НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
(повна назва інституту/факультету)

Кафедра автоматики та управління в технічних системах
(повна назва кафедри)

«На правах рукопису»
УДК _____

«До захисту допущено»

Завідувач кафедри
_____ Ролік О. І.
(підпис) (ініціали, прізвище)

“__”_____2019 р.

Магістерська дисертація

зі спеціальності (спеціалізації) _121 Інженерія програмного забезпечення_____
(код і назва спеціальності)

_____

на тему: Інтелектуальна система фільтрації коментарів з використанням_____
машинного навчання_____

Виконав: студент _6_ курсу, групи __ІТ-83мп__
(шифр групи)

Писаренко Олег Анатолійович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник професор, Дорошенко А. Ю. _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

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

Київ – 2019 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет (інститут) Інформатики та обчислювальної техніки
(повна назва)

Кафедра Автоматики та управління в технічних системах
(повна назва)

Освітньо-кваліфікаційний ступінь магістр
(назва ОКР)

Спеціальність 121 Інженерія програмного забезпечення
(код і назва)

Спеціалізація 121 Інженерія програмного забезпечення комп'ютерних систем
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
О.І. Ролік
(підпис) (ініціали, прізвище)
« » 2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Писаренку Олегу Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації Інтелектуальна система фільтрації коментарів з використанням машинного навчання
Науковий керівник дисертації Дорошенко А. Ю. д.ф.-м.н., проф.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «____» _____ 2019 р. № _____

2. Строк подання студентом дисертації _____

3. Об'єкт дослідження інтелектуальні алгоритми аналізу текстових коментарів

4. Зміст пояснювальної записки а) аналіз існуючих рішень; б) розробка моделі машинного навчання; в) розробка системи; г) розробка інтерфейсу користувача д) оцінка якості системи; е) розробка стартап-проекту.

5. Перелік завдань, які потрібно розробити: проаналізувати предметну область, існуючі рішення для побудови систем фільтрації коментарів, розробити моделі та обрати алгоритми, розробити архітектуру системи, провести оцінку роботи

системи, протестувати розроблену систему, провести маркетинговий аналіз стартап проекту.

6. Перелік графічного (ілюстративного) матеріалу: структурна схема моделі для аналізу клієнтських уподобань, варіації моделі, діаграма варіантів використання, структурна схема, функціональна схема, схема алгоритмів роботи, діаграма послідовності, діграма розгортання.

7. Орієнтовний перелік публікацій: _____

_____

7. Консультанти розділів дисертації:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

8. Дата видачі завдання 05 вересня 2019

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів дисертації	Примітка
1	Аналіз предметної області, огляд літератури	15.09.2019	
2	Дослідження методів аналізу текстових матеріалів	18.09.2019	
3	Розробка моделі машинного навчання для аналізу коментарів	30.09.2019	
4	Технічне рішення та реалізація системи фільтрації коментарів	20.10.2019	
5	Тестування та аналіз роботи системи	10.11.2019	
6	Розробка стартап-проекту	20.11.2019	
7	Оформлення матеріалів дисертації	03.12.2019	

Студент _____ О. А. Писаренко
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ А. Ю. Дорошенко
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Магістерська дисертація 116 с., 30 рис., 43 табл., 9 додатків, 23 джерел.

Тема магістерської дисертації “Інтелектуальна система фільтрації коментарів з використанням машинного навчання”.

Актуальність магістерської дисертації полягає в тому, що алгоритми аналізу тексту мають широке застосування у наш час, як наприклад алгоритми аналізу текстових коментарів. Дані алгоритми дозволяють оцінити емоційне забарвлення коментарів на інтернет-платформах та відео-сервісах та відфільтрувати ті коментарі, які можуть нанести шкоду певній соціальній групі, і як наслідок, покращити якість змісту чатів та вивести обговорення на конструктивний рівень.

Об’єктом дослідження є інтелектуальні алгоритми аналізу текстових коментарів.

Предметом дослідження є точність методів машинного навчання для аналізу текстових коментарів.

Метою цієї роботи є створення інтелектуальної системи фільтрації коментарів з використанням машинного навчання, яка зможе зменшити вплив агресивних та шкідливих коментарів на спілкування інтернет-спільноти. Задачею роботи є:

1. дослідити предметну область;

2. дослідження методів аналізу текстових матеріалів;

3. розробка моделі машинного навчання для аналізу коментарів;

4. розробити систему фільтрації коментарів з використанням машинного навчання у вигляді веб-сервісу;

5. дослідити ефективність розробленого методу у порівнянні з існуючими.

Ключові слова: нейронні мережі, система фільтрації коментарів, машинне навчання, аналіз текстових даних.

ABSTRACT

Master's Thesis 116 p., 30 figures, 43 tables, 9 appendixes, 23 sources.

Theme of the master's thesis "Intelligent system of comment filtering using machine learning".

The reason for the relevance of a master's thesis is that text analysis algorithms are widely used in our time, such as text comment analysis algorithms. These algorithms allow you to evaluate the emotional coloring of comments on online platforms and video services, and to filter out those comments that can hurt a particular social group and, as a consequence, improve the quality of the content of the chats and bring the discussion to a constructive level.

The object of the study is intelligent algorithms for analyzing text comments.

The subject of the study is the accuracy of machine learning methods for analyzing text comments.

The purpose of this work is to create an intelligent system of comment filtering using machine learning that can reduce the impact of aggressive and harmful comments on the online community. The task of the work is:

1. explore the subject area;

2. research of methods of analysis of text materials;

3. developing a machine learning model for comment analysis;

4. develop a comment filtering system using machine learning as a web service;

5. to investigate the effectiveness of the developed method in comparison with the existing ones.

KEYWORDS: NEURAL NETWORKS, COMMENT FILTERING SYSTEM, MACHINE LEARNING, TEXT DATA ANALYSIS.

ЗМІСТ

ВСТУП

В час швидкого розвитку інтернету, соціальних мереж та стрімінгових сервісів виникла необхідність в реалізації інструментів та платформ для обговорення та спілкування між людьми. Не завжди таке спілкування є конструктивним та по суті, а тим більше, коли спільнота стає досить великою або популярною, там з'являються люди із зовсім інакшою метою – не конструктивна критика авторів, агресивна та нецензурна дискусія з іншими коментаторами, приниження та образи певних груп людей за різними ознаками.

Зовсім нещодавно навіть виникли нові мовні терміни як «булінг» та «харасмент», які також можна віднести до спілкування в мережі Інтернет.

Таким чином виникла ідея в створенні програмного інструменту, завдяки якому автори та модератори різних інтернет-майданчиків зможуть автоматизувати стеження за своєю спільнотою коментаторів та завчасно фільтрувати коментарі тих людей, які можуть образити та негативно вплинути на певну групу спільноти, тим самим зменшити можливість виникнення напруження та незадоволеності спільноти, яка приходить на ту чи іншу платформу за адекватним спілкуванням з іншими людьми.

