, компактний вектор ознак подається на вхід блоку створення біометричних образів, де виконується нейромережеве співставлення закритого ключа і вектора, що подається на вхід.

Співставлення компактних векторів біометричних образів і криптографічних ключів реалізовано за допомогою наступних методів:

- побудова і навчання двонаправленої асоціативної пам’яті на основі нейронної мережі Б. Коско (bidirectional associative memory, BAM);
- побудова і навчання одношарових і багатошарових нейронних мереж прямого розповсюдження на основі перцептронів (БШП).

У результаті порівняльного аналізу алгоритмів вилучення первинних біометричних ознак і співставлення сформованого образу з закритим ключем в рамках запропонованої системи автентифікації встановлено, що найбільш ефективним на тестових даних є підхід з використанням глибоких згорткових мереж (AlexNet) і нейромережевої двонаправленої гетероасоціативної пам'яті (BAM).

Блок відновлення ключа необхідний для трансформації компактного вектора ознак в необхідний ключ користувача. Слід відзначити, що при спробі несанкціонованої авторизації система видає випадковий ключ, який в системі не використовується.

Розділ 3. РОЗРОБКА АЛГОРИТМУ ТА ПРОГРАМНОГО МОДУЛЮ АВТЕНТИФІКАЦІЇ

3.1 Формулювання вимог до розробки та впровадження власного програмного модулю автентифікації

Метою дипломної роботи є створення надійної системи автентифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ.

Функціональні вимоги:

1) При першому відвідуванні вебсторінки користувач повинен мати змогу зареєструватися в системі, якщо він ще не має облікового запису;
2) Зареєстрований користувач має змогу увійти в систему, попередньо надавши дозвіл на використання веб-камери для отримання біометричного зразку;
3) Після надання біометричного зразку користувачу на екран виводиться повідомлення, яке містить інформацію про вдалість спроби автентифікації (успіх або невдача).

Системні вимоги: програмний модуль розробляється у вигляді вебсторінки, яку можна відкрити в будь-якому сучасному браузері. Також сторінку можна переглянути з мобільних пристроїв та планшетів.

Описані вище випадки, що можуть відбутися в системі, зображені на use case діаграмі, побудованій засобами UML (рис. 3.1).

Можливі стани в системі зображені на UML stateChart діаграмі (рис 3.2). Пояснення stateChart діаграми:

1. При вході в систему користувач бачить початкову сторінку, де йому пропонується або увійти в систему.
2. Система робить запит на дозвіл використання веб-камери для отримання біометричного зразку.

3. В разі, якщо доступ до камери надано, система виконує процес розпізнавання та автентифікації.
4. У випадку успішної автентифікації користувач потрапляє в систему і отримує відповідне повідомлення.

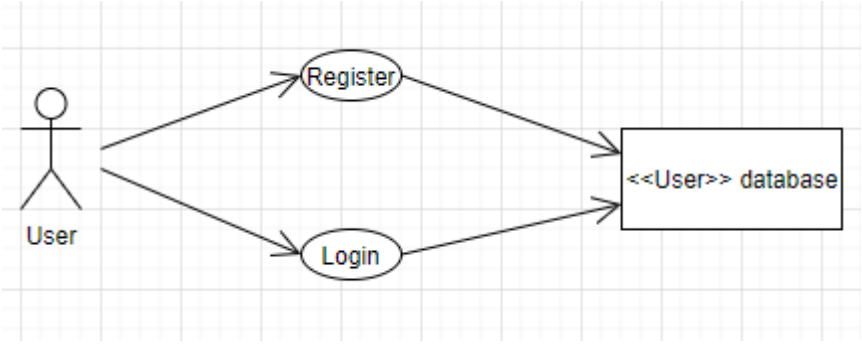

Рис. 3.1. Use case діаграма

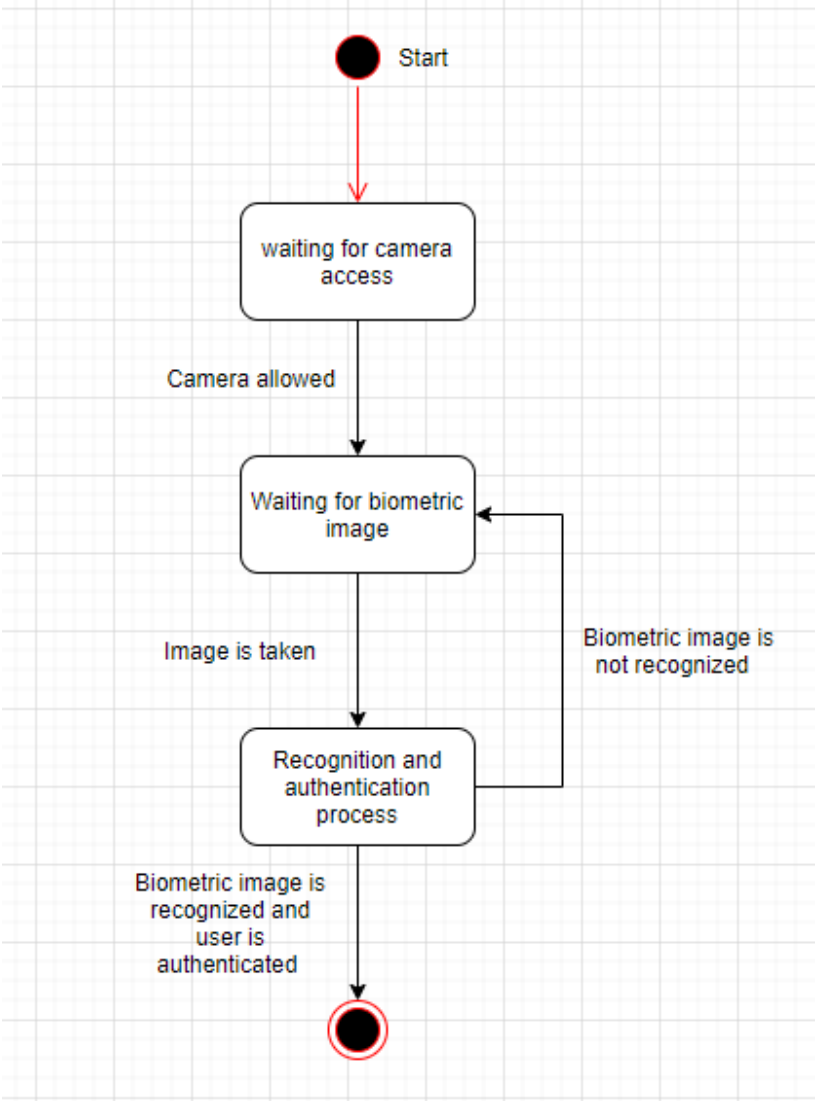

Рис. 3.2. Діаграма станів

На рис. 3.3 зображена діаграма активностей. Пояснення до діаграми:

У неавторизованого користувача є 2 вибори: увійти в існуючий обліковий запис або створити новий.

При створенні нового облікового запису:

1. Користувач вводить свої дані (в даному випадку – користувацьке ім’я).
2. Система показує користувачеві запит на використання камери.
3. Якщо дозвіл не надано – повторюємо п. 2.
4. Користувач робить знімок.
5. Система додає біометричний зразок користувача до бази.
6. Система показує користувачеві повідомлення про успішну реєстрацію.

Якщо користувач хоче увійти в існуючий обліковий запис:

1. Система показує користувачеві запит на використання камери.
2. Якщо дозвіл не надано – повторюємо п. 1.
3. Користувач робить знімок.
4. Система проводить розпізнавання користувача за існуючими в базі біометричними зразками.
5. Якщо користувача не було розпізнано – повертаємось до п. 3.
6. Система показує користувачеві повідомлення про успішний вхід.

