Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Центр післядипломної освіти

(повна назва)

Кафедра Програмної інженерії

(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів класифікації даних для програмної реалізації системи шифрування інформації

(тема)

Виконав:

Студент 6 курсу, групи ІПЗздм-19-1

Пікуль Марина Володимирівна

(прізвище, ініціали)

Спеціальність 121 Інженерія програмного забезпечення

(код і повна назва спеціальності)

Тип програми Освітньо-наукова

(освітньо-професійна або освітньо-наукова)

Керівник доц. Назаров О. С.

(посада, прізвище)

Допускається до захисту

Зав. кафедри _____ З.В. Дудар

(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Центр післядипломної освіти
(повна назва)

Кафедра Програмної Інженерії
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 121 Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми Освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія програмного забезпечення
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«_26__»___березня___ 20 21___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЕКТ)

студентові Пікуль Марині Володимирівні
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Дослідження методів класифікації даних для програмної реалізації системи шифрування інформації
затверджена наказом по університету від «_26__»_березня_2021 р. №__34Стз_____

2. Термін здачі студентом закінченої роботи «_17_» __травня_ 2021 р.

3. Вихідні дані до роботи (проекту) *Визначити найбільш ефективний метод класифікації даних для додатку для підтримки класифікації тез по напрямах. Використовувати ОС Windows, клієнт-серверну СУБД Sqlite, середу розробки PyCharm Professional 2019, мову програмування Python, бібліотеку Keras та TensorFlow*

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити) *мета роботи, аналіз предметної області та концептуальне моделювання, опис прийнятих проектних рішень(об'єктні моделі та методи, структура бази даних), опис програмної реалізації, тестування. Додатки: а) слайди презентації, б) приклади кодів програми*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)
Слайди презентації: актуальність роботи, мета роботи, аналіз аналогів,задачі, функціональні вимоги, скріншоти системи,тестування системи, висновки

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	доц. Назаров О. С.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Об'єктний аналіз поставленої задачи	06.04.2021	виконано
2	Розробка моделі взаємодії даних	06.04.21-08.04.21	виконано
3	Проектування структури програми та даних	08.04.21-09.04.21	виконано
4	Розробка коду програми	09.04.21-23.04.21	виконано
5	Тестування і налагодження програми	23.04.21-25.04.21	виконано
6	Написання пояснювальної записки	25.04.21-02.05.21	виконано
7	Перевірка на унікальність тексту	11.05.2021	виконано
8	Нормоконтроль	14.05.2021	виконано
9	Архівування диплома в електронний архів	17.05.2021	
10	Попередній захист	19.05.2021	
11	Допуск до захисту у зав. кафедри	19.05.2021	

Дата видачі завдання «_26_» __березня__2021 р.

Студент _____ Пікуль М.В.
(підпис)

Керівник роботи (проекту) _____ доц., ктн., Назаров О. С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 100 стор., 14 рис., 2 табл., 14 джерел, 6 додатків

КВАЛІФІКАЦІЙНА РОБОТА, БАЗА ДАНИХ, МЕТОДИ, ДАНІ, ІНФОРМАЦІЯ, КЛАСИФІКАЦІЯ, СПЕЦІАЛЬНІСТЬ, ТЕЗИ, ШИФРУВАННЯ.

Об'єкт дослідження – методи розпізнавання тексту.

Предметом дослідження є ефективність методів розпізнавання тексту на основі унікальних слів чи символів.

Мета дослідження – виявити найбільш ефективний метод класифікації даних для системи шифрування тез конференцій з попереднім розпізнаванням тексту.

Метод рішення - мова Python, середа розробки PyCharm Community Edition 2019, бібліотека Keras та TensorFlow.

В результаті дослідження буде створена система шифрування тез конференцій на основі попереднього розпізнавання напряму теми тексту.

ARTICLE, CERTIFICATION WORK, CLASSIFICATION, DATA, DATABASE, ENCRYPTION, INFORMATION, METHODS, SPECIALTY.

The object of study - methods of text recognition.

The subject of research is the effectiveness of text recognition methods based on unique words or symbols.

The purpose of the study is to find the most effective method of data classification for a system for encrypting conference abstracts with prior text recognition.

Solution method - Python language, PyCharm Community Edition 2019 development environment, TensorFlow and Keras libraries.

As a result of the research, a system of encryption of conference abstracts will be created on the basis of preliminary recognition of the direction of the text topic.

Я, Пікуль Марина Володимирівна, студент групи ІПЗздм-19-1, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження методів класифікації даних для програмної реалізації системи шифрування інформації», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.
Я ознайомлена з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

У даний час криптографічні методи використовуються не тільки для захисту інформації від ймовірного недозволеного доступу, а й лежать в основі багатьох нових інформаційних технологій електронного документообігу, віртуальних фінансів, таємного електронного голосування та ін [1, с. 232]. Сучасна криптографія може вирішити три великі проблеми: захист інформації (забезпечення конфіденційності); аутентифікації (забезпечення розпізнавання інформації і джерела повідомлень); забезпечення анонімності.

Перша проблема знана вже багато років, інші дві є відносно недавніми, з їх рішенням пов'язаний ряд багатообіцяючих напрямків теоретичної та прикладної криптографії. Втім, звичайна криптографічний завдання реалізації секретності інформації не втратила своєї актуальності і в даний час. Це пов'язано, в основному, з тим, що у часи масового використання комп'ютерних технологій основна мета захисту електронної інформації стала гострою проблемою [2, с. 114]. При цьому до механізмів шифрування пред'являються суворі умови (швидкість модифікації інформації, витривалість до аналітичних прийомів аналізу і т.д.), як це передбачено їх роботою в різних електронних механізмах. Для технологічного вживання криптографічних методів властиво зростання норм до шифрів одночасно по стійкості, швидкості та легкості реалізації.

Але сучасна тенденція така, що кожен день проводиться велика кількість шифрувань. Основний зміст процесу шифрування полягає у тому, щоб привести інформацію у неможливий до читання (зашифрований) вид. Тобто аби захистити інформацію від несанкціонованого доступу на даний момент існує багато варіантів та методів.

Криптографія дозволяє проводити трансфер даних в прихованій формі, забезпечуючи захищеність, стійкість і цілісність даних. При захисті засекреченої інформації криптографія сприяє великій мірі безпеки анкетних даних окремих персон і груп. Можливість доступу до даних, що зберігаються в комп'ютерних базах даних, значно зросла. Компанії зберігають загальні та особисті дані на комп'ютерах частіше, ніж будь-коли раніше. Велика частка приватних даних, що зберігаються, строго засекречена і не призначена для громадського перегляду. Тому такі дані має бути можливість шифрувати аби можна було їх передавати по будь-яких лініях зв’язку.

Але наразі маємо таку ситуацію, що рівень захищеності даних у нашій країні нещадно малий і є велика вірогідність втрати важливих даних.

Важливими даними вважаються також тексти тез конференцій, в тому числі міжнародних, які являють собою багату наукову спадщину. Конференції влаштовуються з метою обміну досвідом між науковцями та обговорення цікавих ідей та відкриттів. А отже тези – це інтелектуальна власність всіх учасників конференціїї і тому вона не має бути у без дозволу у відкритому доступі.

Також необхідно підкреслити, що на кожну конференцію надходить велика кількість тез, написаних на різні теми, що змушує організаторів конференції дуже довго передивлятися кожну роботу аби розподілити всі тези по темах.

Це приводить нас до того, що можливість розподіляти тези за темами можна автоматизувати, наприклад використовуючи методи глибинного машинного навчання, за допомогою яких можна буде виділяти окремі набори тез, а також заносити їх у певну базу, до якої можна буде отримувати доступ, але, можливо, тільки після дозволу авторів тез.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз проблеми шифрування

Як було зазначено вище, існує проблема передачі зашифрованої інформації відкритими каналами та її зберігання. Тому важливо розуміти, як можна провести цю дію та що для цього потрібно.

1.1.1 Визначення поняття інформації та інформаційної безпеки

Інформація завжди грала важливу роль в житті людини. Загальновідомим є наступне висловлювання – «Хто володіє інформацією, той володіє світом».

Дані – це сукупність зафіксованих відомостей, в формі, придатній для постійного зберігання, передачі і обробки.

За допомогою перетворення та обробки отримують інформацію.

Інформація – це сукупність будь-яких відомостей або даних, які передають в усній, письмовій чи іншій формі [1].

На даний момент не існує єдиного наукового визначення цього поняття. Це пов’язано з тим, що у кожної області знань є свої специфічні набори описів даного терміну.

В інформатиці під інформацією розуміють деяку послідовність символів, яка має смислове навантаження та представлена у вигляді, зрозумілому комп’ютеру.

Сучасні методи зберігання, передачі та обробки інформації допомагають розвитку злочинів, пов’язаних з інформацією. Тому розвиток інформаційної безпеки є одним з головних напрямів розвитку ІТ.

Інформаційна безпека – це рівень захисту інформації від незаконного перегляду, знищення або змінення, і, більш того, захищеність інформаційних ресурсів від дій, що виконуються для порушення їх працездатності [1].

Частинами інформаційної безпеки є:

- об’єкти безпеки;
- загрози об’єктам безпеки;
- політика забезпечення безпеки;
- система забезпечення безпеки.

Захист інформації – це процес, який спрямований на запобігання ненавмисних і несанкціонованих дій над захищеною інформацією [2].

1.1.2 Визначення складових інформаційної безпеки

Через стрімку інформатизацію суспільства зростає і цінність певної інформації, що, в свою чергу призводить, до збільшення збитків у випадку витоку, змінення чи видалення інформації.

Отже, вважаючи вищесказане, актуальною на даний момент є задача інформаційної безпеки.

Інформаційна безпека має три ключові властивості: конфіденційність, цілісність та доступність інформації, схематично це зображено на рисунку 1.1.

Досягнення інформаційної безпеки можливо, лише за умови наявності усіх цих властивостей. Якщо відсутня хоча б одна з них, то про безпеку не може бути мови.

Отже, кожна інформація тією чи іншою мірою має в собі всі ці властивості.

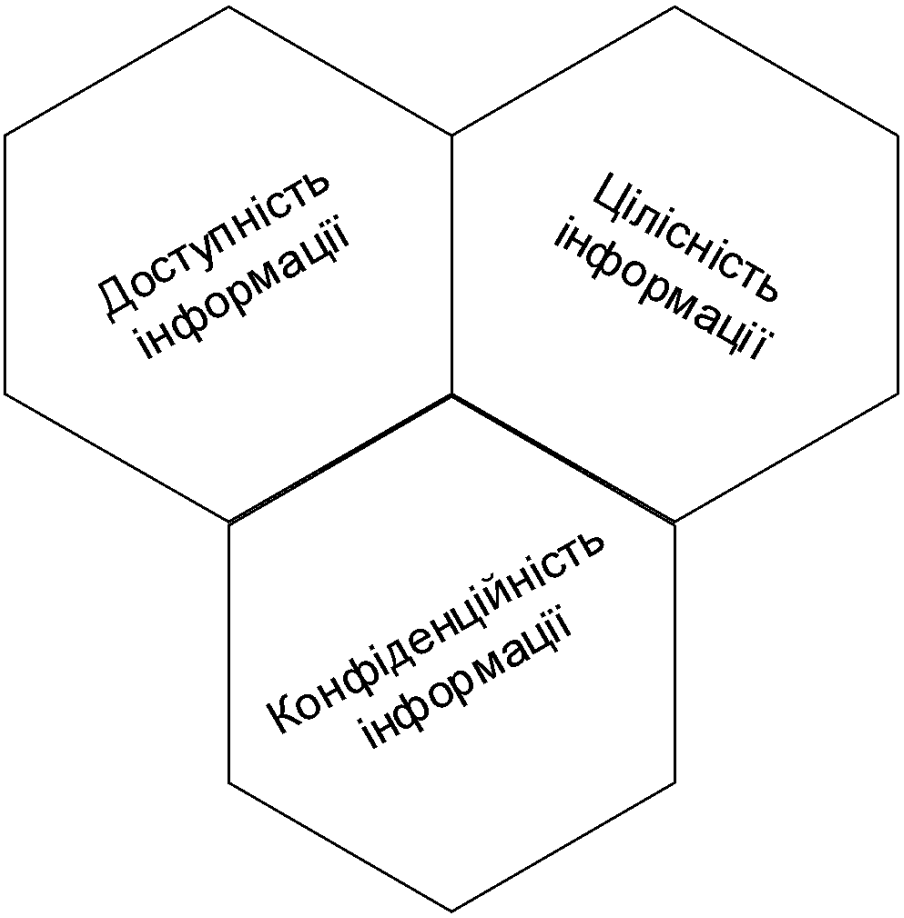

Рисунок 1.1 – Властивості інформаційної безпеки

Конфіденційність інформації – це властивість, що вказує на необхідність обмеження доступу до даної інформації певному колу осіб. Ця властивість інформації стає очевидною через об’єктивну необхідність захисту інтересів окремих суб’єктів інформаційних відношень. Загроза порушення конфіденційності інформації може призводити до ситуації у якій дана інформація стає відомою тому, хто не має повноважень доступу до неї.

Цілісність інформації – це стан інформації, в якому вона існує в незміненому вигляді відносно деякого фіксованого стану. Загроза порушення цілісності складається з будь-яких навмисних змін інформації, що існують в комп’ютерній системі, або проходять трансфер з однієї системи в іншу. Цілісність також може бути порушена, якщо через несанкціоновані зміни трапляються випадкові збої у роботі

технічних засобів, помилки при розробці програмних засобів, ненавмисних помилок персоналу та користувачів у процесі експлуатації комп'ютерної системи. На відміну від несанкціонованих загроз, порушення цілісності можливі санкціоновані зміни інформації, які плануються і виконуються періодично з метою необхідної корекції певних баз даних.

Доступність інформації – це властивість, що характеризує здатність забезпечення своєчасного та безперешкодного доступу користувачів до необхідної інформації. Загроза доступу до інформації виникає кожного разу, коли в результаті зловмисних дій інших користувачів або зловмисників закривається доступ до певного ресурсу комп'ютерної системи. Таке блокування може бути постійним, тобто ресурс, на який робиться запит, ніколи не буде отриманий. Блокування може викликати достатньо тривалу затримку ресурсу, на який робиться запит, щоб він перестав бути корисним. У таких випадках кажуть, що ресурс вичерпано [3].