Метою магістерської дисертації є створення інтелектуальної системи фільтрації коментарів з використанням машинного навчання, яка зможе зменшити вплив агресивних та шкідливих коментарів на спілкування інтернет-спільноти.

Об'єктом даної роботи є інтелектуальні алгоритми аналізу текстових коментарів.

Предметом даної роботи є алгоритми машинного навчання для навчання з вчителем.

1 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

1.1 Perspective

Perspective – спільний проект компаній Jigsaw та Google, який дає доступ до відкритого API, що використовує машинне навчання для для покращення онлайн спілкування [1]. Perspective API оцінює коментар на основі потенційного впливу на розмову. На рисунку 1.1 зображено веб-інтерфейс демонстрації роботи Perspective API.

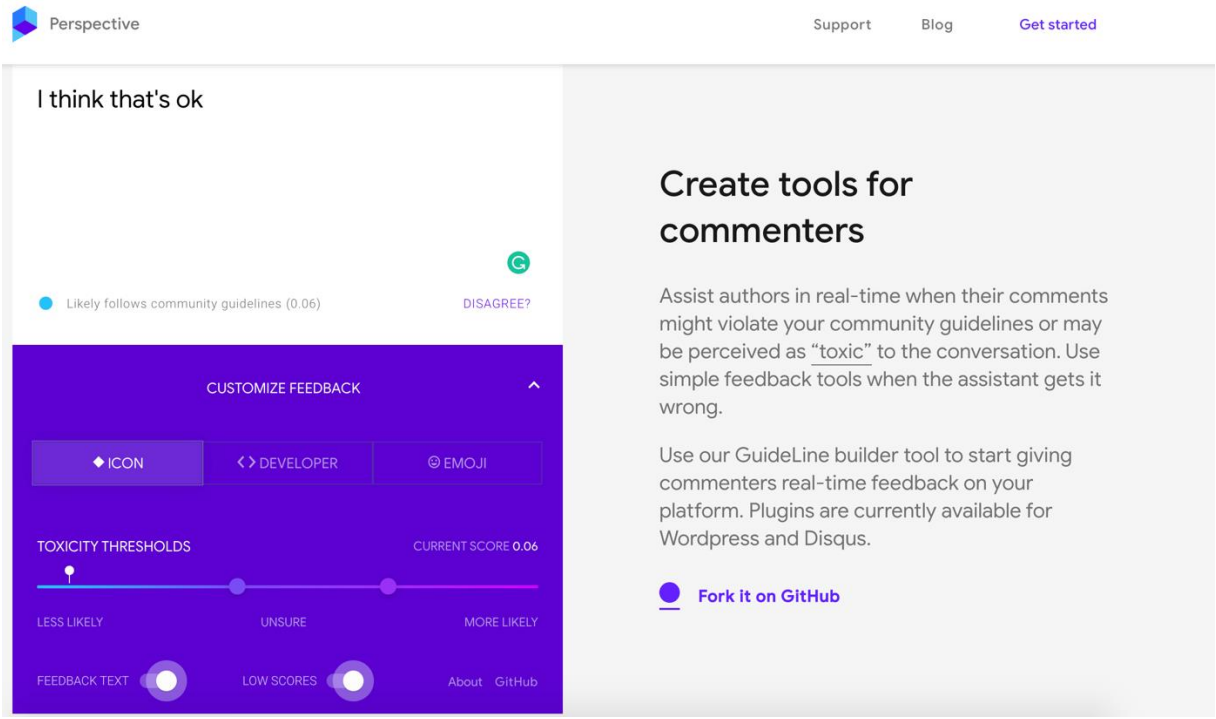

Рисунок 1.1 – Веб-інтерфейс демонстрації роботи Perspective API.

Модель машинного навчання Perspective API навчена на оцінювання коментарів, ґрунтуючись на його схожістю з коментарями, за якими люди сказали, що вони є "токсичними", або можуть змусити їх залишити розмову.

Існує також кілька інших експериментальних моделей Perspective API, які можна випробувати, наприклад, модель, яка намагається виявити несуттєві коментарі.

До переваг Perspective API можна віднести те, що їх модель машинного навчання навчена на набагато більшій кількості даних, що дозволяє їм урізноманітнить вибірку

та отримати більш точні результати в специфічних розмовних тематиках. Також, система є відкритою та безкоштовною.

Недоліком Perspective API є те, що це все ж таки API для розробників, а не закінчена система, яку можна легко підключити до якоїсь платформи коментування.

1.2 Tisane

Tisane API – проект компанії Tisane Labs, для аналізу тексту на різних мовах та надання інформації, такої як:

- тон тексту;
- визначення агресивності;
- тема тексту;
- список речень та метадані про кожне речення.

Веб-інтерфейс демо версії Tisane API наведено на рисунку 1.2.

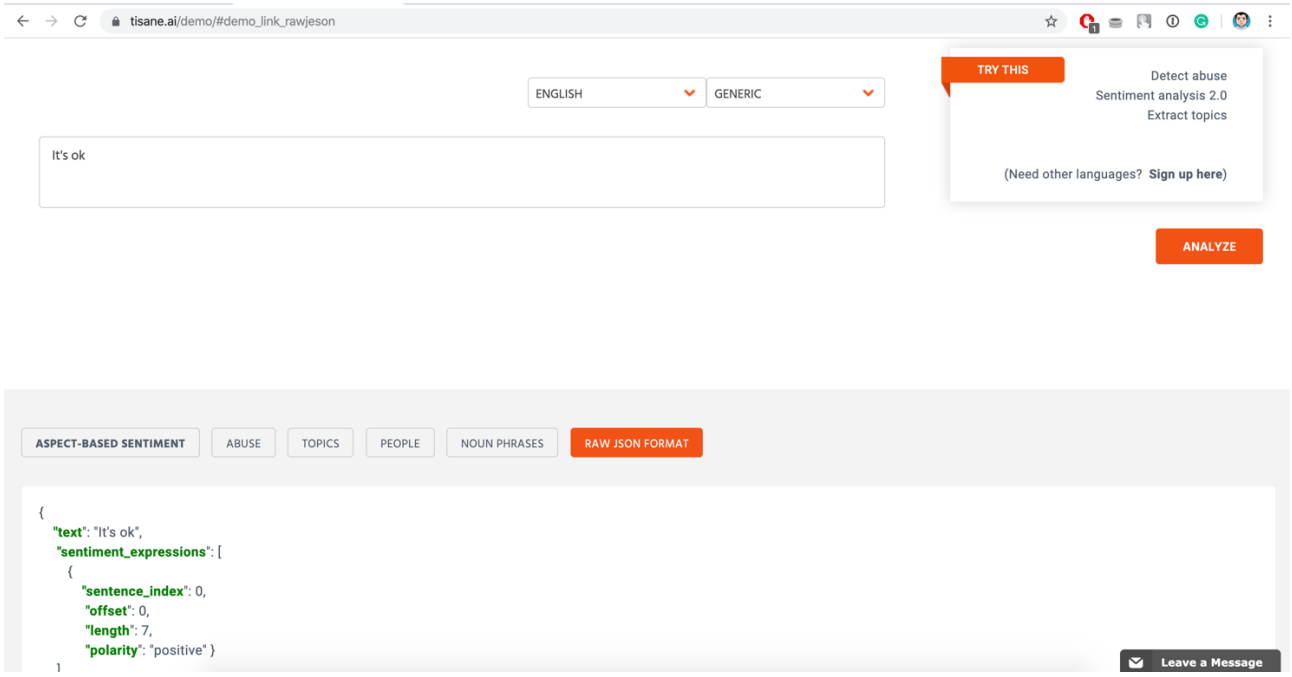

Рисунок 1.2 – Веб-інтерфейс демонстрації роботи Tisane API.