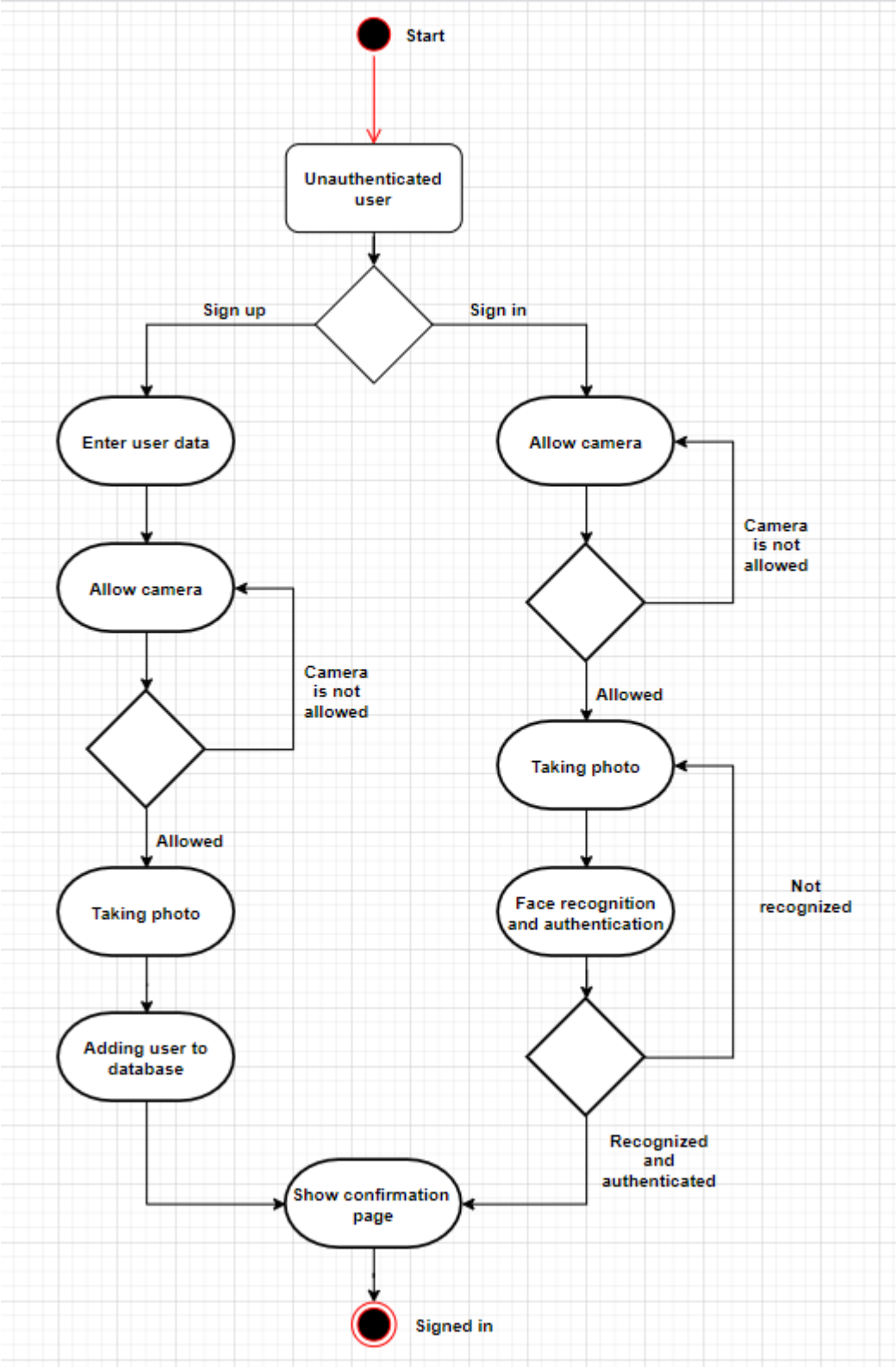

Рис. 3.3. Діаграма активностей

3.2 Розробка алгоритму роботи програмного модулю

3.2.1 Інструменти розробки

Для реалізації програмного модулю автентифікації було обрано наступні інструменти для розробки програмного забезпечення:

1) Середовище розробки: Visual Studio Code – редактор вихідного коду, розроблений Microsoft для Windows, Linux і macOS. Позиціонується як «легкий» редактор коду для кроссплатформенної розробки веб- і хмарних додатків. Включає в себе відладчик, інструменти для роботи з Git, підсвічування синтаксису, IntelliSense і засоби для рефакторинга. Має широкі можливості для кастомізації: призначені для користувача теми, поєднання клавіш і файли конфігурації. Розповсюджується безкоштовно, розробляється як програмне забезпечення з відкритим вихідним кодом, але готові збірки розповсюджуються під проприєтарною ліцензією;
2) Мова програмування на стороні серверу: Python (з використанням різних бібліотек для роботи з нейронними мережами) – високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності розробника і читання коду. Синтаксис ядра Python мінімалістичний. У той же час стандартна бібліотека включає великий набір корисних функцій. Python підтримує структурне, узагальнене, об'єктно-орієнтоване, функціональне і аспектно-орієнтоване програмування. Основні архітектурні риси - динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточних обчислень, високорівневі структури даних. Підтримується розбиття програм на модулі, які, в свою чергу, можуть об'єднуватися в пакети;
3) Еко-система для розробки користувацького інтерфейсу (front-end): Angular 7, HTML 5, SCSS, Bootstrap. Angular – JavaScript фреймворк, що реалізує компонентний підхід, підтримує типізацію даних (TypeScript) та містить в собі велику кількість готових рішень для вирішення стандартних задач. Зокрема, для розмежування доступу до ресурсів на стороні клієнта був використаний вбудований API (application programming interface) для маршрутизації та

навігації по вебсторінці – Angular Auth guard. Даний інтерфейс допомагає розмежувати доступ користувача до ресурсів в залежності від його ролі;

4) База даних: PostgreSQL – реляційна база даних, що має багато вбудованих можливостей і підтримує складні структури, а також приділяє значну увагу збереження цілісності даних в таблицях.

Тип архітектури програмного модулю – клієнт-серверний, оскільки даний модуль розрахований для Інтернет-ресурсів.

3.2.2 Опис алгоритму роботи програмного модулю

3.2.2.1 Попередня обробка ознак

Першочергова задача попередньої обробки ознак – виділення зони інтересу, тобто обличчя, за яким проводимо автентифікацію. Також зображення для збільшення швидкості роботи необхідно нормалізувати, масштабувати зображення та трансформувати його у відтінки сірого. Для виконання цих задач було обрано бібліотеку OpenCV.

OpenCV - це бібліотека з відкритим кодом, яка призначена для аналізу, класифікації та обробки зображень. Широко використовується в таких мовах як C, C ++, Python та Java.

Процес попередньої обробки зображень наступний:

```
def faceCrop(pil_im):
   boxScale=1
   faceCascade = cv.Load('haarcascade_frontalface_alt.xml')
   cv_im=pil2cvGrey(pil_im)
      face=DetectFace(cv_im,faceCascade)
      if face:
         croppedImage=imgCrop(pil_im, face[0],boxScale=boxScale)
      else:
         print 'No faces found:', img
```

Тут pil_im – вхідне зображення, cv_im – зображення переконвертоване в чорно-білу колірну схему, faceCascade – попередньо заготовлений шаблон, за яким

проводиться розпізнавання, face_im – послідовності прямокутних зон, повернутих після розпізнавання, croppedImage – цільове зображення.

Метод, що конвертує PIL зображення в чорно-біле CV зображення:

```
def pil2cvGrey(pil_im):
    pil_im = pil_im.convert('L')
    cv_im = cv.CreateImageHeader(pil_im.size, cv.IPL_DEPTH_8U, 1)
    cv.SetData(cv_im, pil_im.tostring(), pil_im.size[0]  )
    return cv_im
```

Метод для розпізнавання обличчя зазначений нижче. Він приймає на вході зображення в сірих тонах і виділяє на цьому зображенні обличчя за шаблонами, вказаними в haarcascade.

```
def DetectFace(image, faceCascade):
    min_size = (20,20)
    haar_scale = 1.1
    min_neighbors = 3
    haar_flags = 0
    cv.EqualizeHist(image, image)
    face = cv.HaarDetectObjects(
          image, faceCascade, cv.CreateMemStorage(0),
          haar_scale, min_neighbors, haar_flags, min_size
       )
     return face
```

Для підвищення контрасності зображення і розширення діапазону інтенсивності було застосовано прийом вирівнювання гістограми (cv.EqualizeHist).

Функція cv.HaarDetectObjects знаходить прямокутні області на даному зображенні, які, ймовірно, можуть містити обличчя, і повертає ці області у вигляді послідовності прямокутників.

Обрізаємо початкове зображення відповідно до отриманої послідовності прямокутниців з наявним обличчям:

```
def imgCrop(image, cropBox, boxScale=1):
```

```
xDelta=max(cropBox[2]*(boxScale-1),0)
yDelta=max(cropBox[3]*(boxScale-1),0)
PIL_box=[cropBox[0]-xDelta, cropBox[1]-yDelta, cropBox[0]+cropBox[2]+xDelta, cropBox[1]+cropBox[3]+yDelta]
return image.crop(PIL_box)
```

3.2.2.2 Генерація ознак

Для генерації біометричних ознак було обрано підхід використання згорткової мережі AlexNet.

Починаємо реалізацію з імпорту наступних бібліотек:

- TensorFlow: Платформа з відкритим кодом для впровадження, навчання та розгортання моделей машинного навчання.
- Keras: Бібліотека з відкритим кодом, що використовується для реалізації архітектур нейронних мереж, що працюють як на центральних, так і на графічних процесорах.
- Matplotlib: інструмент візуалізації python, який використовується для ілюстрування інтерактивних діаграм та зображень.

```
import tensorflow as tf
from tensorflow
import keras
import matplotlib.pyplot as plt
import os
import time
```

Навчаємо мережу на підготовленому попередньо наборі даних. Для посилання на імена класів зображень на етапі візуалізації, список, що містить класи, ініціалізується іменем змінної CLASS_NAMES:

```
CLASS_NAMES= ['face', 'other']
```

Набір даних розділений на 50000 навчальних даних та 10000 тестових даних. Останнім розділом набору даних, який нам потрібен, є дані валідації. Дані валідації беруться з останніх 5000 зображень у рамках навчальних даних.

validation_images, validation_labels = train_images[:5000], train_labels[:5000]

train_images, train_labels = train_images[5000:], train_labels[5000:]

TensorFlow надає набір функцій та операцій, що дозволяють легко обробляти та модифікувати дані за допомогою певного вхідного конвеєра.