1.1.3 Класифікація методів та засобів захисту інформації

Для того, щоб забезпечити захист інформації, часто використовуються наступні засоби (методи):

– фізична перепона, тобто перешкода на шляху зловмисника до інформації, що захищається;

– приховування за допомогою перетворення даних у форму, що не підлягає сторонньому сприйняттю;

– захист від атак шкідливих програм за допомогою використання антивірусних програм;

– регламентація, тобто такі умови, при яких можливе найкраще виконання норм та стандартів;

- примус до дотримання певних правил поводження щодо інформації;
- спонукання до непорушення встановлених порядків.

Основні засоби захисту інформації поділяються на фізико-технічні, або формальні та організаційно-соціальні, або неформальні. Вони включають в себе наступні засоби [4]:

- фізичні – автономно працюючі механічні, електричні, або електромеханічні пристрої та системи, за допомогою яких створюють перешкоди на шляху загроз;
- апаратні – електроні і електромеханічні пристрої, що вбудовуються в схему апаратури системи роботи над даними чи пов'язані з нею рішення, що створені для захисту інформації;
- програмні – спеціальні набори програм або окремі програмні компоненти, що вбудовані в ПО для захисту його інформації;
- організаційні – організаційно-технічні заходи для приховування інформації;
- законодавчі – нормативно-правові акти, за допомогою яких встановлюють відповідальність, права та обов’язки осіб, що мають доступ до захищеної інформації;
- морально-етичні – правила та норми, виконання яких допомагає захисту інформації.

На рисунку 1.2 схематично зображено описані вище методи та засоби.

1.1.4 Класифікація криптоалгоритмів захисту інформації

Більшість засобів захисту інформації використовують криптографію, що належить до програмних засобів (див. рисунок 1.2).

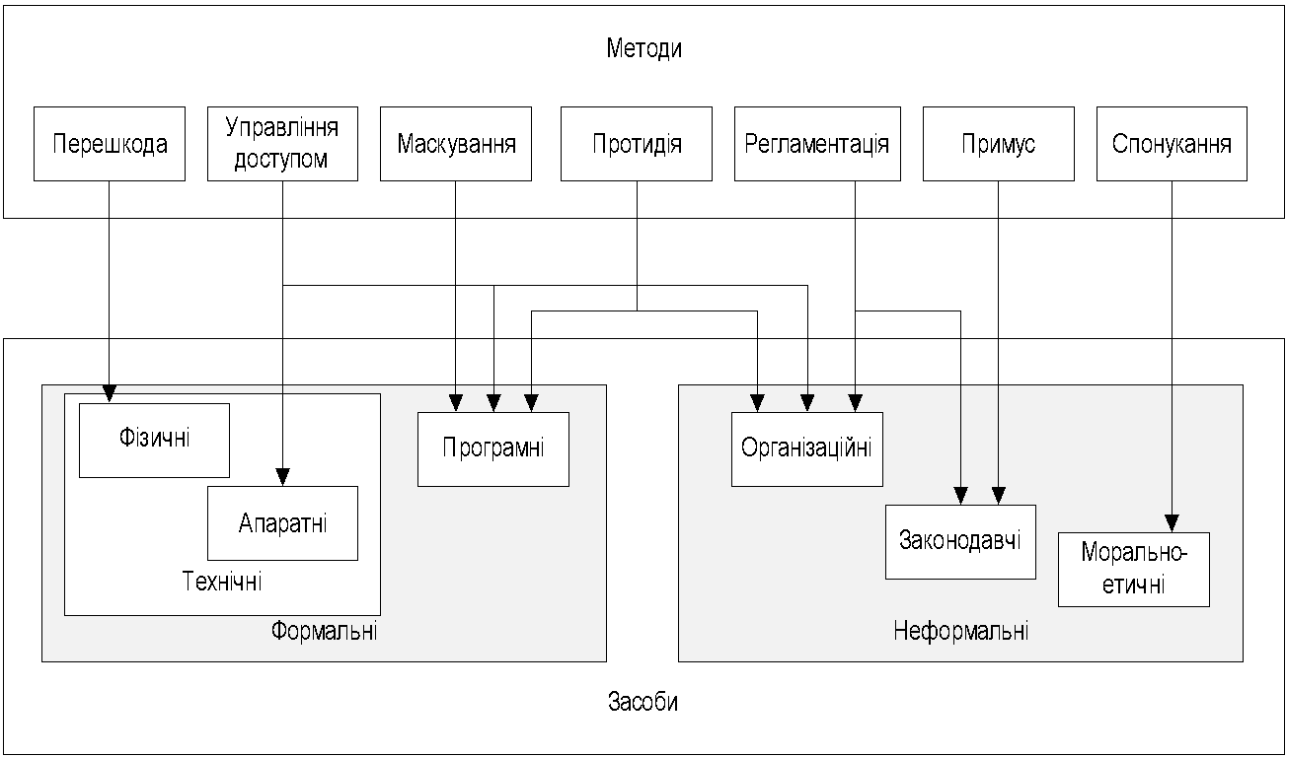

Рисунок 1.2 – Класифікація методів за засобів захисту інформації

Існує декілька варіантів класифікації криптографічних алгоритмів.

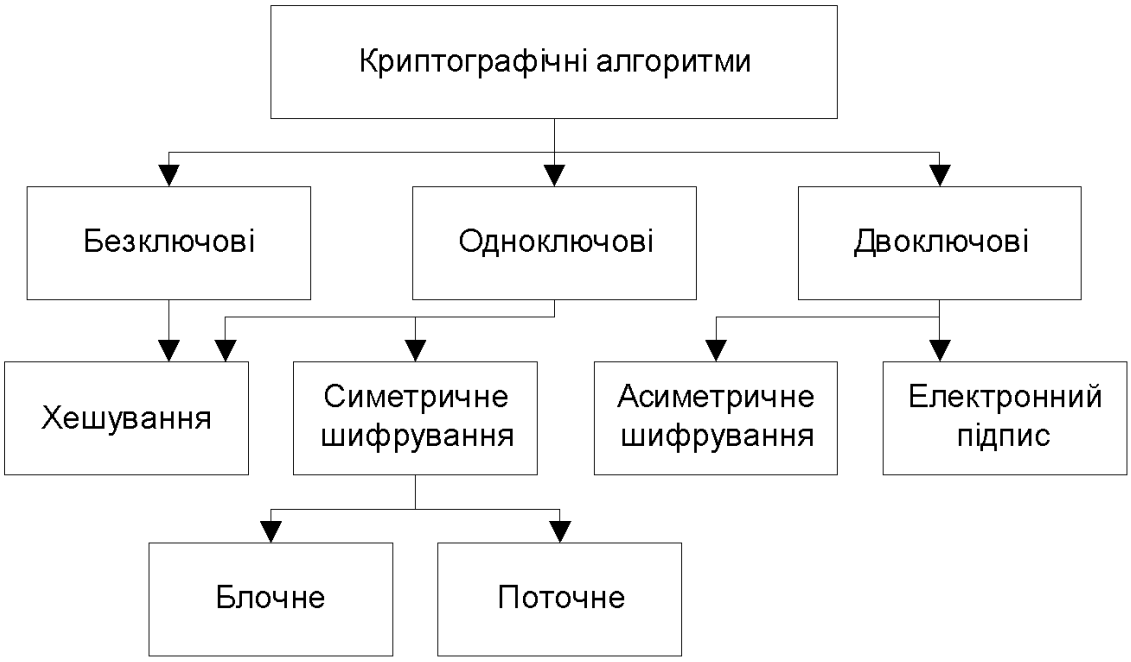

Рисунок 1.3 – Класифікація криаптоалгоритмів

Будемо використовувати ту, що ділить їх залежно від кількості ключів:

– безключові алгоритми – для розрахунку не використовуються ключі;

– одноключові алгоритми – для розрахунку використовується лише один ключ, секретний;

– двоключові алгоритми – для розрахунку використовуються два ключі, секретний та відкритий.

Більш детальна класифікація алгоритмів зображена на рисунку 1.3.

1.1.5 Важливість якісного шифрування інформації

Як було зазначено вище, основною метою шифрування даних є захист їх від несанкціонованого доступу. Широке використання інформаційних технологій робить актуальною та закономірною проблему захисту інформації. Дослідження показують, що лише половина фахівців з інформаційної безпеки вважають свою компанію чи установу такою, що готова протистояти сучасним інформаційним загрозам, зокрема і таким, що можуть призвести до неконтрольованого поширення інформації за межі інформаційних систем, у яких вона обробляється.

У сучасному інформаційному суспільстві велику загрозу конфіденційності та цілісності інформації представляє кіберзлочинність. Зростання кількості кібератак та доступність програмно-технічних засобів для їх реалізації зумовлює необхідність розробки сучасних засобів інформаційної безпеки громадян та держави в цілому. Отже, актуальною задачею є створення та вдосконалення систем захисту інформації, зокрема алгоритмів криптографічного захисту, що в більшості випадків є базовим ядром таких систем. На даний час основним питанням, що підлягає вирішенню, є збільшення об’ємів інформації, що може оброблятися функціями криптографічного перетворення та посилення технічного захисту даних в Україні.

Поняття технічного захисту інформації в Україні визначає такі джерела загроз для інформації [2]:

- окремі фізичні особи;
- інші держави;
- злочинні угрупування;
- політичні партії;
- суб’єкти підприємницького процессу;
- ненавмисні та навмисні дії співробітників;
- стихійні лиха та техногенні катастрофи.

Мотивація цих джерел може бути різною: від повної відсутності у стихійних лих до економічних та політичних переваг (таблиця 1.1).

Таблиця 1.1. Джерела загроз для інформації

Джерела	Мотивація
Інші держави	Одержання переваг у зовнішньополітичній, зовнішньоекономічній, військовій, та інших сферах
Політичні партії	Одержання переваг у політичній боротьбі, боротьбі за владу
Суб’єкти підприємницької діяльності	Одержання політичних, економічних переваг, нанесення шкоди
Окремі фізичні особи	Самоствердження, отримання економічних переваг і винагород
Навмисні та ненавмисні дії персоналу	Помилки персоналу,низька кваліфікація працівників; образа, зрада, промушення
Стихійні лиха та техногенні катастрофи	Відсутність мотивації

На основі поданої класифікації джерел можна скласти одну із можливих класифікацій загроз для інформації:

- наслідки помилок проектування системи захисту;
- наслідки помилок персоналу;
- наслідки стихійних лих і техногенних катастроф;
- навмисні дії порушників;
- відмови обладнання.

У теорії криптографії доведено, що якщо система захисту інформації побудована беручи до уваги усі сучасні методи та засоби захисту, а підприємство має ретельно відібраний та добре навчений персонал, який не допускає помилок, то навмисні дії порушників у такій системі неможливі [2]. Це правило вироблено роками досвіду спеціалістів захисту інформації і має універсальний характер. Воно не залежить від рівня системи захисту, сумлінності користувачів та адміністраторів, апаратного та програмного забезпечення.

Тому важливо правильно класифікувати інформацію для шифрування, аби можна було її ефективно та якісно зашифрувати та виключити випадок несанкціонованого доступу до інформації.

1.2 Аналітичний огляд методів класифікації інформації

Міркуючи про методи класифікації інформації дуже важливо розуміти визначення поняття «типи даних». Характеристика, що визначена для об’єкта та передбачає множину припустимих значень, формат їхнього зберігання, розмір виділеної пам’яті та набір операцій, які можна виконувати над ними вважається поняттям типів даних. Дане поняття стосується будь-яких даних, які існують у цифровому просторі.

У найбільш широкому плані типи даних трактуються, як множина значень та операцій над ними [2]. При цьому, на думку автора цитованої роботи, тип даних

характеризується одночастно множиною допустимих значеньб що можуть приймати дані, що належать до цього типу та набором операцій, що можна виконувати над даними, що належать до цього типу. Першу властивість можна розглядати як теоретико-множинне визначення поняття типу, а друге – як процедурне визначення.

Аналізуючи визначення типу даних, можна помітити, що концепція в будь-якому випадку являє собою задані розмірні та структурні характеристики осередку пам'яті, в яку можна помістити деяке значення, яке відповідає цим характеристикам.

1.2.1 Актуальність можливості розпізнавання тексту

Не дивлячись на те, що в наш час більшість документів формується на комп'ютерах, завдання заведення повністю електронного документообігу ще далека до повної реалізації. Зазвичай, існуючі програми охоплюють діяльність окремих організацій, а обмін даними між організаціями здійснюється за допомогою звичайних паперових документів [4].

Проблема трансферу інформації з паперових на електронні носії актуальна не тільки в рамках потреб, що виникають в системах документообігу. Сучасні технології дозволяють нам істотно спростити доступ до ресурсів, накопичених за весь час, за умови, що вони будуть переміщені в електронний вигляд.

Найбільш швидким і простим є зчитування документів за допомогою сканерів [4]. Результатом роботи являється зображення документа. Більш ефективним, у порівнянні з графічним, є текстове представлення інформації. Цей варіант дозволяє істотно зменшити витрати на передачу і зберігання інформації. Тому найбільший практичний інтерес представляє саме переклад паперових носіїв в текстовий документ.

1.2.2 Аналіз проблем, пов’язаних з розпізнаванням слів

Існує декілька великих проблем, пов'язаних з розпізнаванням друкованих і рукописних символів. Найбільш важливі з них такі:

- різні форми обрисів символів;
- викривлення зображень букв;
- варіації масштабу і розмірів символів.

Кожен символ може бути написаний різними стандартними шрифтами, наприклад (Times, Courier , Orator, Gothic, Elite), а також - багатьма нестандартними шрифтами, які використовуються в різних наукових напрямах. При цьому різні символи можуть бути схожими по обрисах. Наприклад, "U" і "V", "S" і "5", "Z" і "2", "G" і "6".