Так як і Perspective API, Tisane API для аналізу тексту використовують алгоритми машинного навчання.

Перевагою Tisane API є дуже детальний аналіз тексту та, як зазначають розробники на своєму сайті, їх проект є більш точним за Perspective API [2].

Але Tisane API також пропонується для використання лише у вигляді API, без інтеграції з популярними веб-сервісами. Через більш глибокий аналіз вхідного тексту час відповіді роботи Tisane API займає досить багато часу. Tisane API є комерційним проектом, а тому не надає свою реалізацію у вигляді відкритого коду. Використовувати сервіс можна безкоштовно, з обмеженням в 50 тисяч записів всього та не частіше 10 запитів в хвилину. Більше розширені плани пропонуються за місячною платної підпискою. Мінімальний тариф починається з 99$ за 100 тисяч запитів та обмеженням в 120 запитів в хвилину.

1.3 Існуючі методи машинного навчання

Машинне навчання та аналіз даних знаходяться на стику галузей математики, програмування і математичної статистики. У загальному випадку, машинне навчання являє собою пошук залежностей між даними і використання знайденої залежності для вирішення поставленого спочатку завдання.

Формально, машинне навчання - математична дисципліна, що виділяє знання з даних, що використовує розділи математичної статистики, чисельних методів оптимізації, теорії ймовірності та дискретного аналізу [3].

Будь-яке завдання в машинному навчанні формально записується як:

X - об'єкти (безліч), Y - відповіді (безліч), $y: X \to Y$ - деяка залежність (цільова функція).

Дано: $\{x_i, \ldots, x_l\} \subset X$ - навчальна вибірка,

$y_i = y(x_i), i = 1, \ldots, l$ - відповіді (відомі).

Потрібно знайти: функцію рішення (decision function) s, що наближає y на безлічі X.

У свою чергу, машинне навчання поділяється на два підтипи, які розрізняються способом, як саме модель буде навчена:

1. Навчання з учителем.

Цей випадок є найбільш поширеним. Необхідно знайти функціональну залежність між об'єктами і відповідями.

2. Навчання без вчителя.

Потрібно шукати залежності між об'єктами, так як відповіді не задаються. Прикладом може служити завдання кластеризації, коли у нас є тільки безліч об'єктів і їх потрібно розділити на кілька класів.

1.3.1 Логістична регресія

Логістична регресія зазвичай використовується для оцінки ймовірності того, що екземпляр належить до того чи іншого класу. Якщо прогнозована ймовірність більше 50%, то модель прогнозує, що екземпляр належить до цього класу (званий позитивним класом, представленим значенням «1»), або ж вона передбачає, що це не так (тобто він належить до негативного класу зі значенням «0»). Це робить її бінарним класифікатором. [4].

Модель логістичної регресії обраховує зважену суму вхідних змінних (плюс зміщення), але замість безпосереднього виводу результату, як це робить модель лінійної регресії, вона виводить ймовірність настання результату події.

Отримання ймовірності про настання результату події $y = 1$ дорівнює:

$$P\{y = 1 | x\} = f(z),$$

де, $z = \theta^T x = \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n, x \; i \; \theta$ - вектора-стовпці значень незалежних змінних $x_1, x_2, ..., x_n$ та параметрів або ще коефіцієнтів регресії - дійсних чисел $\theta_1, \theta_2, ..., \theta_n$, відповідно, а $f(z)$— так звана логістична функція, іноді також названа як сигмоїда:

$$f(z) = \frac{1}{1 + e^{-z}}$$

Графічне зображення логістичної функції зображене на рисунку 1.3.

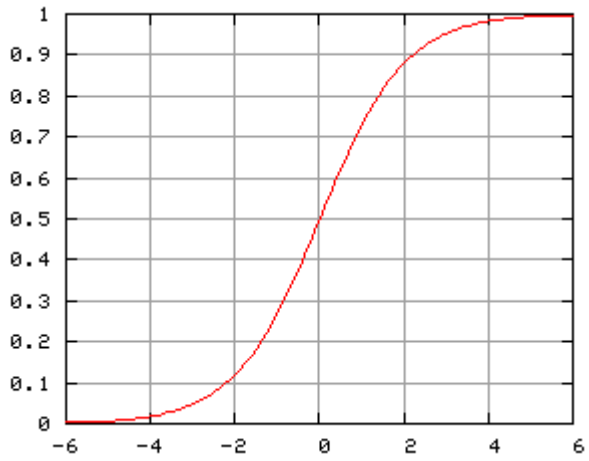

Рисунок 1.3 – Графік логістичної функції

Так як y може приймати значення лише 0 або 1, то ймовірність отримання другого значення:

$$P\{y=0 \mid x\}=1-f(z)=1-f\left(\theta^T x\right).$$

Для стислості функцію розподілу y при заданому x можна записати в такому вигляді:

$$P\{y \mid x\}=f\left(\theta^T x\right)^y\left(1-f\left(\theta^T x\right)\right)^{1-y}, y \in \{0,1\}$$

Фактично, це є розподіл Бернуллі з параметром, що дорівнює $f\left(\theta^T x\right)$.

Для підбору параметрів $\theta_1, \theta_2, ..., \theta_n$ потрібно створити навчальну вибірку, що складається з таких елементів як значення незалежних змінних та відповідних до них значень залежних змінних. Іншими словами, це нескінченність пар $\left(x^{(1)}, y^{(1)}\right), ..., \left(x^{(m)}, y^{(m)}\right)$, де $x^{(i)} \in \Re^n$ - вектор значень незалежних змінних, а $y^{(i)} \in \{0,1\}$ - відповідне їм значення. Така кожна створена пара змінних є прикладом для навчання.