Щоб мати доступ до цих методів та процедур, потрібно перетворити наш набір даних в ефективне представлення даних, з яким знайомий TensorFlow. Це досягається за допомогою API tf.data.Dataset.

train_ds = tf.data.Dataset.from_tensor_slices ((train_images, train_labels))

test_ds = tf.data.Dataset.from_tensor_slices ((test_images, test_labels))

validation_ds = tf.data.Dataset.from_tensor_slices ((validation_images, validation_labels))

Набір навчальних даних: Це група набору даних, що використовується для безпосередньої підготовки нейронної мережі.

Набір даних валідкації: Ця група набору даних використовується під час навчання для оцінки ефективності роботи мережі на різних ітераціях.

Тестовий набір даних: Цей розділ набору даних оцінює ефективність нашої мережі після завершення фази навчання.

Наступним кроком є побудова вхідного конвеєру.

Вхідний ковеєр описується як набір функцій або методів, які викликаються послідовно один за одним. Вхідні конвеєри - це ланцюг функцій, які або діють на дані, або виконують операцію над даними, що протікають через конвеєр.

Визначаємо розмір кожного розділу набору даних, який ми створили; розміри розділів набору даних необхідні, щоб забезпечити ретельне перемішування набору даних перед передачею в мережу.

train_ds_size = tf.data.experimental.cardinality(train_ds).numpy()

test_ds_size = tf.data.experimental.cardinality(test_ds).numpy()

validation_ds_size = tf.data.experimental.cardinality(validation_ds).numpy()

print("Training data size:", train_ds_size)

print("Test data size:", test_ds_size)

print("Validation data size:", validation_ds_size)

Для базового вхідного конвеєру ми проведемо три основні операції:

1. Попередня обробка даних
2. Перемішування даних
3. Пакетування набору даних

```
train_ds = (train_ds
            .map(process_images)
            .shuffle(buffer_size=train_ds_size)
            .batch(batch_size=32, drop_remainder=True))
test_ds = (test_ds
            .map(process_images)
            .shuffle(buffer_size=train_ds_size)
            .batch(batch_size=32, drop_remainder=True))
validation_ds = (validation_ds
            .map(process_images)
            .shuffle(buffer_size=train_ds_size)
            .batch(batch_size=32, drop_remainder=True))
```

Впроваджуємо архітектуру AlexNet CNN з нуля.

Завдяки використанню Keras Sequential API, ми можемо реалізувати послідовні рівні нейронної мережі в наших моделях, які розташовані один біля одного.

Ось типи шарів, з яких складається архітектура AlexNet CNN, разом із коротким описом:

Згортковий шар: згортка - це математичний термін, який описує точкове множення продукту між двома наборами елементів. При глибокому навчанні операція згортки діє на фільтри / ядра та масив даних зображень у згортковому шарі. Тому згортковий шар - це просто шар, в якому розміщена операція згортки, яка відбувається між фільтрами та зображеннями, що проходять через згорткову нейронну мережу.

Шар пакетної нормалізації: пакетна нормалізація - це техніка, яка пом'якшує вплив нестабільних градієнтів у нейронній мережі шляхом введення додаткового

рівня, який виконує операції над входами попереднього рівня. Операції стандартизують та нормалізують вхідні значення, після чого вхідні значення трансформуються за допомогою операцій масштабування та зсуву.

Шар MaxPooling: Максимальне об’єднання - це варіант субдискретизації, де максимальним значенням пікселів, що потрапляють у сприйнятливе поле одиниці в шарі субдискретизації, приймається за вихід. Наведена нижче операція максимального об’єднання має вікно 2x2 і ковзає по вхідних даних, видаючи середнє значення пікселів у сприйнятливому полі ядра.

Згладжувальний шар: приймає вхідну форму та згладжує дані вхідного зображення в одновимірний масив.

Щільний шар: щільний шар має вбудовану кількість довільних одиниць / нейронів всередині. Кожен нейрон - це перцептрон.

Деякі інші операції та методи, що використовуються в AlexNet CNN, про які варто згадати, є:

Функція активації: математична операція, яка перетворює результат або сигнали нейронів у нормалізований вихід. Призначення функції активації як компонента нейронної мережі полягає у впровадженні нелінійності всередині мережі. Включення функції активації дозволяє нейронній мережі мати більшу репрезентативну силу та вирішувати складні функції.

Випрямлена лінійна функція активації одиниць (ReLU): Тип функції активації, яка трансформує значення нейрона. Перетворення, накладене ReLU на значення з нейрона, представляється формулою y = max (0, x). Функція активації ReLU зменшує будь-які негативні значення з нейрона до 0, а позитивні значення залишаються незмінними. Результат цього математичного перетворення використовується як вихід поточного рівня і використовується як вхід для послідовного шару в нейронній мережі.

Функція активації Softmax: Тип функції активації, яка використовується для отримання розподілу ймовірності набору чисел у вхідному векторі. Результатом роботи функції активації softmax є вектор, у якому набір значень представляє ймовірність появи класу чи події. Значення всередині вектора складають 1.

Випадання: Техніка випадання працює шляхом випадкового зменшення кількості взаємопов’язаних нейронів у нейронній мережі. На кожному етапі навчання кожен нейрон має шанс бути виключеним.

Фрагмент коду представляє реалізацію Keras архітектури AlexNet CNN:

```
model = keras.models.Sequential([
    keras.layers.Conv2D(filters=96, kernel_size=(11,11), strides=(4,4), activation='relu', input_shape=(227,227,3)),
    keras.layers.BatchNormalization(),
    keras.layers.MaxPool2D(pool_size=(3,3), strides=(2,2)),
    keras.layers.Conv2D(filters=256, kernel_size=(5,5), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.MaxPool2D(pool_size=(3,3), strides=(2,2)),
    keras.layers.Conv2D(filters=384, kernel_size=(3,3), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.Conv2D(filters=384, kernel_size=(1,1), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.Conv2D(filters=256, kernel_size=(1,1), strides=(1,1), activation='relu', padding="same"),
    keras.layers.BatchNormalization(),
    keras.layers.MaxPool2D(pool_size=(3,3), strides=(2,2)),
    keras.layers.Flatten(),
    keras.layers.Dense(4096, activation='relu'),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(4096, activation='relu'),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(10, activation='softmax')
])
```

Щоб навчити мережу, компілюємо її.

Процеси компіляції включають зазначення наступних пунктів:

Функція збитків: метод, який кількісно визначає, наскільки добре працює модель машинного навчання. Кількісна оцінка - це вихід на основі набору вхідних даних, які називаються значеннями параметрів. Значення параметрів використовуються для оцінки прогнозу, а "втрата" - це різниця між прогнозами та фактичними значеннями.

Алгоритм оптимізації: Оптимізатор у нейронній мережі - це алгоритмічна реалізація, яка полегшує процес градієнтного спуску в нейронній мережі, мінімізуючи значення втрат, що надаються через функцію втрат. Щоб зменшити втрати, першорядним є значення ваг у мережі, які вибираються належним чином.

Швидкість навчання: невід’ємний компонент деталі реалізації нейронної мережі, оскільки це значення фактора, яке визначає рівень оновлень, що здійснюються до значень ваг мережі.

```
model.compile(loss='sparse_categorical_crossentropy',
optimizer=tf.optimizers.SGD(lr=0.001), metrics=['accuracy'])
model.summary()
```

Ми також можемо надати короткий опис мережі, щоб мати більше уявлення про склад шарів мережі, запустивши функцію model.summary ().

На даний момент ми готові навчати мережу.

Навчати мережу AlexNet за допомогою модуля Keras дуже просто. Потрібно лише викликати метод fit () і передати відповідні аргументи.

Epouch: Це числове значення, яке вказує на кількість разів, коли мережа потрапляла до всіх точок даних у наборі навчальних даних.

```
model.fit(train_ds,
        epochs=50,
        validation_data=validation_ds,
        validation_freq=1,
        callbacks=[tensorboard_cb])
```

Останнім кроком є оцінка навченої мережі шляхом оцінки мережі.

Етап оцінки забезпечить перевірку ефективності навченої моделі за даними, які не використовувались для тренування. На етапі оцінки моделі використовуємо пакет тестових даних, створений на попередніх етапах.

Оцінити модель дуже просто, просто викликаємо метод evaluate() і передаємо пакетні дані тесту.

model.evaluate (test_ds)

На виході маємо:

312/312 [==============================] - 8s 27ms/step - loss: 0.9814 - accuracy: 0.7439

[0.9813630809673132, 0.7438902]

3.2.2.3 Створення бази біометричних зразків

Якщо в системі реєструється новий користувач, спочатку необхідно додати цього користувача і його біометричні дані в базу. Дані, що нам знадобляться – ім’я користувача та його зображення. Окрім цього, за допомогою бібліотеки Crypto.PublicKey генеруємо для користувача закритий RSA ключ з компактного вектору біометричного зразку.

RSA є найбільш розповсюдженим і використовуваним алгоритмом відкритого ключа. Його безпека заснована на складності розкладання великих цілих чисел. Алгоритм витримував атаки протягом 30 років, і тому він вважається досить безпечним для нових конструкцій.