Змінення цифрових зображень текстових символів можуть бути через:

- порушення друку, зокрема, те, що погано надрукувалося, "злипання" сусідніх символів, плями і помилкові символи на фоні поблизу тексту і т. д.;
- зміщення букв і цифр або частин символів відносно очікуваного положення у тексті;
- зміну нахилу символів;
- змінення форми символу за допомогою цифрування зображення з "грубим" розривом;
- ефекти освітлення при зчитуванні сканером.

Істотним є також вплив остаточного масштабу друку. У загальноприйнятій термінології масштаб 10, 12 або 17 визначається тим, що в дюймі рядки поміщаються 10, 12 або 17 символів. При цьому, звичайно, символи масштабу 10 більше і ширше символу масштабу 12.

1.2.3 Аналіз визначення глибинного та машинного навчання

Для розпізнавання символів та слів у текстах найбільше підійде така методика як глибинне навчання. Глибинне навчання, простіше кажучи, це підрозділ машинного навчання в штучному інтелекті (див. рис. 1.4), алгоритми якого засновані на біологічній структурі та функціонуванні мозку і покликані наділити машини інтелектом.

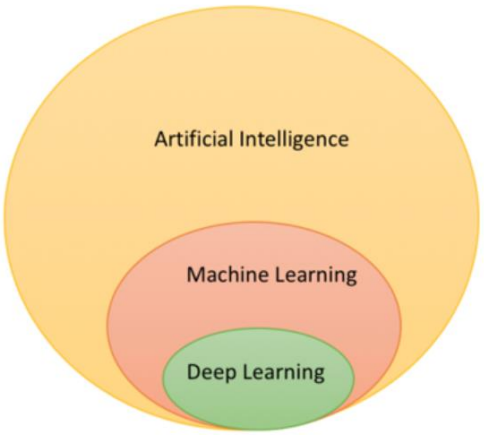

Рисунок 1.4 – Визначення глибинного навчання

Штучний інтелект (ШІ) в найбільш широкому сенсі - це розум, вбудований в машину. Зазвичай ми впроваджуємо в машини інтелект - в результаті машина може самостійно приймати рішення. Наприклад, пральна машина визначає необхідний обсяг води, а також необхідний час для замочування, прання і віджимання. Таким чином, вона приймає рішення, ґрунтуючись на конкретних вступних умовах, а значить робить свою роботу розумніше. Або, наприклад, банкомат, який видає потрібну вам суму, складаючи правильну комбінацію з наявних в ньому банкнот. Такий інтелект впроваджується в машини штучним шляхом - звідси і назва "штучний інтелект".

Важливо відзначити, що інтелект тут запрограмований явно, тобто створений на основі детального списку правил виду "якщо ..., то ...". Інженер-проектувальник ретельно продумав всі можливі комбінації і створив систему, яка приймає рішення, проходить по ланцюжку правил. А що якщо нам потрібно впровадити інтелект в машину без явного програмування, тобто, щоб машина вчилася сама? Тут ми і підходимо до теми *машинного навчання*.

Машинне навчання - це процес впровадження інтелекту в систему або машину без явного програмування.

Прикладом машинного навчання могла б стати система, яка пророкує результат іспиту на основі попередніх результатів і характеристик студента. В цьому випадку рішення про те, здасть студент іспит чи ні, грунтувалося б не на детальному списку всіх можливих правил - навпаки, система навчалася б сама, відстежуючи патерни в попередніх наборах даних.

Де ж в цьому контексті місце глибокого навчання? Машинне навчання успішно вирішує багато питань, але часом не може впоратися із завданнями, які здаються людям дуже простими. Наприклад, воно не може відрізнити кішку від собаки на зображенні або чоловічий голос від жіночого на аудіозаписи і т. п. Результати застосування машинного навчання найчастіше погані при обробці зображень, аудіо та інших типів неструктурованих даних.

При пошуку причин таких результатів прийшло розуміння - ідея скопіювати біологічні процеси людського мозку, який складається з мільярдів нейронів, пов'язаних і скоординованих між собою особливим чином для вивчення нового. Вивчення нейронних мереж йшло одночасно з цим вже кілька років, але прогрес був невеликим через обмеження в даних і обчислювальних потужностях того часу. Коли машинне навчання і нейромережі були достатньо вивчені, з'явилося глибоке навчання, яке передбачало створення глибоких нейронних мереж, тобто довільних нейромереж з набагато більшою кількістю шарів.

1.2.4 Структура нейронної мережі

Спрощена версія глибокої нейромережі може бути представлена як ієрархічна (шарувата) структура з нейронів (подібно нейронам у мозку), пов'язаних з іншими нейронами. На основі вхідних даних одні нейрони передають команду (сигнал) іншим і таким чином формують складну мережу, яка навчається за допомогою певного механізму зворотного зв'язку. На рисунку 1.5 зображена глибока нейронна мережа з кількістю шарів N.

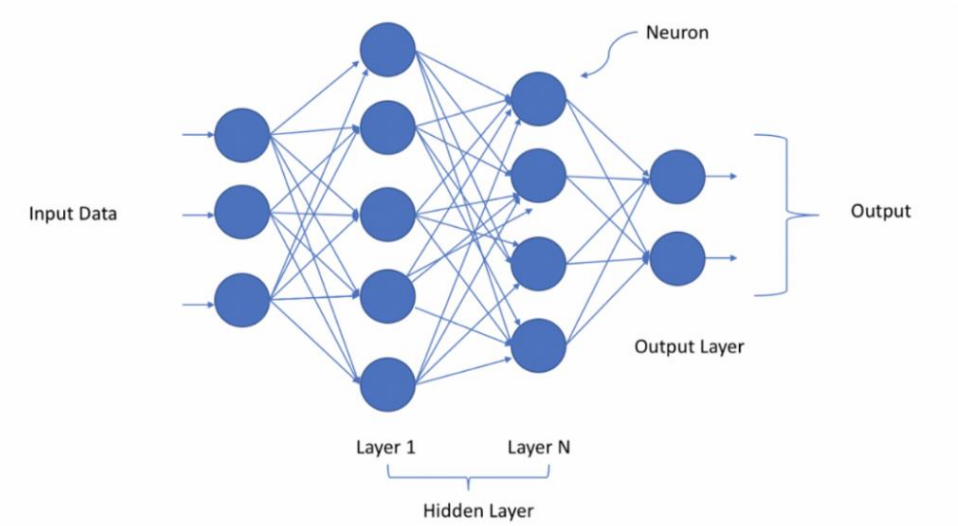

Рисунок 1.5 – Глибока нейронна мережа з кількістю шарів N

Як видно на рисунку 1.5, початкові дані передаються нейронам на першому (не прихованому) шарі, ті в свою чергу передають результуючі дані нейронам на наступному шарі і далі до виходу. Вихід може являти собою прогноз (наприклад "Так" / "Ні"), показаний за допомогою ймовірності. На будь-якому шарі може бути один або безліч нейронів, кожен з яких має обчислювати невелику функцію, що являє собою функцію активації. Ця функція імітує трансфер сигналу пов'язаних з

попередніми нейронам. Якщо вихід вхідних нейронів перевищує допустиме значення, вихідне значення просто пропускається і процес проходить далі. Відповідність між двома нейронами сусідніх шарів має вагу. Вага показує вплив початкових даних на вихід для наступного нейрона і наступний остаточний вихід. Початкові ваги нейромережі непередбачувані, але у ході навчання моделі вони постійно змінюються і навчаються передбачати вірне фінальне значення. У процесі аналізу нейромережі можна виявити кілька зрозумілих структурних елементів (нейрон, шар, вага, вхід, вихід, функція активації і у кінці механізм навчання, або оптимізатор), які допомагають їй поступово оновлювати ваги (спочатку з випадковими значеннями) на ті, які більше підходять для точного прогнозу виходу.

1.2.5 Процес навчання нейронної мережі

Природа процесу навчання людського мозку цілком очевидна. Ми вивчаємо відхилення від типової людини, тобто те, наскільки очі конкретної людини відрізняються від типового ока замість того, щоб вивчати будову людини для розпізнавання людей. Потім ця інформація перетворюється в електричний сигнал деякої сили. Відхилення всіх інших частин обличчя від типових вивчаються однаково. Всі ці відхилення в результаті збираються в нові ознаки і дають початкове значення. Все описане відбувається за частку секунди, і ми просто не встигаємо зрозуміти, що сталося в нашій підсвідомості.

Як показано вище, нейронна мережа намагається імітувати той самий процес, використовуючи математичний підхід. Вхідні дані приймаються нейронами першого шару, і в кожному нейроні обчислюється функція активації. Виходячи з простого правила, нейрон передає початкове значення наступного нейрона, як і мозок людини вивчає відхилення. Чим більший вихід нейрона, тим важливішою є відповідна вхідна

функція. У наступному шарі ці функції об’єднані в нові, які мають форму, яка нам досі незрозуміла, але система засвоює їх інтуїтивно. Неодноразово повторюваний цей процес призводить до утворення складної мережі зв’язків.

1.3 Аналітичний аналіз існуючих секцій на конференціях

Перед тим, як почати класифікацію напрямів конференцій, треба дати визначення терміну наукова конференція. Наукова конференція — форма організації наукової діяльності, при якій учасники презентують свої роботи. Зазвичай за деякий час повідомляється про всі деталі проведення конференції. Потім учасники можуть почати відправку тез і доповідей та інколи грошових внесків.

Часто конференції бувають багатопрофільні, тому вся конференція розбивається на так звані секції – напрями презентацій ддя учасників:

- сільськогосподарські науки;
- ветеринарні науки;
- біологічні науки;
- медицинські науки;
- фармацевтичні науки;
- хімічні науки;
- технічні науки;
- фізико-математичні науки;
- географічні науки;
- геолого-мінералогічні науки;
- архітектура;
- астрономія;

- педагогічні науки;
- психологічні науки;
- соціологічні науки;
- журналістика;
- мистецтвознавство;
- історичні науки;
- філософські науки;
- література;
- філологічні науки.

1.4 Аналоги програмного продукту

За останні пару років стався різкий ріст популярності глибинних нейронних мереж, але більша частина теорії була написана ще у минулому столітті. Для такого росту популярності є дві основні причини. Перша – це істотний зріст продуктивності компьютерів. Тепер навіть у маленьких девайсах є багатоядерні процесори з великою обчислювальною потужністю і графічний прискорювач, який можна використовувати не тільки для компьютерних ігор, але і для обчислювань, у тому числі і для навчання нейронних мереж. Інша причина популярності нейронних мереж полягає в тому, що зараз стався різкий зріст накопиченого об’єму даних і цих даних стало достатньо аби навчити нейронну мережу вирішувати складні задачі, наприклад вести машину без водія. Також грає роль те, що зараз є велика кількість систем глибинного навчання нейронних мереж і вже інує багато популярних додатків, що використовують нейронні мережі.

Розглянемо додатки на основі нейронних мереж, що мають можливість розпізнавання тексту, оскільки саме така функція передбачається у розроблюваному додатку.

Будемо порівнювати додатки по декількох пунктах:

– сканування та оптичне розпізнавання тексту;

– витяг тексту з зображень;

– можливість перекладу тексту на декілька мов;

– вбудований словник для перевірки орфографії;

– обробка у вбудованому текстовому редакторі;

– можливість визначення теми тексту.

Після порівняння програм було отримано такі результати:

Додаток ABBYY Finereader - це оптична система розпізнавання тексту. Він призначений для перетворення з можливістю редагування відсканованих документів, документів PDF та файлів зображень документа, включаючи цифрові фотографії. Ця программа має високу точність розпізнавання, може розпізавати тексти, написані багатьма шрифтами, а також дозволяє розпізнавати текст, отриманий з камери або вбудованої камери мобільного телефону. Додаткові налаштування попередньої обробки зображень дозволяють значно покращити якість зображення та отримати кращі результати розпізнавання. Також програма ABBYY Finereader має можливість розпізнавання текстів різними мовами, у тому числі арабською, в'єтнамською, корейською, китайською та японською.

Більш того, результати розпізнавання програми можна зберігати не тільки на локальному комп'ютері , але і у хмарному сховищі, для того, щоб отримати доступ до них з будь-якого девайсу.

У свою чергу Readiris Pro це програма для перетворення документів в різні формати. Якщо у вас є сканер або МФУ, то можна сканувати і розпізнавати текст через цю утиліту.

Дана програма дозволяє працювати без установки драйвера для вашого обладнання. Досить в налаштуваннях вибрати одну з безлічі моделей техніки.

Програмне забезпечення платне. Можна подивитися основні можливості Readiris Pro і протестувати, практично повноцінну версію, на 10 днів або 100 перетворених документів.

ABBYY Screenshot Reader - проста і зручна програма для фотографування будь-якої області екрана. Це допоможе зберегти зображення всього екрану, вікна програми або вибраної вручну області. Ви можете використовувати ці зображення під час підготовки презентацій або надіслати їх електронною поштою колегам та друзям. ABBYY Screenshot Reader дозволяє зберігати скріншоти зображень в форматах JPEG, Bitmap або PNG. Текстові знімки екрану можна зберігати як редаговані тексти в форматах, - RTF, .TXT, .DOC або .XLS.

За допомогою ABBYY Screenshot Reader ви можете швидко зробити знімок потрібної області тексту та створити його у редагованому форматі. Потім ви можете вставити надрукований текст у документ, використовуючи його, або зберегти в новому документі Microsoft Word та Excel. Програма розпізнає тексти 188 мовами та підтримує 24 мовні інтерфейси.

Також був проаналізований такий сервіс як Google Translate. Google Translate - це найбільша система для роботи з мовами. За статусом на червень 2020 року в ній повністю реалізовано 108 мов світу. За його допомогою можна не тільки перекласти текст на іншу мову, але і цілі веб-сторінки. Це можна зробити як на самій сторінці, так і вставивши URL сторінки у перекладач і, більш того, тексти, що перекладалися, можна зберігати.

Також у Google Translate є багато функцій, пов’язаних з камерою та розпізнаванням тексту. Наприклад, за допомогою даної програми можна наводити камеру на будь-які написи та одразу отримувати переклад у реальному часі. Також можна фотографувати текст і виділяти окремі частини для перекладу.