Зазвичай використовується метод максимальної правдоподібності, згідно з яким вибираються параметри θ, що максимізує значення функції правдоподібності на навчальній вибірці:

$$\theta = \arg\max_\theta L(\theta) = \arg\max_\theta \prod_{i=1}^{m} P\left\{y = y^{(i)} \mid x = x^{(i)}\right\}$$

Максимізація функції правдоподібності відповідає максимізації її логарифма:

$$\log L(\theta) = \sum_{i=1}^{m} \log P\left\{y = y^{(i)} \mid x = x^{(i)}\right\} = \sum_{i=1}^{m} y^{(i)} \log f\left(\theta^T x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - f\left(\theta^T x^{(i)}\right)\right)$$

Наприклад, метод градієнтного спуску може бути використаний для максимізації цієї функції. Суть цього метода полягає у виконанні послідовних ітерацій, починаючи з деякого початкового значення параметра θ:

$$\theta := \theta + \alpha \nabla \log L(\theta) = \theta + \alpha \sum_{i=1}^{m} \left(y^{(i)} - f\left(\theta^T x^{(i)}\right)\right) x^{(i)}, \alpha > 0$$

Зазвичай для цього використовується метод Ньютона і стохастичний градієнтний спуск.

Така модель часто використовується для вирішення задач класифікації - об'єкт можна віднести до класу $y = 1$, якщо передбачена моделлю ймовірність $P\{y = 1 \mid x\} > 0{,}5$, і до класу $y = 0$ при отриманні значення меншого за 0,5. Отримані після цього правила класифікації належать до лінійних класифікаторів.

1.3.2 Метод опорних векторів

Метод опорних векторів - потужна і універсальна модель машинного навчання, що здатна виконувати лінійну або нелінійну класифікацію, особливо добре підходить для класифікації складних, але невеликих або середніх за розміром наборів даних.

Ядерний метод опорних векторів - це розширення звичайного методу опорних векторів, що дозволяє отримувати більш складні моделі, які не зводяться до побудови простих гіперплощин в просторі.

У низькорозмірних просторах лінійні моделі накладають дуже жорсткі обмеження, оскільки лінії і гіперплощини мають обмежену гнучкість. Один із способів зробити лінійну модель більш гнучкою - додати нові ознаки, наприклад, додати взаємодії або поліноми вхідних ознак [5].

У новому представленні даних вже можна відокремити два класи один від одного, використовуючи лінійну модель, гіперплощиною в просторі ознак. Додавання нелінійних ознак може поліпшити прогнозуючу силу лінійної моделі. Однак часто невідомо, які ознаки необхідно додати, і додавання більшої кількості ознак (наприклад, розгляд всіх можливих взаємодій в 100-вимірному просторі ознак) може дуже сильно збільшити вартість так складність обчислень. Однак існує математичний прийом, який дозволяє навчити класифікатор в багатовимірному просторі, практично не вдаючись до обчислення нового, можливо, дуже високорозмірного простору. Цей прийом відомий під назвою «ядерний метод» (kernel trick) і він безпосередньо обчислює евклідові відстані (точніше, скалярні добутки точок даних), щоб отримати розширений простір ознак без фактичного додавання нових ознак.

Існують два способи помістити дані в високорозмірний простір, які найчастіше використовуються методом опорних векторів: поліноміальне ядро, яке обчислює всі можливі поліноміальні комбінації вихідних ознак до певної міри, і ядро RBF (радіальна базисна функція), також відоме як гаусівське ядро. Гаусівське ядро відповідає нескінченному простору ознак. Пояснити гаусівське ядро можна так: воно розглядає всі можливі поліноми всіх ступенів, однак важливість ознак знижується зі зростанням ступеня.

В ході навчання метод опорних векторів обчислює важливість кожної точки навчальних даних з точки зору визначення вирішальної границі між двома класами. Зазвичай лише частина точок навчального набору важлива для визначення границь прийняття рішень: точки, які лежать на границі між класами. Вони називаються

опорними векторами (support vectors) і дали свою назву методу опорних векторів. Щоб отримати прогноз для нової точки, вимірюється відстань до кожного опорного вектора. Класифікаційне рішення приймається, виходячи з відстаней до опорних векторів, а також важливості опорних векторів, отриманих в процесі навчання. Відстань між точками даних вимірюється за допомогою гаусівського ядра:

$$k_{rbf}(x_1, x_2) = \exp(-\gamma\|x_1 - x_2\|^2),$$

де $x1$, $x2$ - точки даних, || $x1$ - $x2$ || - евклідова відстань, а γ - параметр, який регулює ширину гаусівського ядра.

Даний алгоритм має два основних параметри: C і гамма. Параметр гамма регулює ширину гаусівського ядра. Параметр гамма задає ступінь близькості розташування точок. Параметр C являє собою параметр регуляризації, аналогічний тому, що використовується в лінійних моделях. Він обмежує важливість кожної точки. Більш високі значення C відповідають меншій регуляризації. Іншими словами, використовуючи високе значення параметра C, модель намагається підлаштуватися до навчальних даних якнайкраще, тоді як при низьких значеннях параметра C моделі роблять більший акцент на пошуку вектора коефіцієнтів (w), близького до нуля. Використання низьких значень C призводить до того, що модель намагається підлаштуватися під «більшість» точок даних, тоді як використання більш високих значень C підкреслює важливість того, щоб кожна окрема точка даних була класифікована правильно [6].

Ядерний метод вимагає масштабування даних, тобто приведення всіх значень до однієї шкали. Крім того, алгоритм дуже чутливий до налаштування параметрів, розглянутих вище. Моделі методу опорних векторів важко досліджувані, важко зрозуміти, чому був зроблений саме такий прогноз. Однак дані алгоритми стійкі до похибок, так як на результат класифікації найбільш впливають точки, що лежать на границі між класами.

1.3.3 Градієнтний бустинг

Основна ідея градієнтного бустинга полягає в об'єднанні безлічі простих моделей (в даному контексті відомих під назвою слабкі учні або weak learners), дерев невеликої глибини. Кожне дерево може дати хороші прогнози тільки для частини даних, і таким чином для ітеративного покращення якості додається все більша кількість дерев. На кожній конкретній ітерації нова модель навчається з урахуванням помилки попередніх моделей. Градієнтний бустинг дерев часто займає перші рядки в змаганнях по машинному навчання, а також широко використовується в комерційних сферах. На відміну від випадкового лісу він, як правило, трохи більш чутливий до налаштування параметрів, однак при правильно заданих параметрах може дати більш високе значення правильності [7].

Був розглянутий найбільш загальний метод бустинга, окремими випадками або модифікаціями якого, так чи інакше, є сучасні методи бустинга. Розглянута задача розпізнавання об'єктів у багатовимірному просторі X з простором міток Y. Нехай нам дана навчальна вибірка $\{x_i\}^N$, де $x_i \in X$. І нехай на ній відомі істинні значення міток кожного об'єкта $\{y_i\}^N$, де $y_i \in Y$. Необхідно побудувати розпізнаваючий оператор, який якомога точніше зможе передбачати мітки для кожного нового об'єкта $x \in X$. Нехай нам задано деяке сімейство базових алгоритмів H, кожен елемент $h(x; a) \in H : X \to R$ якого визначається деяким вектором параметрів $a \in A$.