Алгоритм може використовуватися як для конфіденційності (шифрування), так і для автентифікації (цифровий підпис). Варто зазначити, що підписання та дешифрування значно повільніші, ніж перевірка та шифрування. Криптографічна сила в основному пов'язана з довжиною модуля n. Достатньою довжиною вважається 2048 біт.

```
from Crypto.PublicKey import RSA
def generateKey(compactVector):
    key = RSA.generate(2048)
    privatekey = key.exportKey(passphrase=compactvector, pkcs=8)
```

```
    publickey = key.publickey().exportKey()
    return privatekey, publickey
```

Зберігаємо дані про користувача в базу даних. Для цього використовуємо бібліотеку psycopg2.

Psycopg - найпопулярніший адаптер бази даних PostgreSQL для мови програмування Python. Основними його особливостями є повна реалізація специфікації Python DB API 2.0 та безпека потоків (декілька потоків можуть мати спільне з'єднання). Він був розроблений для багатопотокових додатків, які створюють і руйнують безліч курсорів і роблять велику кількість одночасних INSERT або UPDATE.

Для збереження зразку в базу спочатку відкриваємо підключення до бази, виконуємо операцію вставки, після чого не забуваємо закрити підключення для економії ресурсів:

```
import psycopg2
def SaveToDatabase(username, id_item, FileImage):
    con = psycopg2.connect(
      database="postgres",
      user="postgres",
      password="*********",
      host="127.0.0.1",
      port="5432"
    )
    cur = con.cursor()
    query = ""
    query  += "INSERT INTO biometric_data"
    query  += "("
    query  += "id_item"
    query  += ", blob_file"
    query  += ", username"
    query  += ") VALUES ("
```

```
query  += "(%id_item)"
query  += ", '(%FileImage)'"
query  += ", '(%username)'"
query  += ")"
cur.execute(query)
con.commit()
con.close()
```

3.2.2.4 Нейромережеве співставлення образів і ключів

Спочатку ми ініціалізуємо ваги та розміри шарів для нейронної мережі.

Визначаємо змінні таким чином:

- [num_samples, n_x]: Форма вхыдних даних
- n_x: Кількість вхідних зразків X
- n_h: Вузли в прихованому шарі
- n_y: Кількість цільових значень для кожної вибірки

Проініціалізуємо розміри шарів залежно від вхідних та вихідних значень та розміру прихованого шару.

```
def layer_sizes(x, n_h, y):
    n_x=x.shape[1]
    n_h=4
    n_y=y.shape[1]
    return n_x, n_h, n_y
```

Далі ініціалізуємо ваги залежно від розмірів шару.

```
import numpy as np
def initialize_weights(n_x, n_h, n_y):
    W1 = np.random.rand(n_x, n_h)
    W2 = np.random.rand(n_h, n_y)
    return W1, W2
```

Пряме розповсюдження складається з групи лінійних регресій, поєднаних з нелінійною функцією активації на кожному шарі.

У якості функції активації використовується сигмоїдальна функція.

```
def forward(x, W1, W2):
    Z1 = np.dot(x,W1) #first layer linear forward
    A1 = sigmoid(Z1) #first layer activation
    Z2 = np.dot(A1,W2) #output linear forward
    output = sigmoid(Z2) #activation output layer
    return output, A1
```

Зворотне розповсюдження відповідно оновлює параметри ваги у тому напрямку, де втрати мінімізовані.

```
def backward(x, y, A1, output, W1, W2):
    #calculate gradients
    d_w2 = np.dot(A1.T, (2*(y - output) * sigmoid_derivative(output)))
    d_w1 = np.dot(x.T,  (np.dot(2*(y - output) * sigmoid_derivative(output), W2.T) * sigmoid_derivative(A1)))
    #update weights
    W2+=d_w2
    W1+=d_w1
    #return updated weights
    return W1,W2
```

Навчання мережі. Одна епоха у навчанні складається з одного кроку прямого і зворотного розповсюдження. Використано набір даних з 30 функцій та 569 зразків, який навчає мережу протягом 1000 епох з 4 вузлами у прихованому шарі.

```
X, y = load_dataset(return_X_y=True)
#preprocessing
scaler=StandardScaler()
X=scaler.fit_transform(X)
y=y.reshape(X.shape[0],1)
n_x, n_h, n_y = layer_sizes(X, 4, y) #initiate layer sizes
W1, W2 = initialize_weights(n_x, n_h, n_y) #initialize weights
loss=[]
```

```
epochs=1000
m=x.shape[0]
for i in range(epochs):
    output, A1 = forward(X, W1, W2)
    iter_loss=(1/(2*m))*np.sum((y-output)**2)
    loss.append(iter_loss)
    W1, W2 = backward(X, y, A1, output, W1, W2)
```

3.2.2.5 Відновлення ключа

У випадку, якщо система знайшла користувача, який має відповідний вхідному біометричний зразок, ми повертаємо згенерований для цього користувача ключ. В іншому випадку – генеруємо випадковий ключ RSA і повертаємо його. З таким ключем в системі не вдасться виконати жодних дій.

```
from Crypto.PublicKey import RSA
def getUserKey(user):
    if (user)
        return user.getKey()
    else
        return RSA.generate(2048)
```

При цьому метод user.getKey() не бере ключ з бази, а заново його генерує, беручи за основу збережений в базі компактний вектор біометричного зразку, який належить користувачу.

3.3 Розробка інтерфейсної частини програми

Інтерфейсна частина програми містить три основні сторінки: main.html (початкова сторінка), signup.html (сторінка реєстрації), login.html (сторінка входу).

На основній сторінці розміщуємо два посилання: на сторінку реєстрації та на сторінку входу відповідно.

Тіло розмітки сторінки наступне:

```
<body ng-controller="mlController" ng-cloak>
```

```
    <div>
    <h1>Програмний модуль автентифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ</h1>
    </div>
    <div class="center">
      <a class='button' href='login.html'>Login</a>
      <br>
      <a class='sign-up' href='signup.html'>Sign up</a>
    </div>
    <div id="result" class="center"></div>
  </body>
```

Для даної сторінки підключаємо наступні стилі, що задають фон, та основні властивості тексту на сторінці, а також форму і розміщення посилань:

```
h1 {
  text-align: center;
  color: white;
}
body {
  font-family: Arial;
  background-color: #333333;
  margin: 0px;
}
div { width: 100%; margin: 5px; }
.center {
  margin: auto;
  text-align: center;
}
.button {
  margin-top: 100px;
  margin-bottom: 10px;
```

```
  background-color: white;
  border: none;
  color: #333333;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 32px;
  border-radius: 10px;
  font-weight: bold;
}
.sign-up {
  color: white;
  font-size: 25px;
}
```

Сторінка реєстрації містить в собі форму, куди користувач вводить username та кнопку continue, при натиску на яку з’являється модальне вікно, в якому можна зробити фото і підтвердити реєстрацію.

Тіло розмітки сторінки реєстрації наступне:

```
<body ng-controller="mlController" ng-cloak>
    <div>
        <h1>Програмний модуль автентифікації з використанням нейромережево
го перетворення біометричного зразку в
            криптографічний ключ</h1>
    </div>
    <div class="center">
        <form onsubmit="takeSnapshot()">
            <fieldset id="username-form">
                <input type="text" name="username" placeholder="Enter your name">
                <button onclick="gotoSnapshot()">Continue</button>
```

```
            </fieldset>
            <fieldset id="snapshot-form">
               <div id="webcam" class="center cam-
size" style="width:600px; height:400px"></div>
               <input type="submit" ng-click="takesnapshot()" class="button"
      value="Take Snapshot">
            </fieldset>
         </form>
      </div>
      <div id="result" class="center"></div>
   </body>
```

Стилі, застосовані до сторінки реєстрації включають ті ж сстилі, що і для головної сторінки та додатково стилі для поля вводу імені користувача та блоку забору біометричної інформації:

```
[ng\:cloak], [ng-cloak], [data-ng-cloak], [x-ng-cloak], .ng-cloak, .x-ng-cloak {
  display: none !important;
}
.message {
  color: blue;
  text-align: center;
  color: #cfd4d3;
}
.cam-size {
  width:600px;
  height:400px;
  position:absolute;
}
#snapshot-form {
  display: none;
}
```

```
input[type=text] {
  width: 30%;
  padding: 12px 20px;
  margin: 8px 0;
  box-sizing: border-box;
  margin-top: 100px;
}
.result {
  display: none;
}
#webcam { position: relative;}
```

Скрипт, який робить фото та надсилає запит з даними користувача до серверу, а також обролює відповідь:

```
var snapshot;
$scope.takesnapshot = function (formdata) {
    Webcam.snap(function (data_uri) {
        document.getElementById('result').innerHTML = '<img src="' + data_uri + '"/
>';
        snapshot = data_uri;
    });
    registerUser(formdata);
}
function registerUser(formdata) {
    const data = { image: snapshot, username: formdata.username };
    fetch('http://localhost:8080/registration', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify(data),
```

```
    })
        .then(response => response.json())
        .then(data => {
            if (data.status === 200) {
                alert(`${username}, welcome to system. Registration successful`);
            }
        })
        .catch((error) => {
            console.error('Error:', error);
        });
}
$scope.gotoSnapshot = function () {
    document.getElementById('username-form').style = 'display: none';
    document.getElementById('snapshot-form').style = 'display: block';
}
```

Даний скрипт містить 3 функції:

1) gotoSnapshot – функція, що спрацьовує при натиску на кнопку Continue та приховує блок з полем вводу і замість нього показує користувачу блок для зняття фото.
2) takesnapshot – функція, що робить знімок обличчя. Викликається після натиску на кнопку Take snapshot.
3) registerUser – функція, що надсилає дані, отримані від користувача, на сервер і оброблює відповідь. Якщо відповідь має статус 200 – виводить користувачу повідомлення про успішну реєстрацію, інакше – показує помилку.