Використовуючи зібрані дані, можна створити таблицю, яка б наглядно демонструвала порівняння характеристик програм (див. табл. 1.2).

Таблиця 1.2 – Порівняння аналогів програми

	ABBYY Finereader	Readiris Pro	ABBYY Screenshot Reader	Google Translator
Можливість розпізнавання тексту	+	+	-	+
Підтримка декількох мов	+	-	+	+
Визначення теми тексту	-	-	-	-
Стиснення тексту	+	+	-	-

Отже поглянувши на дану таблицю можна помітити, що кожна програма має свої переваги, але у той же час має ряд недоліків. Більше того: у жодної з програм немає можливості визначати тему тексту, з яким проходить робота.

Також важливо відмітити, що у даних програм немає функції шифрування розпізнаного тексту, що є важливим фактором для розроблюваної програми. Тож буде доцільно розробити систему, яка б перекривала більшість або навіть всі недоліки конкурентів та дозволяла використовувати, наприклад, існуючі бібліотеки для розпізнавання тексту, що значно пришвидшить її здатність до навчання.

1.5 Постановка задачі

Виходячи з попередньо проведеного аналізу можна встановити, що для системи, яка буде проектуватися, дуже важливим фактором успіху вважається висока захищеність даних продукту, що досягається точністю класифікації даних для подальшого шифрування тез. Для цього можна створити програмну систему, яка використовує деякі існуючі бібліотеки для розпізнавання тексту для їх об’єднання та вдосконалення.

Аналізуючи вищеперераховані вимоги, можна визначити, що однією із задач являється розроблення програмної системи з високим рівнем вміння розпізнавання текстів, яка могла б класифікувати їх по напрямах.

Вхідними параметрами для системи, що розроблюється, мають бути некласифіковані тексти тез.

Проміжною вихідною інформацією є інформація про секцію, тобто напрям до якого належать певний текст.

Кінцева вихідна інформація – це прокласифіковані та зашифровані тексти тез.

Таким чином основними задачами роботи є:

- обробка вхідних текстів тез;
- зберігання, завантаження та видалення даних про статті;
- використання сучасних бібліотек класифікаціїї даних;
- створення веб-додатку для представлення результатів обробки;
- створити можливість продивлятися наявні конференції.

2 ВИБІР МЕТОДІВ ШИФРУВАННЯ

2.1 Двоключові алгоритми

З поданої раніше класифікації буде розглянуто та проаналізовано двоключові криптоалгоритми, що дають змогу передавати повідомлення через відкриті канали зв’язку.

Вони в свою чергу поділяються на асиметричне шифрування та електронний підпис.

2.1.1 Визначення асиметричного шифрування

Асиметричне шифрування – це шифрування, що засноване на використанні пари ключів – закритому (секретному) та відкритому.

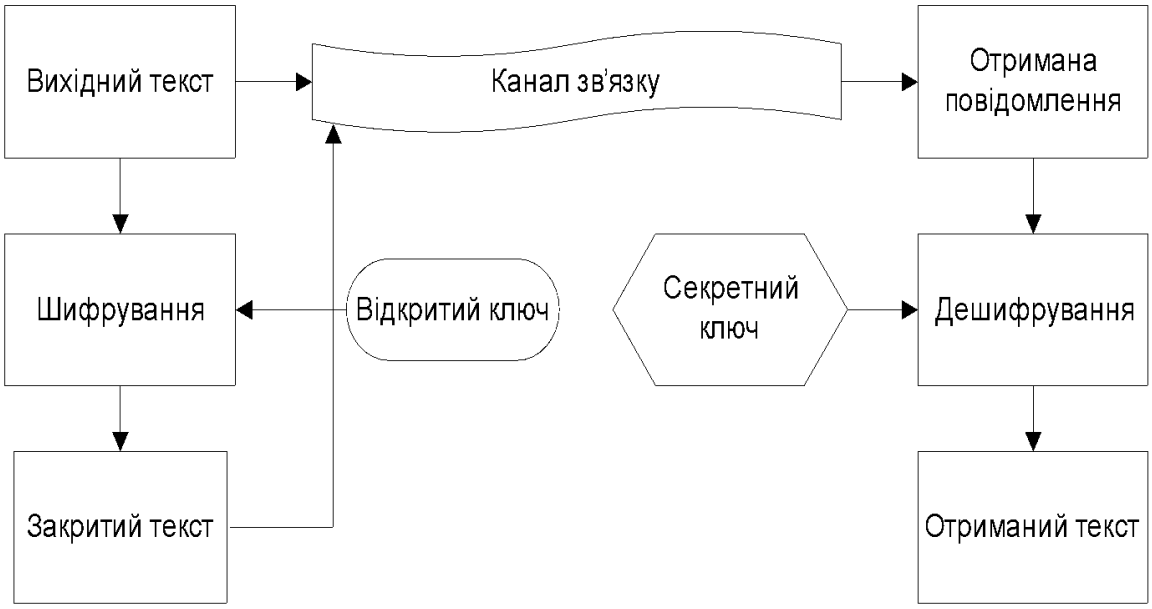

Рисунок 2.1 – Загальний алгоритм асиметричного шифрування

Суть методу полягає в тому, що вихідний текст (документ) можна зашифрувати будь-яким ключем з пари, а розшифрувати тільки за допомогою другого ключа з даної пари. Закритий ключ буде відомим лише власнику, а відкритий спокійно розповсюджується.

На рисунку 2.1 схематично зображений алгоритм асиметричного шифрування.

2.1.2 Підвищення криптостійкості за допомогою хешування

В реальності немає потреби шифрувати документ, досить і обчислення його хеш-коду.

Хешування – це трансформація масиву початкових даних випадкової довжини в (вихідний) бітовий рядок певної довжини, що виконується певним алгоритмом (хеш-функцією) [5].

Хеш-функція – це легко обчислювана функція, що стискає вхідні значення та яка не має ефективного пошуку колізій [5].

В результаті перетворень і отримують хеш-код. Він є унікальним для будь-якого об’єму інформації. Якщо хеш-код створений, наприклад для текстового документа, то при кожному редагуванні буде відбуватись і його змінення. Тому основними властивостями є:

- унікальність – кожному набору інформації притаманний певний, унікальний хеш-код;
- змінність – навіть незначні зміни вхідних даних призводять до кардинальної зміни хеш-коду;
- незворотність – не існує функції, за допомогою якої можливо відновити з хеш-коду вхідний документ.

Існує безліч функцій, за допомогою яких можна отримати хеш-код, їх класифікація представлена на рисунку 2.2.

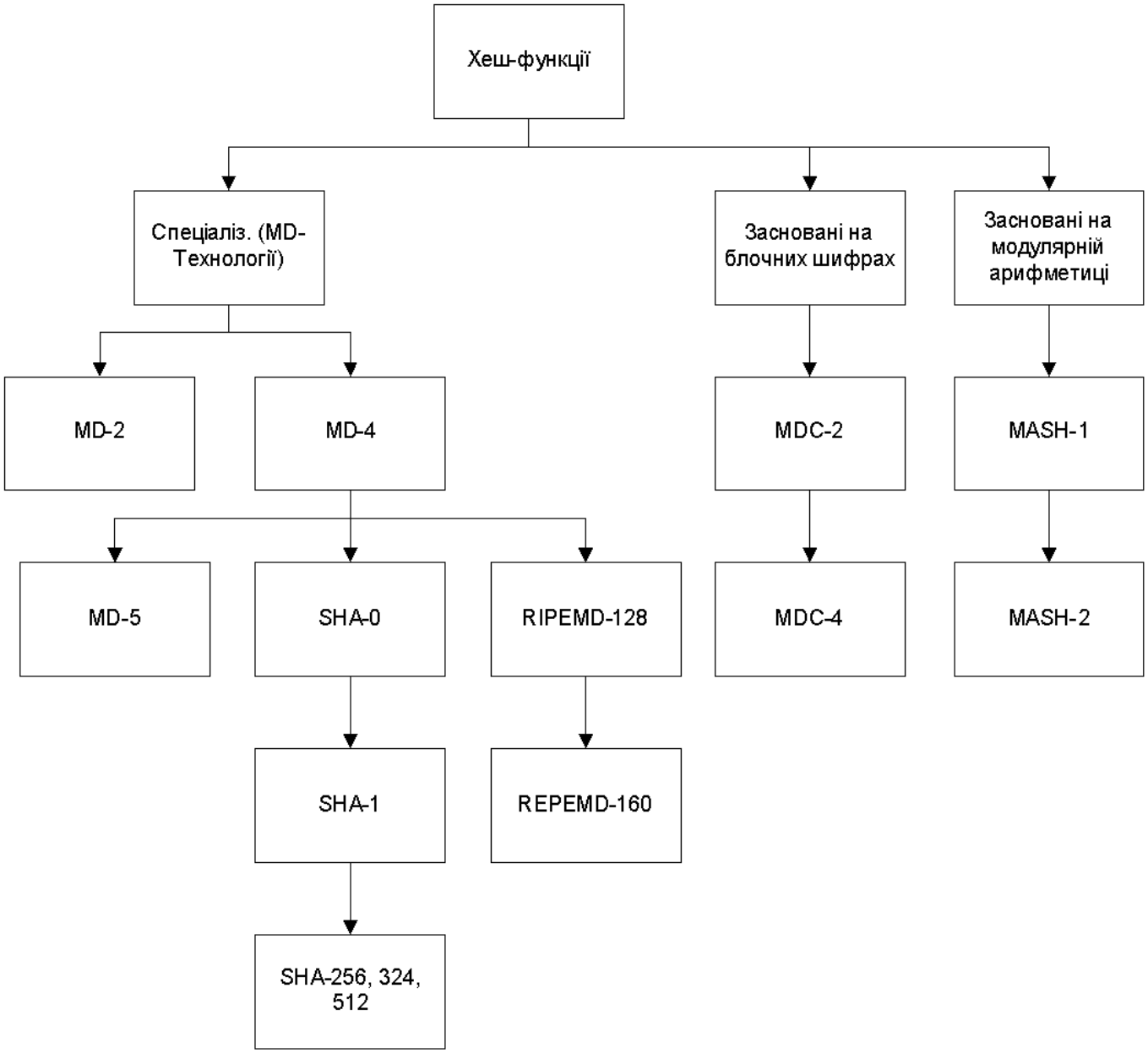

Рисунок 2.2 – Класифікація хеш-функцій

2.1.3 Визначення електронного цифрового підпису

У 1976 році Уітфілдом Діффі та Мартіном Хелманном було введено таке поняття як електронний цифровий підпис.

Пізніше, в 1977 році Рональдом Ривестом, Аді Шаміром і Леонардом Адлеманом розробили криптосистему RSA. Її можна було використовувати для створення примірного ЕЦП.

Після RSA були розроблені й інші алгоритми, наприклад, алгоритми цифрової підписи Рабина, Меркле.

У 1984 році Шафі Гольдвассер, Сильвіо Мікалі та Рональд Ривст сформулювали вимоги безпеки до алгоритмів ЕЦП, а також описали атаки на ЕЦП.

Електронний цифровий підпис (ЕЦП) – це деяка додаткова інформація, яка відповідає даному електронному документу. Сформувати підпис міг тільки справжній власник, у якого є закритий ключ.

Також під ЕЦП розуміють криптографічну систему, яка дозволяє підписувати цифрові повідомлення и перевіряти правильність формованих цифрових підписів.

ЕЦП зазвичай призначений для:

- визначення особи, що підписала електронний документ;
- контролю цілісності документа;
- захисту документа від змін;
- доказу авторства документа.

Розглянемо більш детально процеси генерації ЕЦП.

Етап генерації ключів призначений для генерації відкритого та закритого ключа для кожного абонента. Вони пов’язані один з одним особливим математичним співвідношенням. Закритий ключ зберігається в таємниці і використовується для підпису документа. Відкритий ключ відомий всім користувачам і застосовується для перевірки підпису. Він є дуже важливим атрибутом, тому що дозволяє

аутентифікувати автора підпису і справжність електронного документа, але не дозволяє провести обчислення секретного ключа.

Цей етап може бути виконаний двома варіантами, або абонент самостійно генерує ключі, або в певних випадках за генерацію ключів відповідає спеціальний центр.

На етапі підпису документа відбувається стиснення документа за допомогою хеш-функції, значення якої залежить від змісту документа. Далі, залежно від певного алгоритму ЕЦП проходить безпосередньо підпис документу [10].

На етапі перевірки підпису необхідно знати відкритий ключ користувача, що поставив підпис. Спочатку обчислюється хеш-функція документа. Потім проводяться математичні обчислення, в залежності від алгоритму ЕЦП. Це означає, що відбувається перевірка тієї чи іншої відповідності, що пов’язує хеш-функцію документа, електронний підпис під документом і відкритий ключ користувача.

Якщо відповідність виявляється виконаною, то електронний підпис визнається дійсним, а сам документ достовірним, в іншому випадку документ вважається підміненим, а ЕЦП під ним недійсним [6].

На рисунку 2.3 схематично зображені описані вище процеси.

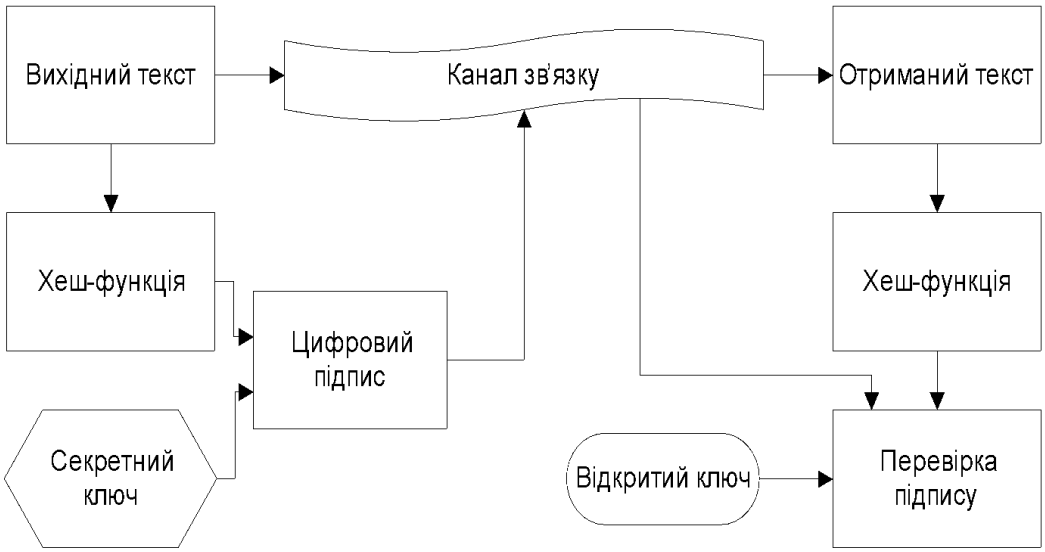

Рисунок 2.3 – Загальний алгоритм ЕЦП

2.1.4 Алгоритми ЕЦП

Існує багато алгоритмів ЕЦП. Найбільш відомими є: RSA, DSA та Ель-Гамаля (ElGamal) [11].