Фінальний алгоритм класифікації представлено у вигляді композиції:

$$F_M(x) = \sum_{m=1}^{M} b_m h(x; a_m), b_m \in R, a_m \in A.$$

Однак підбір оптимального набора параметрів $\{a_m, b_m\}^N$ – дуже працемістке завдання. Тому потрібно намагатися побудувати таку композицію шляхом жадібного нарощування, кожен раз додаючи в суму доданок, що є найбільш оптимальним алгоритмом з можливих.

Припустимо, що вже побудований класифікатор F_{m-1} довжини m−1. Таким чином задача зводиться до пошуку пари найбільш оптимальних параметрів $\{a_m, b_m\}$ для класифікатора довжини m:

$$F_m(x) = F_{m-1}(x) + b_m h(x; a_m), b_m \in R, a_m \in A$$

Оптимальність тут мається на увазі відповідно до принципу явною максимізації відступів. Це означає, що вводиться деяка функція втрат:

$$Q = \sum_{i=1}^{N} L(y_i, F_m(x_i)) \to \min$$

яка показує, наскільки "сильно" передбачена відповідь $F_m(x_i)$ відрізняється від правильної відповіді y_i. І потім мінімізується функціонал помилки. Функціонал помилки Q - дійсна функція, що залежить від точок $\{F_m(x_i)\}^N$ в N-мірному просторі, і необхідно вирішити задачу мінімізації цього функціоналу. Зроблено це реалізацією одного з кроку методу градієнтного спуску. В якості точки, для якої ми буде шукати оптимальний приріст, розглянемо F_{m-1}. Знайдемо градієнт функціоналу помилки:

$$\nabla Q = \left[\frac{\partial Q}{\partial F_{m-1}}(x_i)\right]_{i=1}^{N} = \left[\frac{\partial(\sum_{i=1}^{N} L(y_i, F_{m-1}))}{\partial F_{m-1}}(x_i)\right]_{i=1}^{N} = \left[\frac{\partial L(y_i, F_{m-1})}{\partial F_{m-1}}(x_i)\right]_{i=1}^{N}$$

Таким чином, в силу методу градієнтного спуску, найбільш вигідно додати новий доданок в класифікатор наступним чином:

$$F_m = F_{m-1} - b_m \nabla Q, b_m \in R$$

де b_m підбирається лінійним пошуком по дійсним числам R:

$$b_m = \underset{b \in R}{\operatorname{argmin}} \sum_{i=1}^{N} L(F_{m-1}(x_i) - b \nabla Q_i)$$

Однак ∇Q є лише вектором оптимальних значень для кожного об'єкта x_i, а не базовий алгоритм з сімейства H, визначений ∀x ∈ X. Тому необхідно знайти $h(x, a_m) \in H$ найбільш схожий на ∇Q. Зробимо це мінімізувавши функціонал помилки, оснований на принципі явної максимізації відступів:

$$a_m = \underset{a \in A}{\operatorname{argmin}} \sum_{i=1}^{N} L(\nabla Q_i, h(x_i, a)) \equiv \text{обучить} \left(\{x_i\}_{i=1}^{N}, \{\nabla Q_i\}_{i=1}^{N} \right),$$

що просто відповідає базовому алгоритму навчання. Далі шукається коефіцієнт b_m, використовуючи лінійний пошук:

$$b_m = \underset{b \in R}{\operatorname{argmin}} \sum_{i=1}^{N} L(F_{m-1}(x_i) - b h(x_i, a_m)).$$

Далі розглянемо даний алгоритм безпосередньо для завдання бінарної класифікації. У разі бінарної класифікації $Y = \{-1, +1\}$. Тоді часто мається на увазі, що кожен алгоритм $h \in H$ повертає дійсну «ступінь» приналежності об'єкта до певного класу, а результуюча відповідь $\tilde{F}$ виходить застосуванням порогового правила до композиції. У разі класифікації зазвичай використовується функція втрат від одного аргументу: L(y, F) = L(yF), тобто відступ замінюється добутком справжнього класу і передбаченого значення. У такому випадку під градієнтом функціонала помилки можна мати на увазі вектор ваг навчальних об'єктів поелементно помножений на вірні значення класів:

$$\nabla Q = \left[\frac{\partial L(y_i F_{m-1})}{\partial F_{m-1}}(x_i) \right]_{i=1}^{N} = \left[y_i \frac{\partial L(y_i F_{m-1})}{\partial (y_i F_{m-1})}(x_i) \right]_{i=1}^{N} = [y_i w_i]_{i=1}^{N},$$

де $w_{i=\frac{\partial L(y_i F_{m-1})}{\partial (y_i F_{m-1})}}$. Тоді алгоритм навчання у відповідності з принципом максимізації відступів буде мати наступний вигляд:

$$h(x, a_m) = \text{обучить}\left(\{x_i\}_{i=1}^N, \{\nabla Q_i\}_{i=1}^N\right) =$$

$$= \underset{a_m \in A}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(\nabla Q_i h(x_i, a_m)\right) = \underset{a_m \in A}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(y_i w_i h(x_i, a_m)\right).$$

Таким чином ω_i можна розглядати з точки зору ваг (ступеня «важливості»), які надаються об'єктам і враховуються при навчанні кожного базового алгоритму.

На сьогоднішній день градієнтний бустинг є одним з найпотужніших алгоритмів розпізнавання та класифікації даних. Для даного алгоритму можна розглядати різні функції втрат. Це дозволяє вирішувати, як завдання класифікації, так і завдання регресії. До того ж, можливість вибору довільної функції втрат дозволяє акцентувати увагу на особливостях даних в задачі. Крім того, можливий розгляд будь-якого сімейства базових алгоритмів. Це дає широкі можливості врахування особливостей розв'язуваної задачі. Бустинг над вирішальними деревами вважається одним з найбільш ефективних варіантів бустинга [8]. А враховуючи, що вирішальні дерева в свою чергу теж використовують базові алгоритми (наприклад, порогові, лінійні і т.п.), в результаті виходить величезна кількість варіантів для налаштування даного алгоритму машинного навчання. Завдяки достатній простоті методу і чіткому математичному обґрунтуванню, в кожній конкретній варіації бустинга можливо провести деякі математичні та алгоритмічні оптимізації, які можуть помітно прискорити роботу алгоритму.

Однак градієнтний бустинг не позбавлений недоліків. Це трудомісткий метод, і працює він досить повільно. Найчастіше потрібна побудова сотень або навіть й тисяч базових алгоритмів для композиції. Також варто врахувати, що без додаткових модифікацій він має властивість повністю підлаштовуватися під дані, в тому числі під помилки і похибки в них. Ідея бустинга зазвичай погано застосовується до побудови композиції з досить складних і потужних алгоритмів. Побудова такої композиції займає

багато часу, а якість істотно не збільшується. В підсумку результати роботи градієнтного бустинга складно інтерпретованих, особливо якщо в композицію входять десятки алгоритмів.