Сторінка входу в систему містить блок, в якому можна зробити фото, як тільки користувач робить фото, надсилається запит на сервер.

Розмітка сторінки входу:

```
<body ng-controller="mlController" ng-cloak>
    <div>
```

```
<h1>Програмний модуль автентифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ</h1>
<p class='message'>Для входу в систему необхідно надати сайту дозвіл вико ристовувати камеру<p>
</div>
<div id="webcam" class="center cam-size" style="width:600px; height:400px">
</div>
<div class="center">
  <button ng-click="takesnapshot()">
    Take Snapshot
  </button>
</div>
<div id="result" class="center"></div>
</body>
```

Стилі, застосовані до сторінки входу ідентичні тим, що застосовані для сторінки реєстрації, тільки не містять налаштування поля вводу.

Скрипт, який робить фото та надсилає запит до серверу, а також обролює відповідь:

```
var snapshot;
$scope.takesnapshot = function (username) {
    Webcam.snap(function (data_uri) {
        document.getElementById('result').innerHTML = '<img src="' + data_uri + '"/
>';
        snapshot = data_uri;
    });
    loginUser();
}
function loginUser() {
    const data = { image: snapshot };
```

```
    fetch('http://localhost:8080/login', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify(data),
    })
        .then(response => response.json())
        .then(data => {
            if (data.status === 200) {
                alert(`${data.username}, welcome to system. Login successful`);
            }
        })
        .catch((error) => {
            alert("Access is denied!");
        });
}
```

Даний скрипт містить 3 функції:

1) takesnapshot – функція, що робить знімок обличчя. Викликається після натиску на кнопку Take snapshot.
2) loginUser – функція, що надсилає дані, отримані від користувача, на сервер і оброблює відповідь. Якщо відповідь має статус 200 – виводить користувачу повідомлення про успішний вхід, інакше – доступ заборонено.

3.4 Опис та тестування розробленого програмного модулю

При першому вході в систему користувач бачить сторінку, де йому пропонується авторизуватися (рис. 3.4).

Якщо користувач не має облікового запису на вебсторінці, він може створити такий запис натиснувши Sign up. В такому випадку він побачить форму реєстрації, зображену на рис. 3.5.

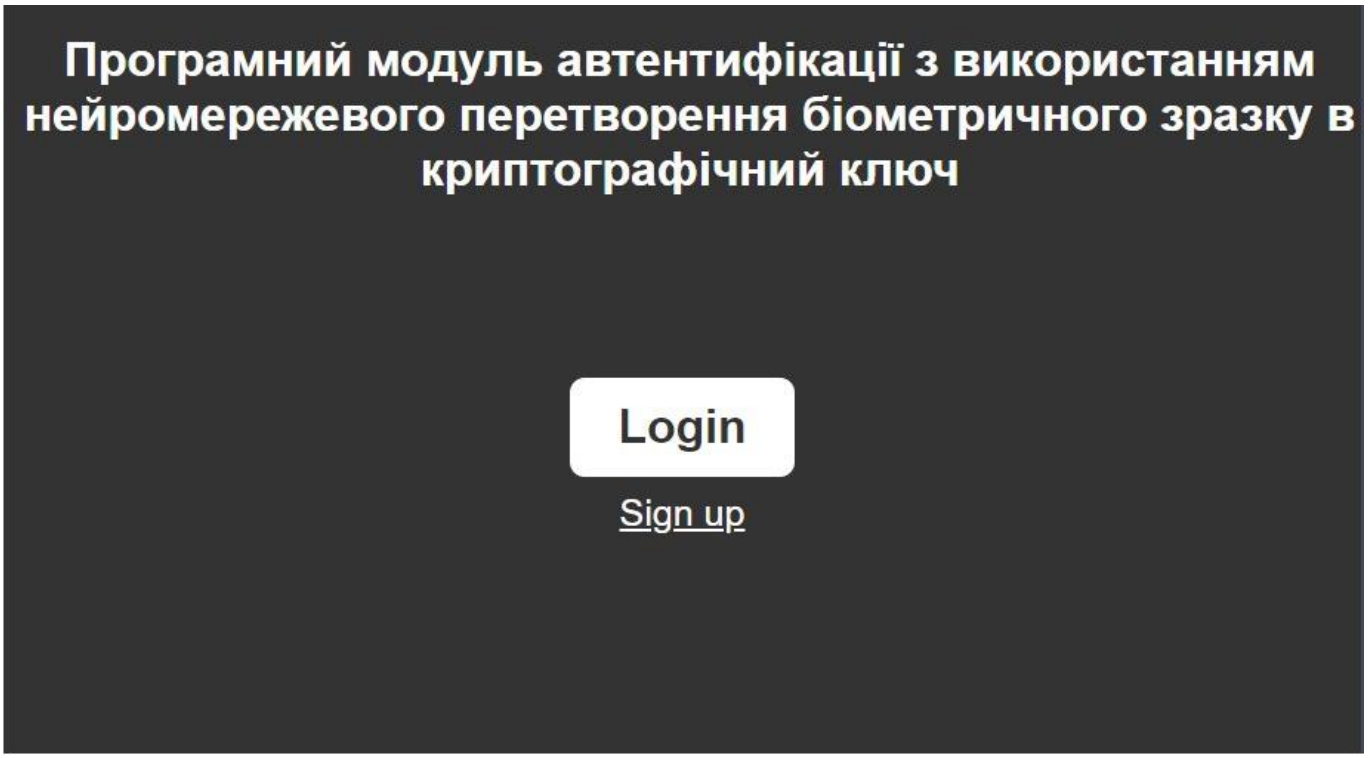

Рис. 3.4. Головна сторінка

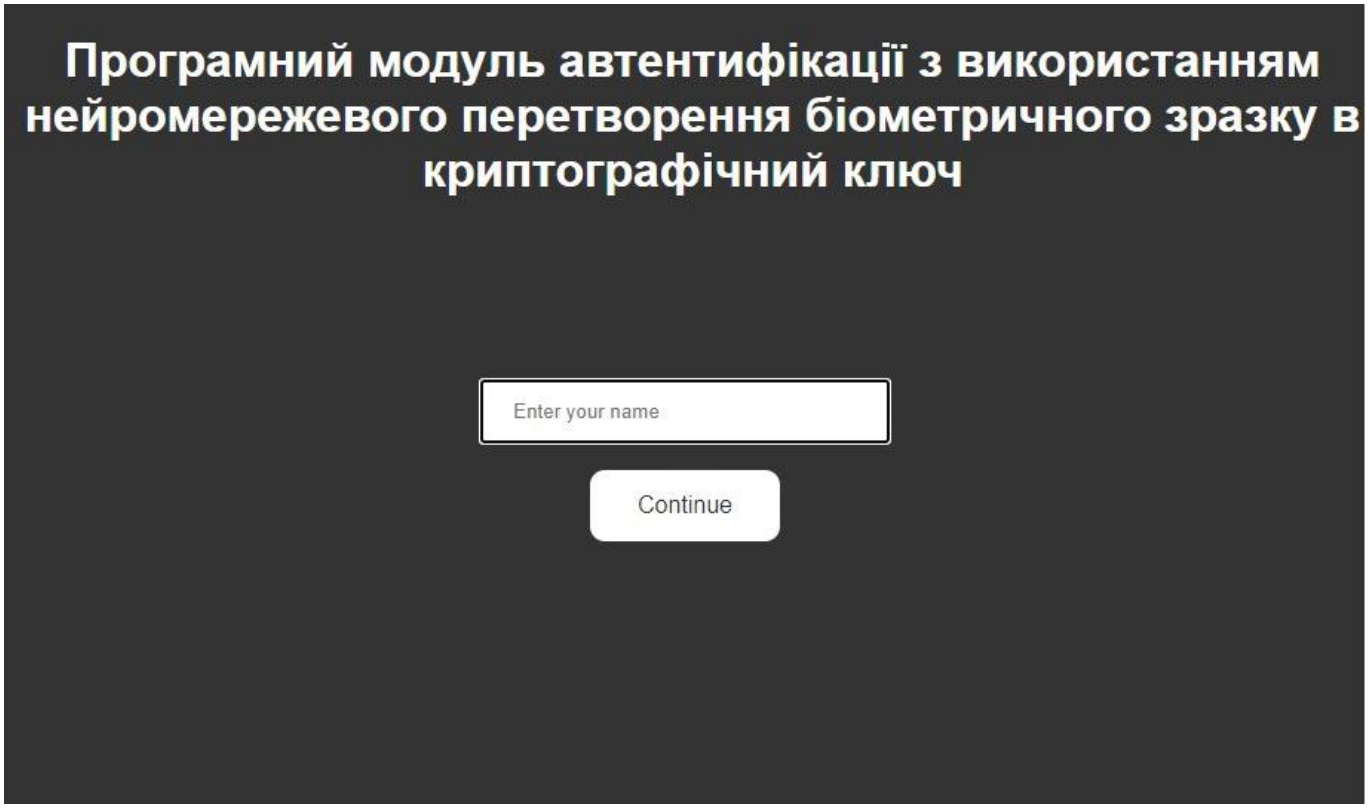

Рис 3.5. Сторінка реєстрації

Після заповнення даних, користувач може натиснути кнопку Continue і тоді перед ним з’явиться запит на доступ до камери (рис. 3.6). Необхідно натиснути «Разрешить», інакше у системи не буде можливості зробити забір біометричного зразку.