1) Стандарт асиметричного шифрування RSA.

Вибрати два випадкових простих числа p та q. Розрахувати N=pq. Вибрати випадкове ціле число 1<e<φ(N), що задовольняє умові gcd e,φN =1. Розрахувати число d, таке що N) .

Генерація підпису – S=mdmod N.

Перевірка (верифікація) підпису – m≡Se(mod N).

На рисунку 2.4 схематично зображено описаний вище алгоритм.

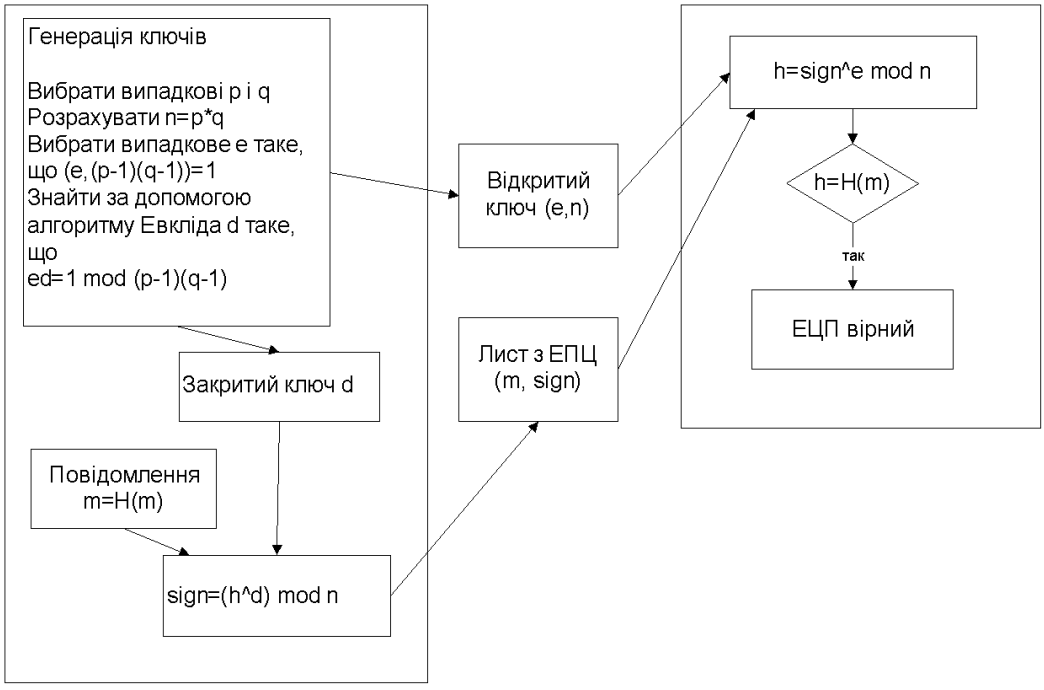

Рисунок 2.4 – Схема ЕЦП на основі алгоритму RSA

2) Алгоритм DSA.

Вибрати хеш-функцію H(x). Вибрати два випадкових простих числа *p* и q так, щоб q |p-1, g∈F*=0,1,…,p-1 – елемент порядку g. Закрытий ключ x∈(0,q), відкритий ключ y=gx (mod p).

Генерація підпису. Вибрати два випадкове число k∈(0,q). Розрахувати r=gk mod p(mod q) , s=k-1 Hm+x*r(mod q) . вибір нового k, якщо r=0 или s=0.

Підпис – (r,s).

Перевірка (верифікація) підпису. Розрахувати: w=s-1 mod q, u1=Hm*w(mod q), u2=r*w(mod q) та v=(gu1*yu1) mod p (mod q).

На рисунку 2.5 схематично зображено описаний вище алгоритм.

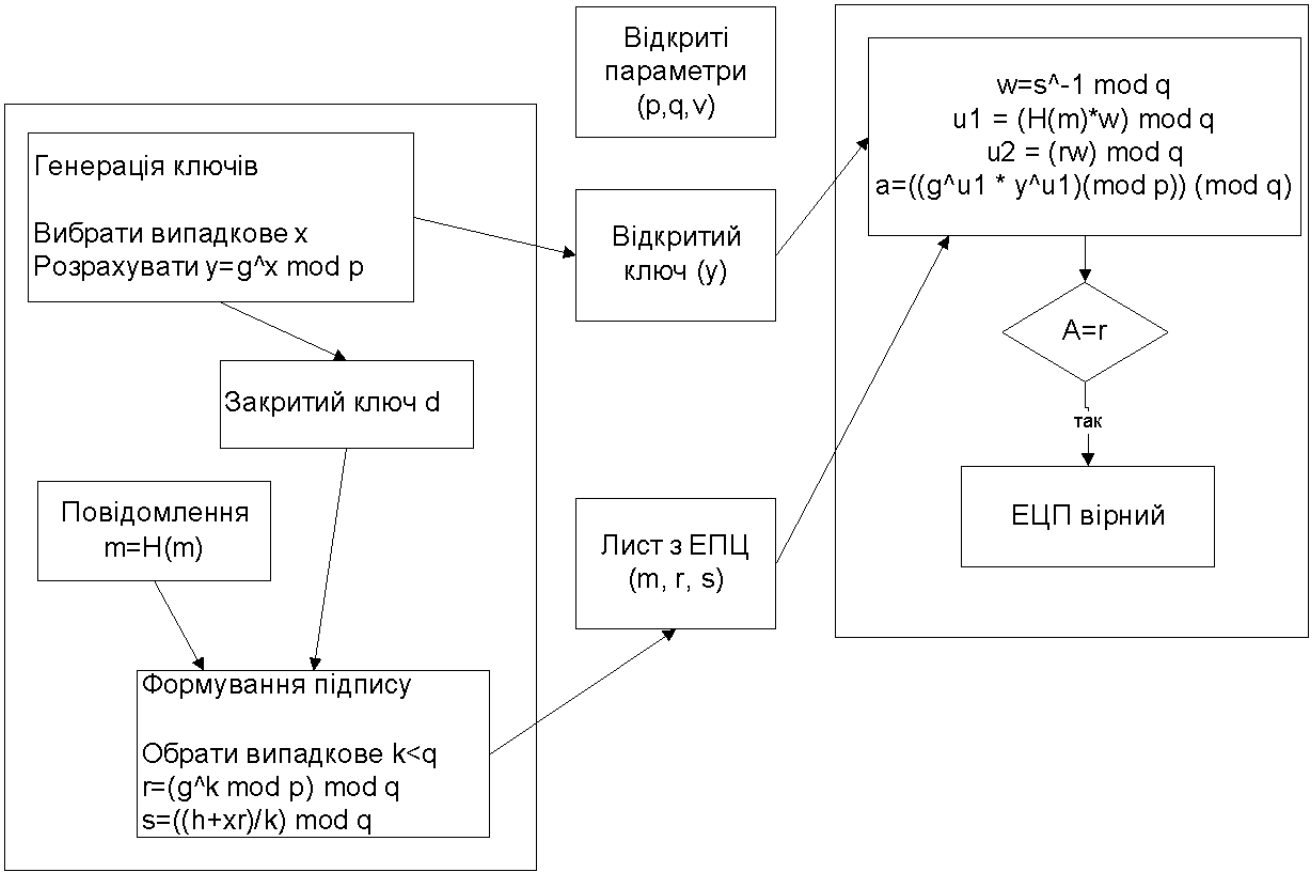

Рисунок 2.5 – Схема ЕЦП на основі алгоритму DSA

3) Алгоритм Ель Гамаля.

Вибрати два випадкових простих числа *p* и gFN* – випадковий утворюючий елемент. Вибрати випадкове число x таке, що 1<x<p-1. Відкритий ключ – y=gx(mod p) , закритий (секретний) ключ – x.

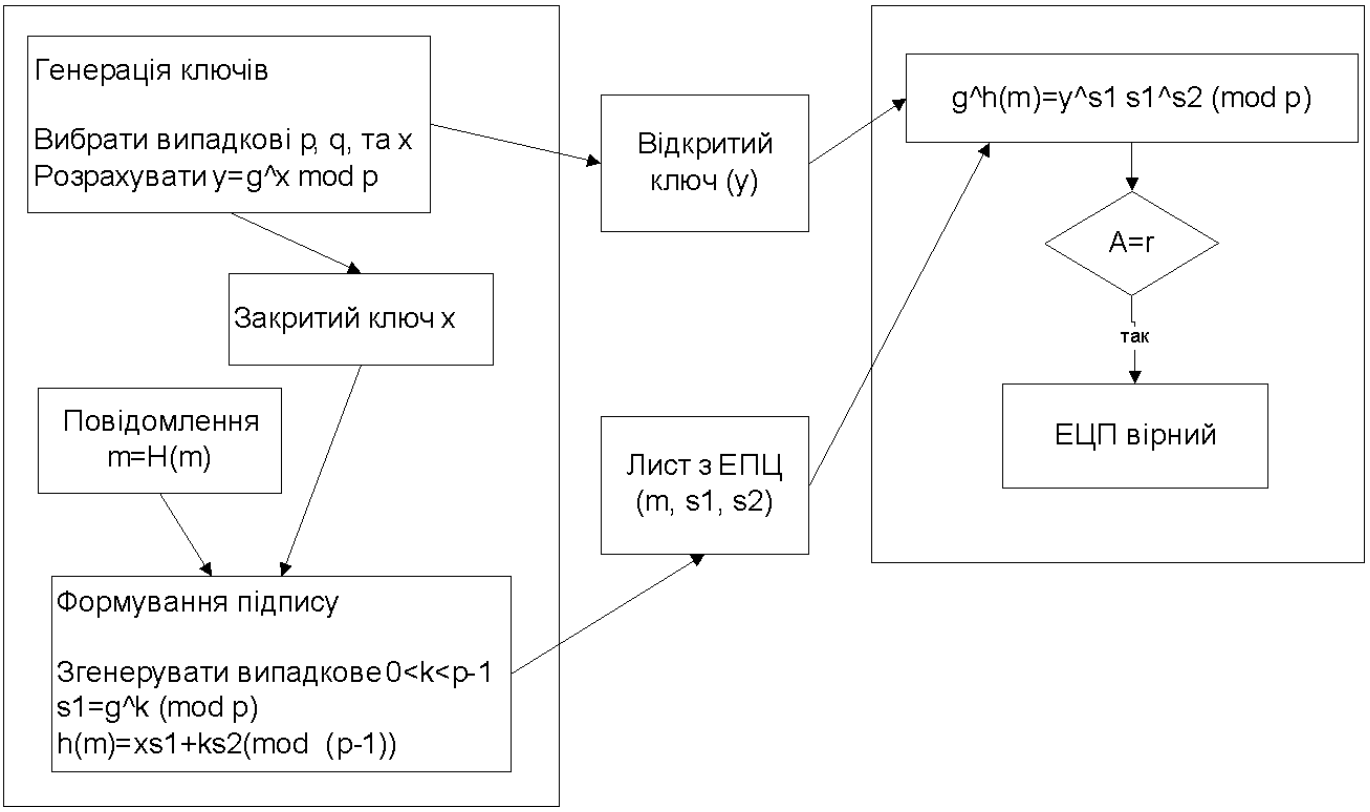

Рисунок 2.6 – Схема шифрування алгоритму Ель-Гамаля

Генерація підпису. Генеруємо випадкове число k, 0<k<p-1, що задовольняє виразу gcd k,p-1 =1. Перша частина підпису – s1=gk(mod p). друга – s2 шукається як вирішення рівняння hmxs1+ks2(mod (p-1)).

Перевірка (верифікація) підпису – gh(m)ys1s1s2 (mod p).

На рисунку 2.6 схематично зображено описаний вище алгоритм.

2.1.5 Атаки на ЕЦП

Витривалість більшості схем електронного підпису залежить від стійкості хеш-функцій та асиметричних алгоритмів шифрування. Тому розглянемо існуючі загрози та атаки на ЕЦП.

Класифікація атак на схеми електронного підпису [7]:

– атака з відкритим ключем, що відомий. Вважається найслабшою, бо зловмисник завжди може отримати відкритий ключ користувача;

– атака з підписаними повідомленнями, що відомі. Крім відкритого ключа у зловмисника ще є набір повідомлень, підписаних ключем;

– проста атака з вибором повідомлень, що підписані. Існує можливість вибору підписаного повідомлення, відкритий ключ в даному випадку отримується після вибору певного повідомлення;

– спрямована атака з вибором повідомлень. Схожа на попередню атаку, але при отриманні певних підписаних повідомлень відомий і відкритий ключ;

– адаптивна атака з вибором повідомлень. Зловмиснику відомий відкритий ключ, можна також вибрати підписані повідомлення і відомі підписи всіх повідомлень, щ обули підписані раніше.

Типи загроз електронного:

– повне розкриття. Зловмисник знаходить секретний ключ користувача;

– універсальна підробка. Зловмисник знаходить алгоритм, що функціонально схожий з алгоритмом генерації ЕЦП;

– селективна підробка. Підробляється підпис певного повідомлення;

– екзистенціальна підробка. Підробка підпису принаймні одного випадково обраного повідомлення.