Розглянемо одну з різновидів градієнтного бустинга - екстремальний градієнтний бустінг (XGBoost) [9]. Зокрема - коли параметр booster встановлений варіант gbtree, тобто модель буде будується на основі дерев рішень. У XGBoost розмір дерева і величина ваг контролюються стандартними параметрами регуляризації. Це теоретично призводить до відсутності необхідності оптимізації параметрів. На практиці все ж безліч параметрів використовується для управління розміром і формою дерев. Регуляризація, однак, виявилася дуже потужною і зробила алгоритм набагато більш стійким. У XGBoost дерева можуть мати різну кількість кінцевих вузлів. Крім того, в цьому алгоритмі використовується Ньютон-Бустинг, який включає метод наближень Ньютона-Рафсона, що забезпечує прямий шлях до мінімумів, ніж градієнтний спуск. Крім того, використовується додатковий параметр рандомізації для зменшення кореляції між деревами: чим менше кореляція між класифікаторами, тим краще ансамбль класифікаторів. Як правило, XGBoost швидше, ніж градієнтний бустінг, але градієнтний бустінг має широкий спектр застосування. XGBoost супроводжується надзвичайно потужної реалізацією, яка успішно використовується в рішенні безлічі завдань на сьогоднішній день. Це дало йому миттєву популярність, він використовується в багатьох змаганнях по машинноному навчанні, і часто дуже успішно.

Далі було розглянуто основні параметри, які зазвичай необхідно налаштувати, щоб отримати найкращі результати роботи моделі [10]. Зазвичай, щоб запобігти перенавчання, необхідно вибрати одну із стратегій попереднього обмеження - налаштувати параметри max_depth (максимальна глибина дерева, збільшення цього значення зробить модель більш складною, і підвищить ймовірність перенавчання), subsample (співвідношення вибірок навчальних примірників; установка його на 0,5 означає , що XGBoost буде випадковим чином відбирати половину навчальних даних до вирощування дерев, і це запобіжить перенавчання, розбиття відбуватиметься один

раз в кожній ітерації бустинга) або min_child_weight (мінімальна сума ваги, необхідна для дочірнього дерева, якщо в результаті кроку розбиття дерева вийде листовий вузол з сумою ваги екземпляра менше, ніж min_child_weight, то процес побудови припинить подальше розбиття). Крім попередньої обробки ще один важливий параметр градієнтного бустинга - це параметр learning_rate, який контролює, наскільки сильно кожне дерево буде намагатися виправити помилки попередніх дерев. Більш висока швидкість навчання означає, що кожне дерево може внести більш сильні коригування і це дозволяє отримати більш складну модель.

Додавання більшої кількості дерев в ансамбль, здійснюване за рахунок збільшення значення n_estimators, також збільшує складність моделі, оскільки модель має більше шансів виправити помилки на навчальному наборі.

Таким чином, основний недолік алгоритму XGBoost полягає в тому, що він вимагає ретельної настройки параметрів, і для навчання може знадобитися багато часу. Як і інші моделі на основі дерев рішень, алгоритм добре працює на даних, що представляють суміш бінарних і безперервних ознак, не вимагаючи масштабування. Як і інші моделі на основі дерева, він погано працює на високорозмірних розріджених даних.

1.3.4 Штучні нейронні мережі

Вивчаючи будову і зв'язок нейронів головного мозку, дослідники знайшли натхнення для того, щоб розробити архітектуру штучних нейронних мереж [11]. Штучні нейронні мережі існують вже досить давно: вони були вперше представлені ще в 1943 році нейрофізіологом Уорреном Маккаллохом і математиком Уолтером Піттсом. Перші успіхи штучних нейронних мереж до 1960-х років привели до широко поширеного переконання, що скоро буде реально створити по-справжньому інтелектуальні машини. Коли стало ясно, що це очікування не справдилося (як мінімум, на довгий час), фінансування проектів припинилося, і штучні нейронні мережі були забуті на кілька років [12].

Нещодавно нейронні мережі пережили своє відродження, завдяки «глибокому навчанню». Незважаючи на те, що глибоке навчання обіцяє великі перспективи в різних сферах використання машинного навчання, алгоритми глибокого навчання, як правило, жорстко прив'язані до конкретних випадків використання [13]. Далі розглянута модель багатошарового перцептрона (MLP), його також називають простою нейронною мережою прямого поширення, а іноді і просто нейронною мережою.

Перецептрон можна розглядати як узагальнення лінійних моделей, яке перш ніж прийти до рішення проблеми виконує кілька етапів обробки даних. У лінійної регресії прогноз отримують за допомогою наступної формули:

$$\hat{y} = w[0]x[0] + w[1]x[1] + \cdots + w[p]x[p] + b,$$

где $\hat{y}$ – это зважена сума вхідних ознак x[0],…, x[p]. Вхідні ознаки зважені по обрахованим в ході навчання коефіцієнтам w[0],…, w[p]. Їх можна представити графічно, як показано на рисунку 1.4.

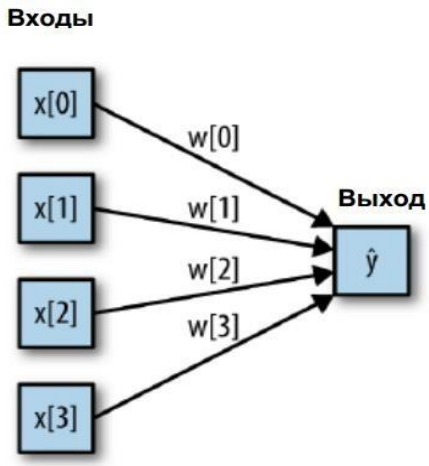

Рисунок 1.4 – Візуалізація логістичної регресії з коефіцієнтами.

Тут кожен вузол, показаний зліва, є вхідною ознакою, з'єднувальні лінії - коефіцієнтами, а вузол справа - це вихід, який є зваженою сумою входів.

Перцептрон складається з одного проміжного вхідного рівня, одного або декількох шарів лінійних порогових елементів, що називаються прихованими рівнями,

і одного кінцевого рівня лінійних порогових елементів, що називаються вихідним рівнем. Кожен шар, крім вихідного, має нейрон зміщення і повністю пов'язаний з наступним шаром, що показано на рисунку 1.5. Коли нейронна мережа має два або більше прихованих шарів, вона називається глибокої нейронною мережею.