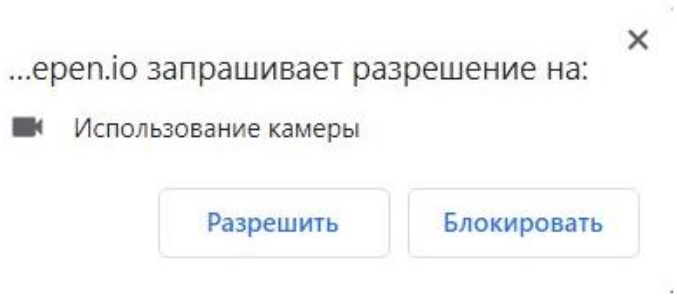

Рис. 3.6. Запит на доступ до камери

Після того, як користувач дозволить сайту використовувати камеру, він побачить вікно у якому можна зробити фото та надіслати його до серверу за допомогою кнопки Take snapshot (рис. 3.7).

Для успішного забору даних важливо, щоб у приміщенні було нормальне освітлення, обличчя було направлене на камеру прямо і повністю поміщалось у вікні для забору даних. Фото має бути портретного типу.

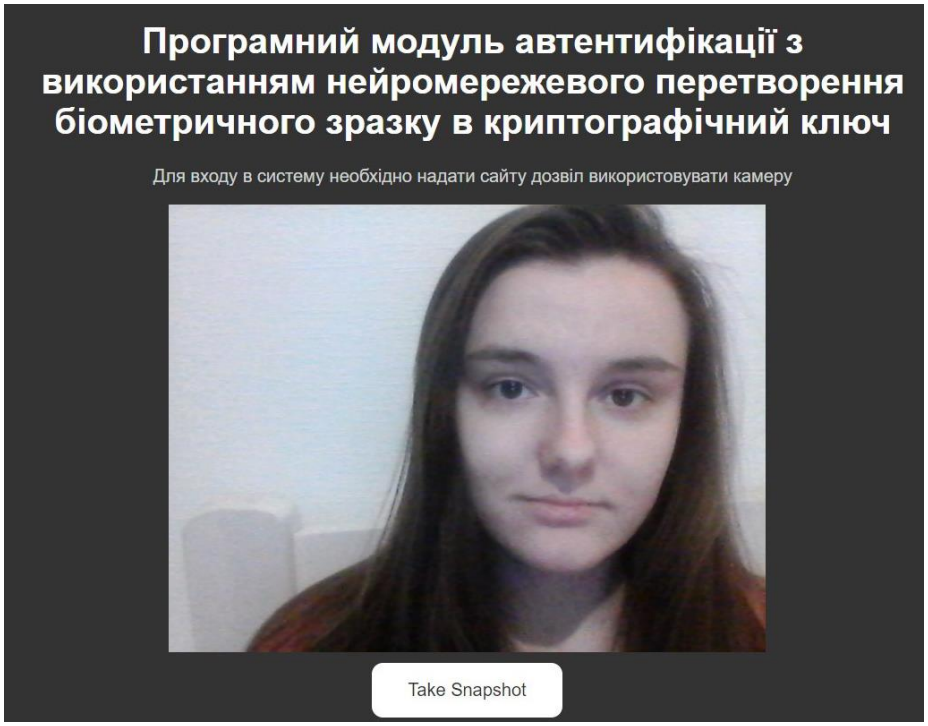

Рис. 3.7. Вікно для забору біометричної інформації

В разі успішної реєстрації користувач побачить відповідне повідомлення (рис. 3.8), якщо реєстрація не пройшла – повідомлення з помилкою (рис. 3.9).

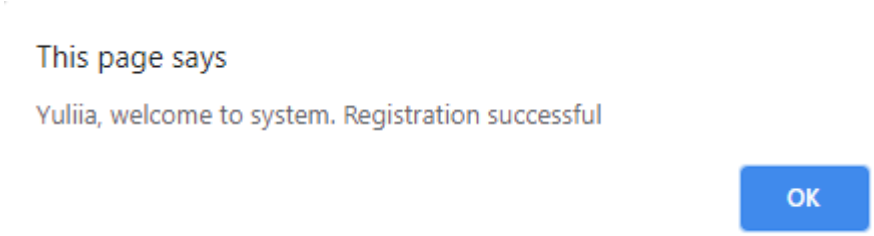

Рис. 3.8. Повідомлення про успішну реєстрацію

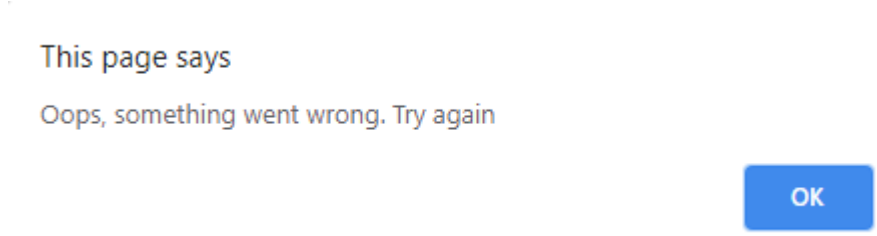

Рис. 3.9. Повідомлення про помилку при реєстрації

При спробі увійти в систему (кнопка Login на головній сторінці – рис 3.4) користувач побачить прохання надати сайту дозвіл використовувати камеру, якщо такий дозвіл не було надано раніше (рис. 3.6), після чого з’явиться вікно для забору біометричної інформації ідентичне тому, що використовувалось при реєстрації (рис. 3.7).

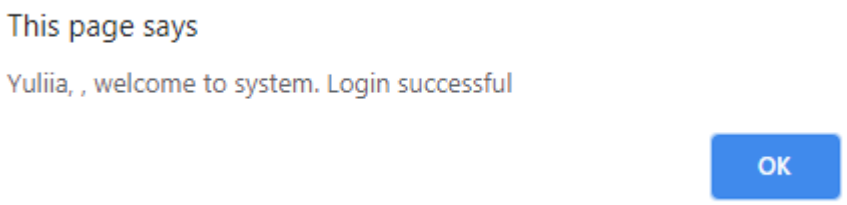

Рис. 3.10. Повідомлення про успішний вхід у систему

Як тільки користувач натисне кнопку Take Snapshot, його фото выдправиться на сервер для проведення автентифікації. В якотсті результату автнетифікації

користувач побачить або повідомлення про успішний вхід у систему (рис. 3.10), або відмову у доступі (рис. 3.11).

Рис. 3.11. Відмова у доступі

3.5 Оцінка ефективності впровадження програмного модулю

Для оцінки ефективності впровадження програмного модулю автентифікації було проведено експеримент, в якому порівнювались різні методи генерації біометричних образів та їх співставлення з криптографічним ключем.

На етапі попередньої обробки було здійснено створення двох навчальних вибірок: позитивної і негативної. У відео потоці зафіксовані різні кути нахилу голови в мінливих умовах освітлення, а також з однорідним фоном. Вихідні кадри мали розширення 1280×720 пікселів, потім на кожному з них проведений пошук області, що містить обличчя людини (ROI - region of interest), з допомогою алгоритму Віоли-Джонса [40].

Виділені області розмічені і масштабовані до розміру 256×256 пікселів таким чином, щоб містити мінімальну кількість пікселів за межами обличчя, що нас цікавить. Додатково проводилася нормалізація зображень вирівнюванням гістограми для нормалізації ділянок кадрів з різними яскравостями. Перша вибірка містить достатню кількість зображень осіб кожного користувача системи. Негативна вибірка утворена зображеннями довільних предметів, пейзажів і т.п. і служить для ізоляції позитивних прикладів в просторі ознак.

Кодування ознак пропонується виконувати за допомогою такої схеми: зображення всіх класів позитивної і негативної вибірок представляються у вигляді цілочисельної матриці уніфікованого розміру [n, n], що розбивається через

підрядник в вектор-стовпець розміром [n×n, 1], що забезпечує інваріант до зміщення області інтересу в вертикальному напрямку.

Кожен елемент вектор-стовпця може бути перетворений в рефлексивний двійковий 8-розрядний код Грея [41], в такому поданні два сусідніх значення колірної шкали відрізняються тільки в одному розряді [42].