Під час використання електронного підпису можна знайти або запобігти наступним порушенням [7]:

- відмова від процесу відправки інформації;
- зміна електронного документу;
- підробка інфоормації;
- нав’язування повідомлень в процесі передачі;
- імітація передачі інформації.

Але в наш час існують і такі порушення, від яких неможливо позбавити систему обміну повідомленнями. Такими порушеннями вважаються повтор передачі повідомлення та підробка часу відправлення повідомлення.

2.2 Управління ключами

Під час роботи з системами криптографічного захисту існує проблема, що пов’язана з створенням, зберіганням та розподіленням ключів. Шифрування є ефективним лише тоді, коли вирішені ці питання. Адже майже всі криптосистеми мають секретний ключ, за винятком хеш-функцій. Дана проблема пов’язана з тим, що інколи простіше провести атаку не на криптографічний алгоритм, а на ключову систему.

Ключова система – це сукупність ключів, що використовуються в криптосистемі. За допомогою управління ключовою системою визначається стійкість криптосистем.

Розглянемо функції управління ключами в криптосистемах:

- формування (генерація) криптографічних ключів;
- розподіленея і аутентифікація секретних ключів;
- аутентифікація відкритих ключів;
- використання ключів;

– зберігання ключів (підтримка цілісності відкритих ключів та захисту особистих ключів);

– депонування ключів;

– заміна ключів;

– скасування ключів;

– видалення (знищення) ключів.

2.3 Довжина ключа

Зазвичай ступінь захищеності інформації залежить не тільки від вибраного алгоритму шифрування, а і від довжини ключа (біт). Зрозумілим є той факт, що чим більший ключ, тим кращим буде і захист. Адже кожне побітове додавання збільшує кількість можливих варіантів ключів в два рази. Але при цьому буде збільшуватись не тільки кількість варіантів ключів, а й кількість процедур, що необхідні для шифрування/дешифрування інформації.

Довжина ключа для симетричного шифрування	Довжина ключа для симетричного шифрування
56 біт	384 біта
64 біта	512 біт
80 біт	768 біт
112 біт	1792 біта
128 біт	2304 біта

Рисунок 2.7 – Відповідність довжин ключів при однаковому рівні безпеки проти атаки методом тотального перебору ключів

Алгоритми шифрування поділяються на симетричні та асиметричні, і для забезпечення однакового рівня безпеки довжина ключа буде різною. На рисунку 2.7 зображена відповідність довжин симетричного та асиметричного шифрування.

3 ПРОЕКТУВАННЯ СИСТЕМИ

3.1 Архітектура програмної системи

При аналізі існуючих систем було виявлено, що сучасні рішення недостатньо вирішують поставлену задачу і можуть бути модернізовані. Тож було прийнято рішення створити систему, яка б давала змогу передавати різні типи сигналів, які до цього будуть зашифровуватися.

Для дизайну системи було обрано шаблон проектування MVVM, який полегшує розробку графічного інтерфейсу, а точніше його відокремлення від проектування бізнес логіки, відомої як модель. Шаблон MVVM складається з трьох частин [6]:

- модель (Model), як і в звичайному шаблоні MVC Модель представляє собою фундаментальні дані, що потрібні для роботи додатку;
- вид/(вигляд) (View) як і в звичайному шаблоні MVC, Вигляд — це графічний інтерфейс, тобто кнопки, вікно тощо;
- модель вигляду (ViewModel, що означає «Model of View») з одного боку є представленням Вигляду, а з іншого надає обгортку даних з Моделі, які мають пов'язуватись. Тобто вона являє собою Модель, яка перетворена у Вигляд, а також складається з команд, якими може скористатися Вигляд для зв'язку з Моделлю. Фактично ViewModel створена для того, щоб:
- створювати зв'язок між вікном та моделлю;
- слідкувати за змінами в даних, що проходять за допомогою користувача;
- встановлювати логіку роботи view (механізм команд).

Модель представлення являється частиною, яка створена для перетворення даних для їх підтримки і використання у подальшому. Таким чином, модель

представлення більше походить на модель, ніж на представлення і проводить обробку більшої частини логіки відображення даних (рисунок 3.1).

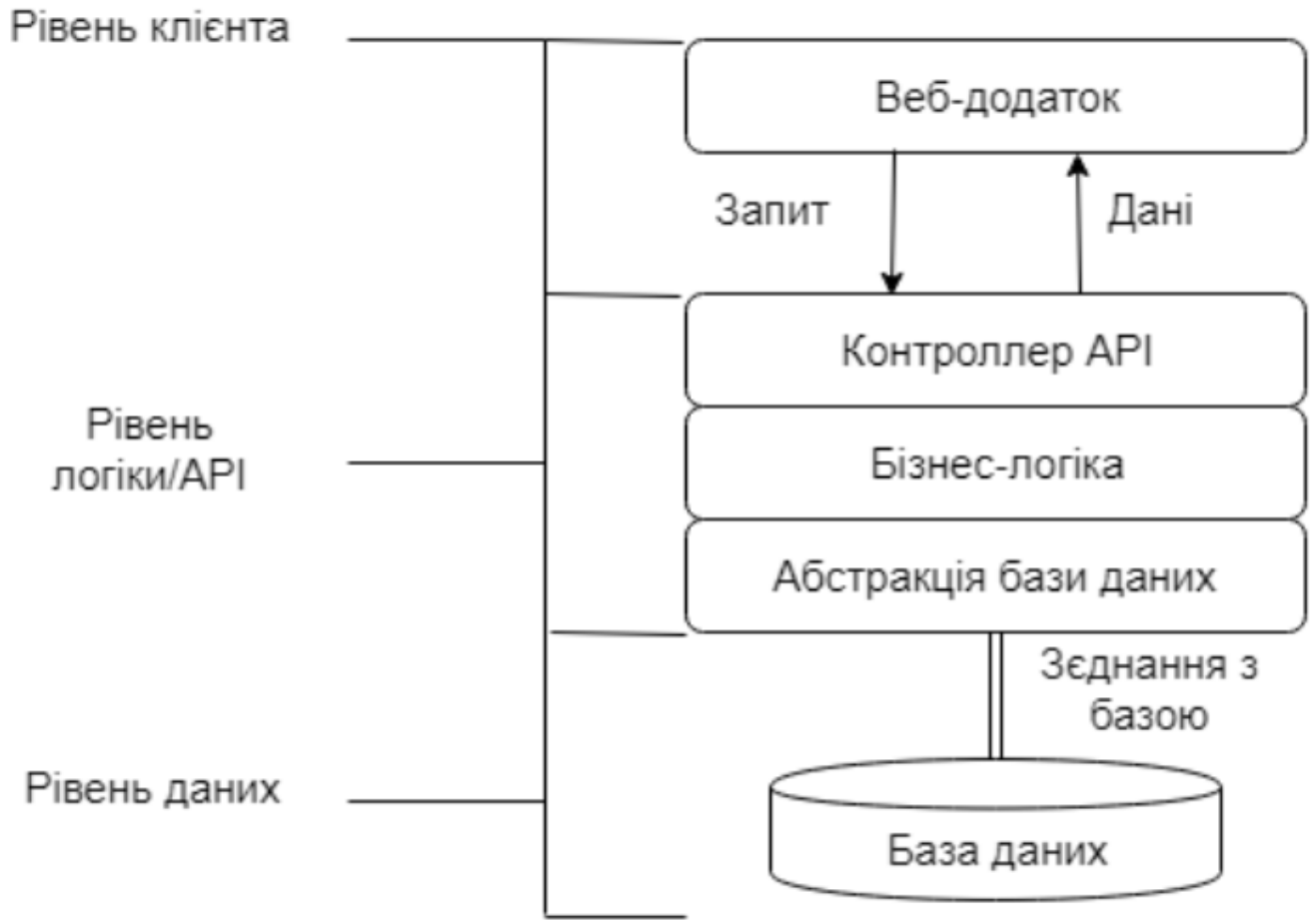

Рисунок 3.1 – Схема архітектури системи

У даній системі місце представлення займає веб-додаток, що використовуватиме бібліотеку для розпізнавання текстів TensorFlow для визначення наукового напряму тексту. Тому дана бібліотека виконує роль моделі бізнес-логіки. Тобто через веб-додаток надсилається запит до певних методів даної бібліотеки, він проходить до бібліотеки, там формується відповідь і відсилається назад до веб-додатку для того, щоб той мав змогу вивести користувачу дані, які людина потребувала.

Також на Python написаний back-end системи, що містить функції, необхідні для роботи додатку. Через це back-end виступає моделлю згідно шаблону проектування MVVM. Тобто back-end є компонентом, який зберігає і обробляє дані введені користувачем і надає дані представленню для відображення.

3.2 Структура бази даних

Всі дані будуть зберігатися з використанням СУБД SQLite, а сама БД буде створена за допомогою підходу Code First [5] і в ній будуть зберігатися три таблиці: відправники, водії та замовлення. Будемо вважати, що автором статті вважається одна людина і автор статті буде завантажувати роботу у систему самостійно.

У таблиці Користувачі будуть такі стовпці:

- id користувача;
- ПІБ користувача;
- телефон;
- електронна адреса.

У таблиці Статті будуть такі стовпці:

- id статті;
- назва;
- код напряму;
- id користувача;
- id конференції;

– файл статті.

У таблиці Конференції будуть такі стовпці:

– id конференції;
– місто конференції;
– країна конференції;
– дата конференції;
– електронна пошта конференції.

При реєстрації автор вказує свої ім’я, телефон та електронну адресу. Після успішної реєстрації його дані заносяться у базу даних та цьому запису присвоюється унікальне значення – ID користувача.

Після того, як автор роботи завантажив файл з тезами для конференції, він повинен вказати назву статті та вибрати із випадаючого списку конференцію, на яку він хочу відіслати тексти тез. Після успішного вводу цієї інформації система опрацьовує файл з тезами аби визначити код наукового напряму тексту. Після визначення ці дані заносяться у базу разом із ID користувача (автора тез), ID конференції і файлом тез. Цьому запису присвоюється унікальне значення – ID статті.

Після занесення даних до бази автор та відповідальний коференції отримують листа на електронну пошту, в якому є:

– ім’я автора;
– назва конференції;
– назва роботи;
– код напряму.

3.3 UML-моделювання системи

Моделювання розроблюваної системи проводиться з використанням мови UML для побудови діаграм, що допоможуть відобразити функціональність та внутрішню структуру системи. UML – мова графічного описання для моделювання діаграм в галузі розробки програмного забезпечення. UML це дуже універсальна мова, це відкритий стандарт, який користується графічними позначеннями для того, щоб створити абстрактну модеьі системи, що називається UML-моделлю [6].

Перед розробкою даної системи були розроблені діаграми наступних типів:

- діаграма варіантів використання;
- діаграма компонентів;
- діаграма станів;
- діаграма діяльності;
- діаграма послідовності.

Для розробки моделей використовувалось програмне забезпечення draw.io.

3.3.1 Use-case діаграма

З рисунку Г.1, де зображена діаграма варіантів використання проекту, бачимо, що актор User має такі варіанти використання:

а) авторизація та реєстрація у системі;

б) управління статтями:

1) завантаження нових тез;

2) перегляд існуючих тез;

3) перегляд тез певного автора.

в) перегляд доступних для подання конференцій.

Необхідно зауважити, що при завантаженні текстів тез користувач повинен ввести назву тез та вибрати конференцію, на яку він хоче відправити тези.

Отже незареєстрований користувач може:

- продивитися конференції, на які можливо надіслати тексти тез;
- зареєструватися;
- авторизуватися.

Причому авторизуватися користувач може лише за умови, що він зареєстрований у системі, а управляти статтями – лише за умови, що він авторизований у системі. В той же час перегляд існуючих статей неможливий без попереднього переходу до певної конференції.

Як бачимо з рисунку Г.1 додатку Г, користувач може управляти статтями, що включає в себе завантаження нових текстів тез та перегляд своїх тез.

3.3.2 Діаграма компонентів

Діаграмою компонентів вважається стала структурна діаграма, що показує дискретизацію програмної системи на компоненти структури та зв'язку (залежності) між компонентами [6]. Компоненти - це незалежні модулі програмного забезпечення. Вони приховують свою реалізацію і взаємодіють один з одним через інтерфейси.

На місці фізичних компонентів можуть знаходитися бібліотеки, файли, виконувані файли , модулі, пакети тощо. Компоненти пов'язуються за допомогою залежності, коли по'єднується певний інтерфейс одного компонента з потрібним та відповідним інтерфейсом іншого компонента. Таким чином демонструються взаємини «клієнт-джерело» між двома структурними компонентами. На рисунку Г.2 додатку Г зображено діаграму компонентів для проекту, що розроблюється.

Залежність ілюструє, що один компонент надає сервіс, необхідний іншому компоненту. Залежність відображається зв’язком від інтерфейсу або порту клієнта до інтерфейсу, що імпортується.

3.3.3 Діаграма станів

Головна мета діаграми станів (statechart diagram) – описувати можливі послідовності переходів і станів, які в результаті характеризують процес роботи системи, яку моделюють, протягом усієї її роботи (див. рис. Г.3, додаток Г) [8].

Діаграма станів ілюструє поведінку сутностей з плином часу, на основі демонстрації їхньої реакції на завершення деяких конкретних дій.

Діаграми станів частіше за все використовуються для ілюстрації поведінки окремих підсистем і систем. Також вони можуть бути застосовані для демонстрації функціональності екземплярів окремих класів, тобто для створення моделей всіх можливих змін станів у конкретних об'єктах.

3.3.4 Діаграма діяльності

Для створення моделей для процесу виконання операцій в мові UML зазвичай використовуються так звані діаграми діяльності.

На рисунку Г.4 додатку Г зображено діаграму діяльності. На ній зображено діяння, стан яких описано на діаграмі станів. Під словом «діяльність» розуміється специфікація поведінки, що виконується у вигляді послідовного і координованого виконання елементів-діяльностей, що поєднані між собою потоками, що починаються

від виходів одного вузла та йдуть до входів іншого. На діаграмі, поданій вище, детально представлено дії користувача і як система відповідатиме на них.