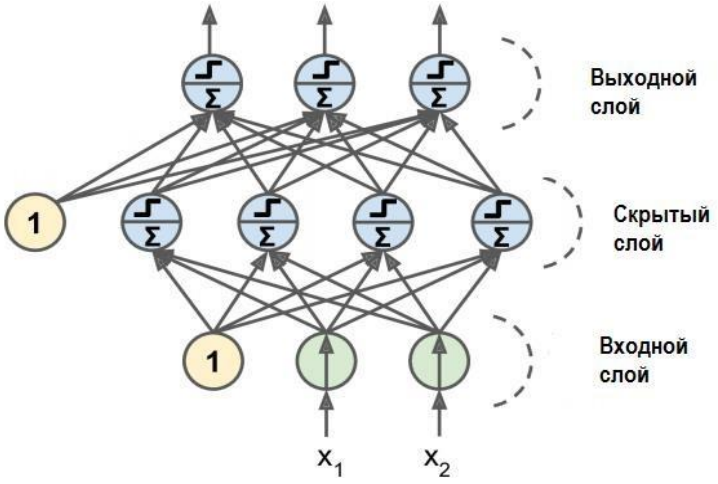

Рисунок 1.5 – Візуалізація багатошарового перцептрона.

У перцептрона процес обчислення зважених сум повторюється кілька разів. Спочатку обчислюються приховані елементи (hidden units), які представляють собою проміжний етап обробки. Вони знову об'єднуються за допомогою зважених сум для отримання кінцевого результату, що показано на рисунку 1.6.

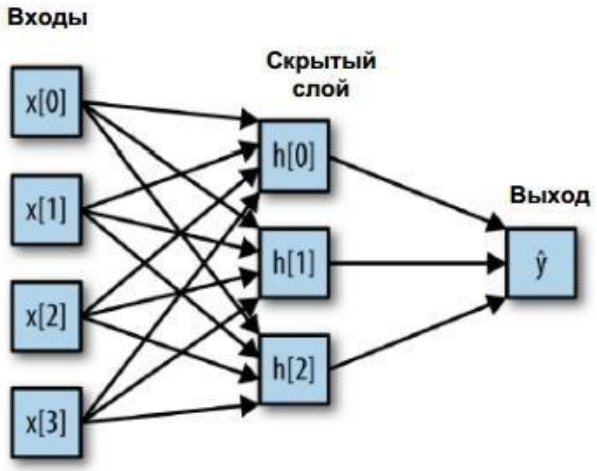

Рисунок 1.6 – Багатошаровий перцептрон з одним прихованим шаром.

У цій моделі набагато більше обчислюваних коефіцієнтів, так званих ваг, коефіцієнт між кожним входом і кожним прихованим елементом (які утворюють прихований шар або hidden layer) і коефіцієнт між кожним елементом прихованого шару і виходом. З математичної точки зору обчислення серії зважених сум - це те ж саме, що обчислення лише однієї зваженої суми, таким чином, щоб ця модель мала більш потужної прогнозної силою, ніж лінійна модель, потрібно виконати ще одну умову. Вона пояснюється на прикладі нейронної мережі з одним прихованим шаром. Вхідний шар просто передає входи прихованого шару мережі, або без перетворення, або виконавши спочатку стандартизацію входів. Потім відбувається обчислення зваженої суми входів для кожного елемента прихованого шару, до неї застосовується функція активації - зазвичай використовуються нелінійні функції випрямленний лінійний елемент (rectified linear unit або relu) або гіперболічний тангенс (hyperbolic tangent або tanh) [14].

У підсумку отримуються виходи нейронів прихованого шару. Ці проміжні виходи можуть вважатися нелінійними перетвореннями і комбінаціями первинних входів. Вони стають входами вихідного шару. Знову обчислюється зважена сума входів, застосовується функція активації і отримуються підсумкові значення цільової змінної. Функції активації relu і tanh показані на рисунку 1.7. Relu відсікає значення нижче нуля, в той час як tanh приймає значення від -1 до 1 (відповідно для мінімального і максимального значень входів). Будь-яка з цих двох нелінійних функцій дозволяє нейронній мережі на відміну від лінійної моделі обчислювати набагато складніші залежності.

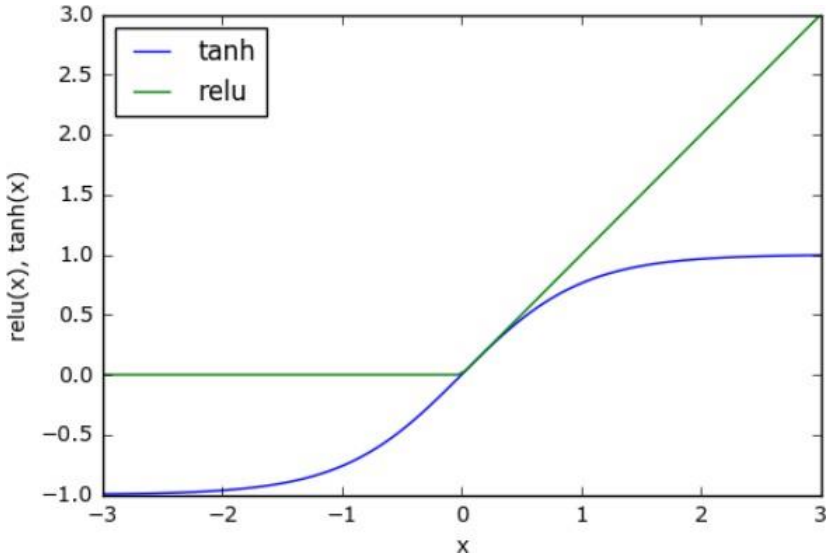

Рисунок 1.7 – Функції tanh і relu активацій.

Для невеликої нейронної мережі, зображеної на рисунку 4, повна формула обчислення y в разі регресії буде виглядати так (при використанні tanh):

$$h[0] = \tanh(w[0,0]x[0] + w[1,0]x[1] + w[2,0]x[2] + w[3,0]x[3])$$

$$h[1] = \tanh(w[0,0]x[0] + w[1,0]x[1] + w[2,0]x[2] + w[3,0]x[3])$$

$$h[2] = \tanh(w[0,0]x[0] + w[1,0]x[1] + w[2,0]x[2] + w[3,0]x[3])$$

$$\hat{y} = v[0]h[0] + v[1]h[1] + v[2]h[2]$$

Тут w - ваги між входом x і прихованим шаром h, а v - вагові коефіцієнти між прихованим шаром h і виходом y. Ваги v і w обчислюються за даними, x є вхідними ознаками, y -обчислений вихід, а h - проміжні обчислення.

Нейронні мережі вимагають того, щоб всі вхідні ознаки були виміряні в одному і тому ж масштабі, в ідеалі вони повинні мати середнє значення 0 і дисперсію 1. Необхідно відмасштабувати дані так, щоб вони відповідали цим вимогам.

Одна з головних переваг нейронних мереж полягає в тому, що вони здатні обробляти інформацію, що міститься у великих обсягах даних, і будувати неймовірно складні моделі. При наявності достатнього часу обчислень, даних і ретельного налаштування параметрів, нейронні мережі часто перевершують інші алгоритми машинного навчання для задач класифікації і регресії.

Це дає і свої мінуси. Нейронні мережі, особливо великі нейронні мережі, як правило, вимагають тривалого часу навчання. Вимагають ретельної попередньої обробки даних. Крім того, налаштування параметрів нейронної мережі - це критично важлива частина побудови моделі.