Відкриті та закриті ключі створені в криптографічній системі з використанням асиметричного алгоритму RSA. Закритий ключ представляється у вигляді цілочисельного вектор-стовпця [m, 1] довжиною m = 132. Кожен елемент вектора перетворюється в рефлексивний двійковий 8-розрядний код Грея, а потім отримана матриця знову ділиться на рядки. З отриманих рядків будується вектор-стовпець [8×m, 1].

Генерація ознак полягає в проектуванні первинного вектора в новий простір ознак і формуванні компактного вектора ознак кожного образу для подальшої нейромережевої обробки. Цей вектор подається в нейромережевий блок у формі двійкового вектора або в формі цілочисельного вектора, який, в свою чергу, генерує закритий ключ, що подається на вхід модуля криптографічної системи.

Параметри проведення експериментів наведені в табл. 2.1, де

1) вихідне зображення - в градаціях сірого або в RGB у вигляді цілочисельної матриці;

2) satlins - шматково-лінійна функція активації, симетрична [43];

3) elliotsig - апроксимація сигмоїдальної функції активації виду гіперболічний тангенс [44];

4) hardlim - порогова функція активації, не симетрична.

Варто відзначити, що у всіх експериментах, крім експерименту з BAM, навчальні вибірки складалися з позитивної (5 класів по 1000 прикладів) і негативної (1000 прикладів). У випадку з BAM навчальна вибірка складалася з позитивної (5 класів по 100 прикладів) і негативної (100 прикладів).

Поділ всіх вибірок у відсотковому співвідношенні був наступним: навчальна вибірка становила 75%, а тестова - 25%.

В ході експериментів, результати яких відображені в табл. 2.2, було встановлено перевагу карт Кохонена в показниках помилки першого роду, але в свою чергу даний метод пов'язаний з проблемами довгого навчання та підвищеної складності додавання користувачів, чого позбавлений метод головних компонент.

Табл. 3.1.

Параметри проведення експериментів для різних типів нейронних мереж

Параметри	Експеримент				
	БШП	ЙГМК + БШП	SOM + БШП	AlexNet + БШП	AlexNet + BAM
Вихідне зображення	[64, 64, 1]	[64, 64, 1]	[64, 64, 1]	[227, 227, 3] у колірний схемі RGB	[64, 64, 1]
Генерація компактного вектора ознак	-	За допомогою ЙМГК відібрано 64 головних компоненти	SOM, 15x15 нейронів з гексагональною решіткою	Згорткова НМ AlexNET	-
Архітектура нейронної мережі (НМ) блоку співставлення					
Розмірність вхідного вектору	4096	64	225	1000	4096
Тип вхідного вектору	Десяткові цілі числа [0, 255]	Дійсні числа	Бінарний вектор активностей нейронів виходного шару карти Кохонена	Дійсні числа	Двійковий код Грея
Розмірність вихідного вектору	132 або 1056	1056	1056	1056	1056
Тип вихідного вектору	Десяткові цілі числа [0, 255] або двійкові в коді Грея	Двійковий код Грея	Двійковий код Грея	Двійковий код Грея	Двійковий код Грея
Тим НМ	БШП	БШП	БШП	БШП	BAM
Кількість нейронів у кожному шарі	4096, 2018, 1056	64, 1056	225, 1056	1000, 1056	4096, 1056
Функції активації нейронів у кожному шарі	Elliotsig, elliotsig, satlins	Elliotsig, satlins	Elliotsig, satlins	Elliotsig, satlins	Satlins, satlins
Постобробка виходу	- / hardlim	hardlim	hardlim	hardlim	hardlim

Використання гетероасоціативної пам'яті на основі нейронної мережі BAM показало хороші результати співставлення, однак ємність подібної мережі дуже мала.

Для співставлення компактного вектора ознак, виділених з біометричного образу, і закритого криптографічного ключа використані багатошарові перцептрони і гетероасоціативна пам'ять на основі нейронної мережі BAM.

Кількість змінюваних параметрів дуже велика і не дозволяє співставляти велику кількість вхідних образів високої розмірності (n) і вихідних ключів (p) достатньої довжини. Загальна кількість образів, що запам'ятовуються мережею, виражається співвідношенням $m=sqrt(min(n,p))$.

Згортокові мережі вимагають істотних обчислювальних ресурсів для виділення метаознак і дозволяють отримати кращі показники чутливості і специфічності.

Табл. 3.2.

Порівняльні результати експериментів

Параметри	Навчання					Тестування				
	БШП	ЙГМК + БШП	SOM + БШП	AlexNet + БШП	Alex Net + BAM	БШП	ЙГМК + БШП	SOM + БШП	AlexNet + БШП	AlexNet + BAM
Абсолютна кількість помилок/ зразків	394 [4500]	362 [4500]	281 [4500]	273 [4500]	230 [4500]	133 [1500]	124 [1500]	95 [1500]	88 [1500]	80 [1500]
Доля коректно розпізнаних образів, %	91,24	91,96	93,76	93,93	94,89	91,13	91,73	93,96	94,13	94,67
Чутливість	0,9688	0,9825	0,9784	0,9729	0,9865	0,9828	0,9806	0,9808	0,9808	1
Специфічність	0,9727	0,9753	0,9830	0,9811	0,9628	0,9763	0,9775	0,9774	0,9774	0,9758

Було встановлено, що досить одного прихованого шару для успішного відображення вхідного образу у вихідний ключ. Використання прикладів перевірочної вибірки, що не увійшли в навчальну та тестову (близько 30 тис.

зображень облич і негативних прикладів), показало можливість мережі генерувати ключ, що не належить жодному з легітимних користувачів.

При використанні тільки позитивної вибірки 42% прикладів класу «чужий» нейронна мережа відносить до одного з наявних класів - породжує ключ легітимного користувача.

Ймовірно така поведінка обумовлена особливістю розташування породжуваних персептронами поділяючих гіперплощин і побудовою незамкнутих областей розташування вихідних образів.

У результаті порівняльного аналізу алгоритмів вилучення первинних біометричних ознак і співставлення сформованого образу з закритим ключем в рамках запропонованої системи автентифікації встановлено, що найбільш ефективним на тестових даних є підхід з використанням глибоких згорткових мереж (AlexNet) і нейромережевої двонаправленої гетероасоціативної пам'яті (BAM).

3.6 Висновки до розділу 3

В результаті виконання дипломної роботи було розроблено алгоритм та реалізовано програмний модуль біометричної автентифікації з використанням нейромережевого перетворення біометричного зразку в криптографічний ключ. Даний програмний модуль реалізовано на основі клієнт-серверної архітектури. Для розпізнавання біометричного зразку, генерції образу та його співсталення щ криптографічним ключем було використоно нейронні мережі, а саме: згорткову мережу AlexNet та двонаправлену асоціативну пам’ять також відому як нейронна мрежа Коско.

Для реалізації вказаних вище нейронних мереж було використано бібліотеки TensorFlow (платформа з відкритим кодом для впровадження, навчання та розгортання моделей машинного навчання) та Keras (бібліотека з відкритим кодом, що використовується для реалізації архітектур нейронних мереж, що працюють як на центральних, так і на графічних процесорах).

Для генерації закритого криптографічного ключа обрано алгоритм RSA.

Для програмної реалізації було використано наступний стек інструментів: Python – на серверній частині; PostrgeSQL – база даних; psycopg2 для запитів до бази даних; Angular 7, HTML 5, SCSS – на клієнтській частині.

ВИСНОВОК

В ході виконання дипломної роботи було проведено аналіз існуючих біометричних криптографічних систем. Було розглянуто ряд методів генерації біометричних ознак та їх співставлення з криптографічним ключем.

Для генерації ознак проналізовано наступні підходи:

- двовимірна карта самоорганізації Кохонена (self-organizing map, SOM);
- ймовірнісний метод головних компонент (ЙМГК, probabilistic principal component analysis, PPCA);
- згорткова нейронна мережа (convolutional neural network, CNN) AlexNet.

Для співставлення компактних векторів біометричних образів і криптографічних ключів проаналізовано наступні методи:

- побудова і навчання двонаправленої асоціативної пам’яті на основі нейронної мережі Б. Коско (bidirectional associative memory, BAM);
- побудова і навчання одношарових і багатошарових нейронних мереж прямого розповсюдження на основі перцептронів (БШП).

У результаті порівняльного аналізу алгоритмів вилучення первинних біометричних ознак і співставлення сформованого образу з закритим ключем в рамках запропонованої системи автентифікації встановлено, що найбільш ефективним на тестових даних є підхід з використанням глибоких згорткових мереж (AlexNet) і нейромережевої двонаправленої гетероасоціативної пам'яті (BAM).

В дипломній роботі запропоновано підхід, основою якого є інтеграція біометричної системи і криптографічного модуля, що дозволяє використовувати в якості виходу нейронної мережі згенерований на основі біометричного зразку секретний криптографічний ключ. Цей підхід вирішує проблему надійного зберігання секретного ключа.

Програмний модуль автентифікації реалізовано на основі клієнт-серверної архітектури. Серверна частина реалізована за допомогою різних інутрішніх

бібліотек Python (TensorFlow та Keras), які значно спрощують побудову та навчання нейронних мереж.

Інтерфейс програми реалізовано у вигляді вебсторінки з можливістю зареєструватися та увійти в систему.