Графічне позначення, що використовується в них, багато в чому схоже із позначенням діаграми стану, оскільки на діаграмах діяльності також є позначення станів і переходів. Різниця полягає в семантиці станів, що використовуються для представлення не діяльності, а дії, та у відсутності підписів подій на переходах. Кожному стану на діаграмі діяльності відповідає виконання якоїсь елементарної операції, і перехід до наступного стану спрацьовує лише тоді, коли ця операція завершена в попередньому стані. Графічно діаграма діяльності представлена у вигляді графіка діяльності, вершини якого є станами дії, а дуги - переходами з одного стану дії в інший.

Таким чином, діаграми діяльності можна вважати приватним випадком діаграм стану. Вони дозволяють реалізовувати в UML особливості процедурного та синхронного управління, викликані завершенням внутрішніх дій та дій. Метамодель UML забезпечує необхідні терміни та семантику для цього. Основним напрямком використання діаграм діяльності є візуалізація особливостей реалізації операцій класів, коли необхідно представити алгоритми їх виконання. Таким чином кожен стан може бути виконанням операції якогось класу або його частини, що дозволяє використовувати діаграми діяльності для опису реакцій на внутрішні події системи.

У контексті мови UML діяльність - це набір індивідуальних обчислень, що виконуються автоматично. У цьому випадку деякі елементарні обчислення можуть призвести до певного результату чи дії (дії). Діаграма діяльності показує логіку або послідовність переходу від однієї діяльності до іншої, приділяючи увагу результату діяльності. Сам результат може змінити стан системи або повернути якесь значення.

3.3.5 Діаграма послідовності

Для моделювання дії об'єктів один з одним у мові UML використовуються такі діаграми, як діаграми взаємодії. На рисунку Г.5 додатку Г зображено життєвий цикл створення нового проекту за допомогою діаграми послідовності.

Коли говорять про ці діаграми, то мають на увазі два шляхи взаємодії. Перший - взаємодії об'єктів можна розглядати відносно плину часу, і тоді для демонстрації тимчасових особливостей передачі і прийому інформації між об'єктами використовується діаграма послідовності.

Описати поведінку користувача за допомогою цієї діаграми можна так: після того, як користувач натискає на кнопку «Створити нове замовлення» клієнтська частина відображає сторінку створення замовлення. Після цього користувач може вказати всі необхідні параметри для цього замовлення. Після цього відбудеться запит на сервер, в результаті якого будуть виведені водії з відповідними параметрами. Користувач може вибрати водія для виконання замовлення. Таким чином водій буде закріплений за замовленням і замовлення буде відіслане на сервер.

Не дивлячись на те, що розглянуті діаграми використовуються для описання динаміки поведінки систем, час, як такий, в них не присутній. Однак часовий нюанс поведінки може бути істотним при моделюванні синхронних процесів, що імітують взаємодії об'єктів. Саме для цього в мові UML використовуються діаграми послідовності.

Діаграма послідовності створена для демонстрації синхронних процесів з плином часу, які описують взаємодію об’єктів. На діаграмі демонструються лише ті об’єкти, які безпосередньо беруть участь у взаємодії. Крайнім з ліва зображається об’єкт який являється ініціатором взаємодії, з права від нього об’єкт з яким він взаємодіє.

3.4 Архітектура клієнтського додатку

Після розгляду і аналізу вимог до веб-орієнтованої клієнтської частини було прийнято рішення використовувати технологію Multiple Page Application. SPA – це веб-додаток або веб-сайт, який поміщається на декількох веб-сторінках з метою зробити роботу користувачів схожою з роботою у настільних версіях додатків. Досягається це через HTML, CSS, JavaScript, що динамічно завантажуються. Основними елементами, що використовуються для створення MPA є:

а) JavaScript, який допомагає втілити принципи MPA.
б) роутинг: навігація між представленнями(view) відбувається на клієнтській частини системи;
в) HTML5;
г) API для серверної частини, наприклад REST або RPC.

3.5 Архітектура серверної частини системи

Якщо описувати модель взаємодії усіх компонентів системи, то можна зрозуміти, що головною ланкою між клієнтським додатком та базою даних є сервер, API [16]. Але за рахунок того, що всі головні елементи системи розділені, то це дозволяє гарно масштабувати систему.

3.5.1 Структура серверної частини

Серверна частина - головний компонент системи. Система не буде повністю працездатною, якщо в ній буде відсутній хоча б один з перерахованих вище основних компонентів, однак без серверної частини система перестане бути працездатною зовсім. Як тільки буде виключений з системи сервер, сама система перестане існувати: замість неї буде кілька окремо взятих компонентів, які ніяк не взаємодіють один з одним [7].

Сервер реалізує всю бізнес логіку та зберігає дані. Він написаний згідно шаблону MVC [16].

На рівні моделі знаходяться усі класи, що описують бізнес модель, та сервісний клас, у якому представлена уся бізнес-логіка. Для виклику методів на сервері використовується WebAPI, що дозволяє отримувати та відправляти різноформатні дані по протоколу https.

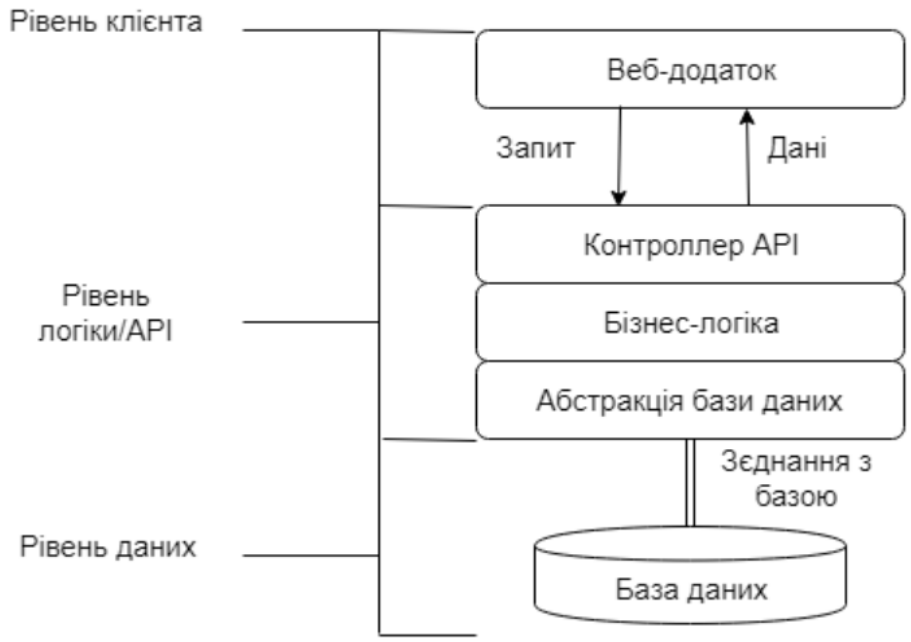

Рисунок 3.2 – Схема архітектури системи

Сервер є багаторівневим, має кілька окремих рівнів, пов'язаних один з одним: рівень загальнодоступного API, бізнес-рівень, рівень доступу до бази даних. Все вищеописане наглядно демонструє схема архітектури системи (див. рис. 3.2).

3.5.2 Взаємодія клієнта і сервера

Через те, що клієнт і сервер на Django створюють єдину систему [7], логічним буде показати як вони взаємодіють. На рисунку 3.3 представлено роботу додатку, який створений за допомогою Python фреймворку – Django.

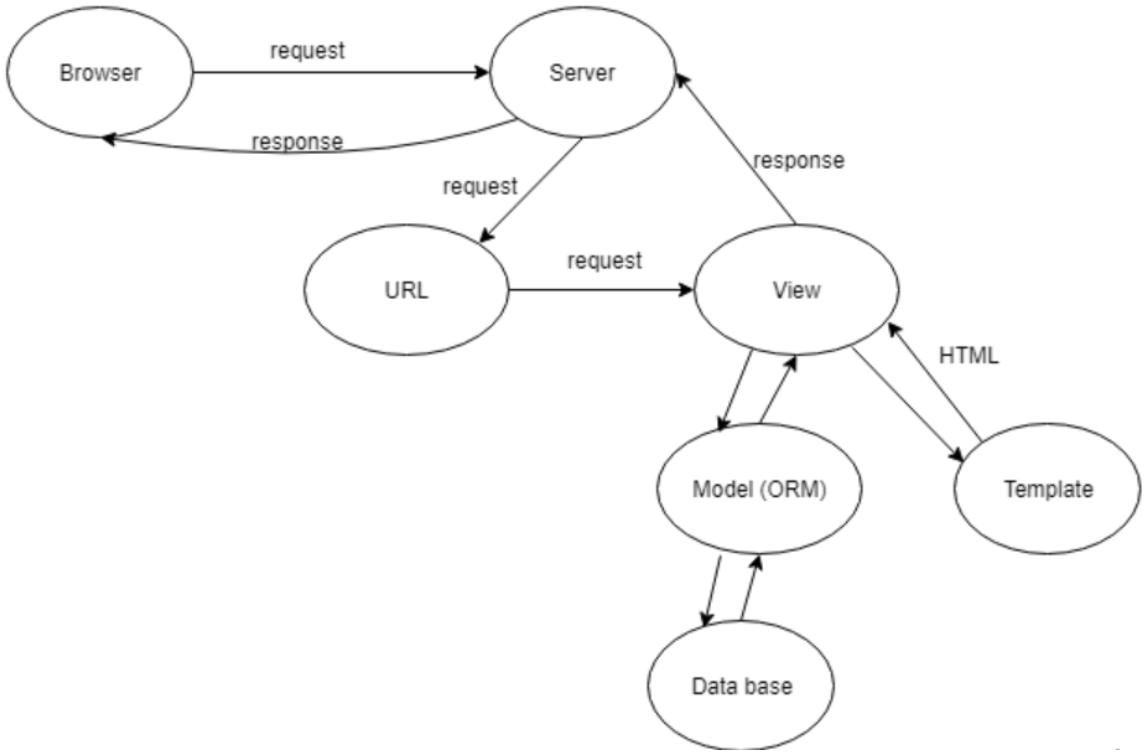

Рисунок 3.3 – Схема взаємодії клієнта та сервера Django додатка

У наступних версіях поліпшення на серверній частині будуть зводитися до поліпшення продуктивності і всебічної оптимізації.

3.5.3 Створення API

Протягом роботи над сервером буде створена API документація:

а) Users:

1) GET api/Users - отримання інформації про користувача;

2) GET api/Users /{id} - отримання користувача по унікальному ідентифікатору;

3) GET api/Users/{id}/articles – отримання списку тез певного користувача;

4) POST api/ Account/Register - реєстрація користувача;

5) POST api/Account/Token – авторизація користувача;

б) Articles:

1) GET api/ Articles: - отримання інформації про всі статті;

2) GET api/Articles/{id} – отримання тез по унікальному ідентифікатору;

3) POST api/ Articles – завантаження статті;

в) Conferences:

1) GET api/ Conferences / - отримання інформації про всі конференції;

2) GET api/Conferences/{id}: - отримання інформації про певну конференцію по унікальному ідентифікатору;

3) POST api/ Conference / - створення конференції;

4) GET api/ Conferences /{id}/articles - отримання інформації про всі статті на певній конференції;

5) GET api/ Conferences /{id}/articles /{article_id}- отримання інформації про певну статтю на конференції.

Усі данні в цьому API повертаються у спеціальному зручному форматі JSON, які передаються між рішеннями системи.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Вибір програмних засобів реалізації для back-end частини

Для того, щоб реалізувати Back-end частину було обрано мову програмування Python та фреймворк Django. Мова Python була обрана через те, що вона з однієї сторони дуже проста у використанні, особливо підходить для задачі розпізнавання тексту, бо має багато реалізованих бібліотек. С іншої сторони при використанні нейронних мереж нам треба проводити дуже багато експериментів – нам треба визначити архітектуру нейронної мережі, яка буде підходити для задач, що вирішуються, оцінити параметри навчання та багато іншого. І Python дуже добре підходить для проведення експериментів.

Він дозволяє створювати систему, що працює і надає необхідний мінімум функціоналу:

– моделі: місток між Python класами та базою даних, який робить усю складну роботу щодо збереження та пошуку даних для вас [17];

– шаблони: генерація HTML коду без використання Python мови, натомість із мінімальними вкрапленнями логіки з допомогою динамічних Django тегів напряму у HTML код; це дає можливість нам мати так званий MVC підхід та розділяти логіку від даних та представлення;

– диспетчер URL: інструмент, з допомогою якого можемо легко будувати ієрархію URL адрес нашого веб-сайту;

– адміністративна частина: іде разом із Django та дозволяє керувати даними веб-сайту та його налаштуваннями без додаткової розробки;

– користувачі: якщо ваша веб-аплікація потребує логованих користувачів, тоді нічого додаткового практично не доведеться розробляти; Django дає весь функціонал, що дозволить вам мати дані користувачів, форми логування, реєстрації та усі налаштування з безпеки, ролі та дозволи для користувачів;

– форми: додаткові Python класи для роботи із веб-формами; вони економлять масу часу при розробці стандартних форм;

– розробницькі інтерфейси (Development APIs): найпоширенішим є REST; з Django можна легко навчити вашу веб-аплікацію “говорити” у форматі REST, який часто буває корисним для мобільних додатків та комунікації із іншими сервісами;

– міграція: такий інструмент як South дозволяє легко змінювати та мігрувати ваші дані в базі, їхній формат та вигляд, а також робити це повторно без будь-яких проблем;

– як і у будь-якому іншому веб-фреймворку такі речі як деплоймент на кінцевий сервер, документація, автоматичні тести, атомарна розробка (модульність), розробницькі інструменти, є добре продумані у фреймворку Django; вони допоможуть вам швидше та з легкістю закінчувати ваші веб-проекти.

Крім того, що можна користуватись усіма вбудованими функціями фреймворку, також існує маса бібліотек та аплікацій.

Ну і ще один не менш важливий аргумент за Django – це OpenSouce фреймворк. Його код можна вільно використовувати, читати та змінювати.

Середа розробки – PyCharm – професійна IDE від JetBrains, що надається студентам під час навчання. Обрана вона була через орієнтацію на Python.

Також важливо відмітити, що для роботи саме з нейронною мережею була обрана бібліотека Keras. Це бібліотека глибинного навчання, яка використовує бібліотеку TensorFlow для виконання ефективних обчислень. Особливість Keras у тому, що вона дозволяє описувати на Python нейронну мережу. За допомогою цієї бібліотеки можна вказати з яких шарів буде складатися нейронна мережа, які там використовуються функції активації, який використовується метод оптимізації для зменшення помилок і інші параметри, важливі для навчання нейронної мережі. Бібліотека Keras використовує бібліотеку TensorFlow для проведення високоефективних обчислень.

4.2 Вибір програмних засобів реалізації для front-end частини

Для Front-end частини системи було обрано такі технології, як HTML і CSS. HTML (від англ. HyperText Markup Language – «мова розмітки гіпертексту») - стандартна мова розмітки документів у Всесвітній павутині. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Мова HTML інтерпретується браузерами і відображається у вигляді документа в зручній для людини формі. HTML – це невід'ємна складова і основа практично будь-який веб-сторінки. Мова HTML, в першу чергу, виступає як засіб логічної розмітки сторінки. Саме HTML дозволяє нам наділяти вміст сторінки певним змістом, а реалізується це за допомогою так званих тегів.

CSS (англ. Cascading Style Sheets – каскадні таблиці стилів) – формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки. Переважно використовується як засіб опису, оформлення зовнішнього вигляду веб-сторінок, написаних за допомогою мов розмітки HTML і XHTML, але може також застосовуватися до будь-яких XML-документах, наприклад, до SVG або XUL.

Простіше кажучи, мова CSS призначена для того, щоб надавати необхідний зовнішній вигляд HTML-документами.

Отже, метою створення CSS було відділення опису логічної структури веб-сторінки від її зовнішнього вигляду. Як було сказано вище, для опису структури використовується HTML, для опису ж того, як ця логічна структура буде виглядати, відповідає як раз CSS.

Роздільний опис логічної структури та подання документа дозволяє більш гнучко управляти зовнішнім виглядом документа і мінімізувати обсяг повторюваного коду, який би неминуче виникав при використанні HTML для опису зовнішнього вигляду документа.

4.3 Опис структури проекту

Як було згадано раніше, PyCharm – інтегроване середовище розробки для мови програмування Python. Воно надає засоби для аналізу коду, інструмент для запуску юніт-тестів і підтримує веб-розробку на Django.

Створений проект складається з трьох папок, бази даних та системного файлу manage.py.

Перша папка називається backend і представляє собою Django додаток. Вона містить в собі ще дві папки – migrations і static.

Директорія migrations містить дані про міграції – керування інкрементними [18], зворотніми змінами реляційної бази даних . Міграція схеми виконується над базою даних коли необхідно оновити, або повернути схему бази даних до якоїсь новішої чи старішої версії.

Міграції виконуються програмно за допомогою інструменту міграцій (англ. schema migration tool). При виклику інструменту міграції з вказуванням бажаної версії схеми, інструмент автоматично застосовує або відкочує міграції в правильній послідовності аж поки не приведе базу даних до бажаного стану.

Директорія static створена для зберігання статичних файлів, таких як файли стилів css, зображень та файлів сценаріїв javascript. Також у static включений модуль sass – скриптова метамова, яка інтерпретується в каскадні таблиці стилів. Sass призначений для підвищення рівня абстракції коду та спрощення файлів CSS. Мова Sass має два синтаксиси:

− sass (оригінальний) — відрізняється відсутністю фігурних дужок, в ньому вкладені елементи реалізовані за допомогою відступів, а правила відокремлюються переведенням рядка;

− scss (новий) — використовує фігурні дужки (подібно до CSS).

Файли sass-синтаксису мають розширення .sass, scss-синтаксису — .scss.

Sass розширює CSS, надаючи кілька механізмів, доступних в більш традиційних мовах програмування, зокрема об'єктно-орієнтованих мовах, але недоступних для CSS. Інтерпретатор Sass транслює SassScript у блоки правил CSS. По суті, Sass — це синтаксичний цукор для CSS.

Крім папок migrations та static у директорії backend присутні такі файли:

– forms.py – файл, у якому ініціалізуються поля форм, що присутні у веб-додатку;

– models.py – файл, у якому прописано поля моделей;

– views.py – файл, що представляє функції, які потрібні для відображення веб-сторінок у веб-додатку та передачі даних з сервера на веб-сторінку.

Друга папка bla-bla-par є головною папкою проекту та містить основні налаштування серверу, що знаходяться у файлі settings.py, та інші системні файли, що потрібні для належної роботи проекту.

Третя директорія, що знаходиться у структурі проекту, називається templates та містить html файли, з яких складається веб-додаток проекту.

4.4 Векторизація та токенізація тексту

Вхідними даними для нейронної мережі завжди являються числа. Але дуже часто нам треба обробити нечисельні дані. Тому нам потрібно перевести початкові дані в набір чисел [9]. З зображеннями все просто – зображення у цифровому форматі вже представляють собою набір чисел, які відповідають інтенсивності пікселів в діапазоні від 0 до 255.

З текстом все трохи складніше – перетворення тексту у набір чисел, який потім можна аналізувати, називається векторизацією. Але для того, щоб аналізувати текст, нам потрібно розбити його на частини.

Токенізація – розбиття тексту на частини таким чином, що кожна частина буде представлятися в цифровому форматі окремо. Текст можна розбити на символи (букви, цифри, знаки пунктуації і т. д.) і в числовому виді представляти окремо кожен символ. Також модливо розділяти текст на слова і ставити в відповідність число або набір чисел. І нарешті зараз існує підхід, коли текст розподіляється на речення та в виді чисел представляється ціле речення.

Після того, як ми розбили весь текст на токени, нам потрібно їх перетворити на числа. Цей процес і називається векторизація. Існує декілька видів:

- числове кодування;
- one hot encoding;
- щільні векторні представлення (embedding).

Перший метод – числове кодування – кожному токену треба назначити свій код. Інший підхід – це коли кожному токену ставиться у відповідність не одне число, а цілий вектор, який містить декілька чисел. Найпростіший варіант – використовувати підхід one hot encoding. У цьому випадку вектор містить стільки чисел скільки можливо використати токенів та всі значення вектору дорівнюють нулю, окрім того, який відповідає токену. І третій варіант називається щільні векторні представлення або embedding. В цьому випадку токену у відповідність ставиться також не число, а вектор, але розмір вектора менше, чим у one hot encoding і в цьому векторі можуть використовуватися не тільки нулі та одиниці, але й інші цифри.

5 ПРОВЕДЕННЯ ЕКСПЕРИМЕНТІВ

Отож, проведемо експерименти аби виконати мету дослідження – встановити найефективніший метод для класифікації даних для системи, що проектується. Для проведення експериментів будем обрати вбудовану базу текстів imdb.

5.1 Числове кодування

Для початку імпортуємо всі необхідні бібліотеки (див. рис. Д. 1, додаток Д). Sequential використовується для представлення нейронної мережі, у якій шари йдуть один за одним послідовно [12].

Через те, що в роботы використовується повнозв'язна нейронна мережа, ми імпортуємо модуль Dense (у Keras шари повнозв’язної нейронної мережі називаються Dense). Також підключаємо утиліти Keras, які нам будуть потрібні для того, щоб привести дані для навчання в відповідний для Keras формат.

Далі завантажуємо дані для навчання. Це можна зробити самим, а можна скористатися вбудованими функціями, що і було зроблено (див. рис. Д. 2, додаток Д).

Далі при інспектуванні даних бачимо, що у наборі даних IMDB використовується частотне кодування слів [13]. Завантажимо словник, який використовувався для кодування (див. рис. Д. 3, додаток Д).

Тепер нам треба перетворити словник, аби за номером отримувати слово (див. рис. Д. 4, додаток Д). Далі переходимо до підготовки даних до навчання (див. рис. Д. 5, додаток Д). Нарешті можна створити нейронну мережу (див. рис. Д. 6, додаток Д) та навчати її (див. рис. Д. 7, додаток Д). На рисунку Д. 8 додатку Д можемо бачити

результати навчання, а на рисунку Д. 9 додатку Д – долі правильних відповідей нейронної мережі.

І нарешті за допомогою вбудованих функцій можна вирахувати, яка загальна доля правильних відповідей досяжна на тестовому наборі даних [14], що показано на рисунку Д. 10 додатку Д.

Отже, доля вірних відповідей для методу числового кодування – 50,888%.

5.2 One hot encoding

Для підготовки даних для навчання у цьому методі потрібна функція для кодування, що показана на рисунку Д. 11 додатку Д, а також на рисунку Д. 12 додатку Д можемо бачити використання даної функції.

Функція навчання нейронної мережі не сильно відрізняється від попереднього методу (див. рис. Д. 13, додаток Д). Також можемо побачити результати навчання нейронної мережі на рисунку Д. 14 додатку Д та долі правильних відповідей під час навчання (див. рис. Д. 15, додаток Д).

Після перевірки цього методу на тестовому наборі даних результат показав 86,555 % правильних відповідей від нейронної мережі.

5.3 Щільні векторні представлення (embedding)

Для перевірки роботи третього типу векторизації також створимо нейронну мережу (див. рис. Д. 16, додаток Д).

Для навчання нейронної мережі після методу embedding використаємо функцію [15] (див. рис. Д. 17, додаток Д) та подивимося на результат на рисунку Д. 18 додатку Д. На рисунку Д. 19 додатку Д можемо бачити долю правильних відповідей під час навчання.

Після перевірки цього методу на тестовому наборі даних результат показав 87,333 % правильних відповідей від нейронної мережі.

5.4 Висновки з експериментів

Отже, за результатами експериментів найбільшу результативну долю правильних відповідей показав метод embedding, що означає, що він являється найбільш ефективним при вирішенні задач потрібного для дослідження типу.

ВИСНОВКИ

В рамках написання кваліфікаційної роботи було спроектовано систему шифрування інформації за допомогою існуючих методів та бібліотек для класифікації даних та їх вдосконалення. Також було спроектовано клієнтський додаток.

Результатом проходження передатестаційної практики є повноцінна система, що складається з клієнту та серверу, яка дозволить завантажувати тексти тез та мати можливість автоматично визначати науковий напрям статті.

Для реалізації всього необхідного функціоналу була проаналізована предметна область, виявлені взаємовідносини між основними об'єктами системи. Для зберігання інформації була спроектована і нормалізована база даних, яка відображає всі зв'язки між сутностями даної предметної області. Проаналізувавши спроектовану систему на відповідність потребам, які необхідні для зберігання інформації про статтю, прийшли до висновку, що система повністю задовольняє поставленим завданням.

Дана програмна система є дуже універсальною у використанні, має зрозумілий інтерфейс, що дозволяє працювати з продуктом користувачам з різними рівнями комп'ютерної грамотності. Також вона має середні вимоги до програмного та апаратного забезпечення, надає можливість досить високої швидкості роботи, безпечна у використанні, що є безумовними перевагами цього додатку.

Як уже згадувалося дана система дозволяє визначати науковий напрям текстів тез, що були завантажені в систему, використовуючи методи класифікації різних типів даних зі спеціальних бібліотек. Зараз на ринку немає аналогів цього програмного продукту.

Цей продукт виділяється через свою унікальну пропозицію – можливість автоматично визначати напрям роботи та сортувати тексти по категоріях для зручності організаторів конференцій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rossiter J. R., Percy L. Advertising Communications and Promotion Management. — Wiley: McGraw-Hill, 2004 – 233 с.

2. Назаров О.С. Теорія прогнозування: навч. посіб. – Харків: ХНУРЕ, 2017. – 300 с.

3. «Understanding RNN and LSTM»/ США - URL: https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e (дата звернення: 27.02.2021)

4. Нишант Ш. Машинное обучение и TensorFlow. – Питер, 2019. – 336 с.

5. Nicholas Petreley, Alex Khaloghli, Scott Mace Ashton-Tate/Microsoft SQL Server: PC databases enter a new era // InfoWorld : журнал. – США: InfoWorld Media Group, Inc., 2007. – Т. 11, № 41. — С. 55-75.

6. Фаулер, М. UML. Основы : пер. с англ. А.: Петухов/ М. Фаулер, К. Скотт. - СПб.: Символ, 2006. - 184 с.

7. В. Подоба. Веб-розробка з Python та Django для початківців – Рівне : ВНТУ, 2013 - 258 с.

8. 4ДСТУ 3008-95. Видання. Документація. Київ, 2010. 16 с. (Звіти у сфері науки і техніки)

9. Marcos Lopez de Prado. Advances in Financial Machine Learning. – Wiley, 2018. – 393 с.

10. «Ассиметричне шифрування»/ Україна – URL: https://indeed-id.ru/blog/asimmetrichnoe-shifrovanie (дата звернення 20.04.2021)

11. «Электронная подпись»/ Росія – URL: https://ru.wikipedia.org/wiki/Электронная_подпись (дата звернення: 19.04.2021)

12. «Методы распознавания текстов»/ Росія – URL: https://habr.com/ru/post/112442/ (дата звернення 29.04.2021)

13. «Глубокие нейросети: руководство для начинающих»/ Росія – URL: https://medium.com/nuances-of-programming/глубокие-нейросети-руководство-для-начинающих-df779507f424 (дата звернення 30.04.2021)

14. Chollet F. Deep Learning with Python. Detroit: Manning Publications, 2014. – 467 с.

15. Gorbenko, I., Kachko, O., Akolzina. Research of post-quantum public key encryption algorithm. // 2018 International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2018 - Proceeding, 2018, 9047585

16. Назаров О. С., Шураєв І. Д. Автоматизація процесу створення 3D моделей на основі зображень. // Міжнародна наукова інтернет-конференція "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення" / Збірник тез доповідей: випуск 34, 2018. – С. 54-56.

17. Назаров О. С., Гавва О. С. Інтелектуальне керування системою управління ходовою частиною. // Інформаційні технології і мехатроніка: освіта, наука та працевлаштування: матеріали Міжнародній науково-практичній конференції. — Х.:ХНАДУ, 2016. — С. 88—90.