Подібну систему автентифікації варто застосовувати в системах, де ймовірно зберігаються персональні дані, користувача, його цінні інформаційні ресурси або де користувач може виконувати певні важливі операції, для яких потрібно використовувати криптографічний ключ.

Можливості для покращення:

- нейронні мережі варто навчити відрізняти реального користувача від сфабрикованого фото, що забезпечить захист від імітації легітимного користувача;
- можна додати можливість користувачу зберігати в якості біометричного зразку не лише зображення свого обличчя, але й певний жест або вираз обличчя, таким чином підвищивши надійність системи.

Даний програмний модуль в подальшому може бути інтегрованим в вебсайти, мобільні та десктопні додатки, а також будь-які інші системи, що потребують підтвердження особистості користувача і мають доступ до вебкамери.

3.7

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. R.P Ramkumar, Dr. S Arumugam “A Novel Iris Recognition Algorithm” 26th-28th July 2012 Coimbatore, India.
2. Васильев В.И. Интеллектуальные системы защиты информации: учеб. пособие / В.И. Васильев. – 2-е. изд. – М.: Машиностроение, 2012. – 199 с.
3. Куликова О.В. Биометрические криптографические системы и их применение // Безопасность информационных технологий. – 2009. – Т. 16, №. 3. – С. 53–58.
4. Иванов А., Малыгин А. Высоконадежная биометрическая аутентификация пользователя: последний дюйм первой мили // Первая миля. – 2007. – Т. 2, № 2. – С. 20–24.
5. Ахметов Б.С. Технология использования больших нейронных сетей для преобразования нечетких биометрических данных в код ключа доступа / Б.С. Ахметов, А.И. Иванов, В.А. Фунтиков и др. // Алматы: ТОО «Изд-во LEM», 2014. – 144 с.
6. Dodis Y., Ostrovsky R., Reyzin L. et al. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data // Siam Journal on Computing – 2006. – Vol. 38, № 1. – P. 97–139.
7. Riaz N., Riaz A., Khan S.A. Biometric template security: an overview // Sensor Review. – 2018. – Vol. 38, № 1. – P. 120–127.
8. A. K. Jain, K. Nandakumar, A. Nagar, “Biometric template security”, EURASIP J, Adv Signal Process 2008, pp. 1-17.
9. A. K. Jain, A. Ross, U. Uludag, “Biometric template security” Challenges and solutions”, In Proc. of European Signal Processing Conf (EUSIPCO) 2005.
10. K. Nandakumar, A. Jain, and S. Pankanti, ”Fingerprint-based fuzzy vault: Implementation and performance”IEEE Trans.Inf. Forensics Security, vol. 2, no. 4, pp.744-757, 2007.

11. A. Juels and M. Wattenberg, ”A fuzzy commitment scheme”, inProc. 6th ACMConf. Computer and Communications Security,G. Tsudik, Ed., pp. 28-36, 1999.
12. Feng Hao, Ross Anderson, and John Daugman, ”Combining Crypto with Biometrics Effectively”, IEEE Transactions on Computers, vol. 55, no. 9, pp. 1081-1088, 2006.
13. Yang, Shenglin, and Ingrid Verbauwhede. ”Automatic secure fingerprint verification system based on fuzzy vault scheme.” Proceedings (ICASSP’05). IEEE International Conference on Acoustics, Speech, and Signal Processing,Vol. 5. IEEE,2005.
14. K. Nandakumar, A. Jain, and S. Pankanti, ”Fingerprint-based fuzzy vault: Implementation and performance”IEEE Trans.Inf. Forensics Security, vol. 2, no. 4, pp.744-757, 2007.
15. Juels and M. Sudan, A fuzzy vault scheme, In International Symposium on Information Theory (ISIT), IEEE Pressm, (2002), p. 408.
16. A. Juels and M. Wattenberg: A fuzzy commitment scheme, in G. Tsudik, editor, Sixth ACM Conference on Computer and Communications Security, pages 28– 36. ACM Press, New York, 1999.
17. U. Uludag, S. Pankant, S. Prabhakar, A. K. Jain, “Biometric cryptosystems: issues and challenges”, In Proc. IEEE 2004, 92 (6), pp. 948-960.
18. C. Soutar, D. Roberge, A. Stoianov, R. Gilroy, B. V. Kumar, “Biometric encryption–enrollment and verification procedures”, In Proc. SPIE, Optical Pattern Recognition IX 1998, 3386, pp. 24-35.
19. C. Soutar, D. Roberge, A. Stoianov, R. Gilroy R, B. V. Kumar, “Biometric encryption using image processing”, In Proc. SPIE, Optical Security and Counterfeit Deterrence Techniques II 1998, 3314, pp. 178-188.
20. C. Soutar, D. Roberge, A. Stoianov, R. Gilroy R, B. V. Kumar, “Biometric encryption”, ICSA Guide to Cryptography, 1999.
21. C. Soutar, G. J. Tomko, G. J. Schmidt, “Fingerprint controlled public key cryptographic system”, US Patent 1996, 5541994.

22. C. Soutar, D. Roberge, A. Stoianov, R. Gilroy R, B. V. Kumar, “Method for secure key management using a biometrics”, US Patent 2001, 6219794.
23. A. Bodo A, “Method for producing a digital signature with aid of a biometric feature”, German patent DE 4243908 A1 1994.
24. G. Davida, Y. Frankel, B. Matt, “On enabling secure applications through offline biometric identification”, In Proc. of IEEE, Symp on Security and Privacy 1998, pp. 148-157.
25. G. Davida, Y. Frankel, B. Matt, “On the relation of error correction and cryptography to an off line biometric based identication scheme” In Proc. of WCC99, Workshop on Coding and Cryptography 1999, pp. 129-138.
26. Осовский С. Нейронные сети для обработки информации / пер. с пол. И.Д. Рудинского. – М. : Финансы и статистика, 2004. – 344 с.
27. Байесовский метод главных компонент [Електронний ресурс]. – Режим доступу: http://www.machinelearning.ru/wiki/images/7/73/BMMO11_11.pdf, вільний доступ (дата звернення: 12.11.2020).
28. Сверточная нейронная сеть. – Ч. 1: Структура, топология, функции активации и обучающее множество [Електронний ресурс]. – Режим доступу: https://habrahabr.ru/post/348000/, вільний доступ (дата звернення: 12.11.2020).
29. Самоорганизующаяся карта Кохонена (алгоритм кластеризации) [Електронний ресурс]. – Режим доступу: https://algowiki-project.org/ru/Самоорганизующаяся_карта_Кохонена_(алгоритм_кластеризации), вільний доступ (дата звернення: 12.11.2020).
30. J.J. Sylvester. On the reduction of a bilinear quantic of the nth order to the form of a sum of n products by a double orthogonal substitution // Messenger of Mathematics, 19, 1889, pp. 42–46.
31. K. Pearson. On lines and planes of closest fit to systems of points in space // Philosophical Magazine 2, 1901, pp. 559–572.
32. Глубокие нейросети (часть III). Выбор примеров и уменьшение размерности. [Електронний ресурс]. – Режим доступу:

https://www.mql5.com/ru/articles/3526#ppca, вільний доступ (дата звернення: 12.11.2020).

33. Свёрточная нейронная сеть [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Свёрточная_нейронная_сеть, вільний доступ (дата звернення: 15.11.2020).
34. AlexNet — свёрточная нейронная сеть для классификации изображений [Електронний ресурс]. – Режим доступу: https://neurohive.io/ru/vidy-nejrosetej/alexnet-svjortochnaja-nejronnaja-set-dlja-raspoznavanija-izobrazhenij/, вільний доступ (дата звернення: 15.11.2020).
35. Kosko B. Bidirectional associative memories. IEEE Transactions on Systems, man, and Cybernetics, 1988, vol. 18, no. 1, pp. 49–60.
36. Ассоциативная память на нейронных сетях [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Ассоциативная_память_на_нейронных_сетях, вільний доступ (дата звернення: 16.11.2020).
37. Нейронная сеть Коско [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Нейронная_сеть_Коско, вільний доступ (дата звернення: 16.11.2020).
38. Feedforward neural network [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Feedforward_neural_network, вільний доступ (дата звернення: 17.11.2020).
39. Roman M. Balabin; Ravilya Z. Safieva; Ekaterina I. Lomakina (2007). "Comparison of linear and nonlinear calibration models based on near infrared (NIR) spectroscopy data for gasoline properties prediction". Chemometr Intell Lab. 88 (2): 183–188.
40. Чуйков А.В., Вульфин А.М. Система распознавания жестов на основе нейросетевых технологий // Вестник УГАТУ. – 2017. – Т. 21, № 3. – С. 113–122.
41. Код Грея [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Код_Грея, вільний доступ (дата звернення: 19.11.2020)
42. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.: Техносфера, 2005. – Т. 1072. – С. 2.

43. Функции активации в нейронных сетях [Електронний ресурс]. – Режим доступу: http://www.aiportal.ru/ articles/neural-networks/activation-function.html, вільний доступ (дата звернення: 19.11.2020)
44. Elliot symmetric sigmoid transfer function [Електронний ресурс]. – Режим доступу: https://www.mathworks.com/help/nnet/ref/elliotsig.html, вільний доступ (дата звернення: 19.11.2020)