Щоб створити модель нейронної мережі на мові програмування Python для неї були створені кілька бібліотек, найбільш відомими з яких є keras, lasagna і Tensorflow. Ці бібліотеки пропонують гнучкий інтерфейс для побудови нейронних мереж і оновлюються відповідно до останніх досягнень в області глибокого навчання. У даній роботі для побудови нейронної мережі використовувалася бібліотека keras, яка використовує «під капотом» фреймоворк машинного навчання Tensorflow.

При побудові моделі для нейронної мережі в першу чергу потрібно визначити її архітектуру: шари і число нейронів в них. У keras є 2 способи визначення нейронної мережі: клас Sequential model і функціональний API [15]. Обидва поділяють мету визначення нейронної мережі, але використовують різні підходи. У даній роботі використовувався клас Sequential model.

Клас Sequential використовується для визначення лінійного стека рівнів нейронної мережі, які в сукупності складаються в модель.

Далі визначається конфігурація процесу навчання в блоці compile, де задаються три основних параметри: loss function, optimizer, metrics. Loss function (функція втрат) - вимірює точність моделі під час навчання. Необхідно мінімізувати значення цієї функції, щоб "направити" модель в потрібному напрямку. Optimizer - показує яким чином оновлюється модель на основі вхідних даних і функції втрат. Metrics - показує кроки тренування і тестування моделі, а також за допомогою якої метрики оцінюється кількість спостережень, які були класифіковані правильно. В процесі навчання моделі

важливі такі параметри як: batch_size, який обмежує кількість прикладів, що подаються моделі за раз, і кількість епох для навчання - epochs (одна епоха - це один раз повністю пройдена моделлю навчальна вибірка).

Якість результату роботи моделі нейронної мережі сильно залежить від правильного налаштування розглянутих вище параметрів.

1.4 Висновки до розділу

В даному розділі були розглянуті існуючі рішення інструментів по аналізу текстових даних такі як Perspective API та Tisane API. Були виділені їх сильні та слабкі сторони. Щодо слабких сторін, можна виділити у них відсутність інтеграції з будь-якою із популярних платформ коментування. Також, були розглянуті та проаналізовані різні алгоритми машинного навчання для використання їх у вирішенні задачі аналізу текстових коментарів для їх подальшого аналізу. Як найбільш сучасний та ефективний інструмент машинного навчання було вирішено використати в роботі нейронну мережу для аналізу коментарів.

2 РОЗРОБКА МОДЕЛІ МАШИННОГО НАВЧАННЯ

2.1 Постановка задачі

Для інтелектуальної системи фільтрації коментарів з використанням машинного навчання постає задача в розробці сервісу оцінки коментаря за деякими ознаками у наступному вигляді:

1. На вхід подається текстовий коментар у вихідному вигляді.
2. Вхідний коментар очищається від зайвого вмісту: видаляються розділові знаки та інші пунктуаційні символи, видаляються емодзі, видаляються також так звані стоп-слова. Текст може містити стоп-слова, такі як "the", "is", "are", тому зазвичай їх видаляють з тексту на етапі підготовки тексту до аналізу.
3. Далі цілий коментар проходить процес токенізації. Кожне речення в коментарі розбивається на окремі слова, кожне з яких потім отримує цифрове значення індекса для кращого розуміння тексту в процесі роботи машинного навчання.
4. На основі історичних тренувальних даних, кожному коментарю виставляється значення 0 або 1 на ознаках, які йому відповідають.
5. За рахунок цих даних модель навчається прогнозувати вірогідність відповідності коментаря до кожної з 6 ознак оцінки коментаря.
6. Результатом роботи моделі є повернення даних по кожній з ознак, яке має значення від 0 до 1, де значення, чим ближче до 1 , тим більше воно буде показувати наскільки точно коментар є таким, що відповідає до тієї чи іншої ознаки та, в результаті, може буде відфільтрованим, видаленим тощо.

Отже — для вирішення задачі оцінки коментаря за різними ознаками для основи створення моделі машинного навчання було вибрано нейронні мережі, оскільки даний алгоритм машинного навчання може охоплювати різні типи зразків вхідних даних, в випадку даної роботи вхідний зразок — характеристики коментаря в числовому векторі. Тобто, з'являється можливість охоплювати контекст зразків вхідних даних довільної довжини, що й необхідно для вирішення поставленої задачі оцінювання коментаря.

2.2 Препроцесинг вхідних даних

Процес перетворення даних у те, що комп'ютер може зрозуміти, називається препроцесингом або попередньою обробкою даних. Однією з основних форм попередньої обробки є фільтрація зайвих даних. При обробці натуральної мови непотрібні слова (дані) називаються стоп-словами.

2.2.1 Видалення стоп-слів

Ідея обробки натуральних мов полягає у виконанні певної форми аналізу чи обробки, де машина може зрозуміти, принаймні, до якогось рівня, що вхідний текст означає, говорить чи має на увазі текст.

Це, очевидно, величезне завдання, але для цього є послідовно описані кроки, яких може дотримуватися кожен. Основна ідея, однак, полягає в тому, що комп'ютери просто ніколи не зрозуміють слів буквально. У людини пам’ять розбивається на електричні сигнали в мозку, у вигляді нейронних груп, що працюють за зразками. Виявляється, комп'ютери зберігають інформацію дуже схожим чином. Тому потрібен спосіб максимально наблизитися до цього, якщо будемо імітувати те, як люди читають і розуміють текст. Як правило, комп'ютери використовують цифри для всього, але в програмуванні часто зустрічаються позначення, де використовуються двійкові сигнали такі як True або False, які переводяться на 1 або 0. Для цього потрібно визначити спосіб перетворення слів у значення, у числа чи схеми сигналу. Процес перетворення даних у те, що комп'ютер може зрозуміти, називається "попередньою обробкою". Однією з основних форм попередньої обробки є фільтрування зайвих даних. При обробці натуральних мов непотрібні слова або дані називаються стоп-словами.

Відразу варто наголосити увагу на те, що деякі слова несуть більше значення та сенсу, ніж інші слова. Також можна побачити, що деякі слова є просто зайвими і є словами-заповнювачами. Прикладом одного з найпоширеніших, неофіційних, зайвих слів є фраза «хм».

Ніхто не хочу, щоб ці слова займали місце в нашій базі даних або займали цінний час на обробку. Як такі, ці слова називаються "стоп-словами", оскільки вони дуже часто є зайвими, і з ними нічого не потрібно робити. Для поставленої задачі стоп-слова маються на увазі такі, які просто не містять та не несуть будь-якого корисного для аналізу значення, і ми хочемо їх видалити.

Для роботи зі знаходженням та видаленням стоп-слів зі змісту вхідного коментаря була обрана Python бібліотека NLTK або Natural Language Toolkit. Ця бібліотека зберігає список стоп-слів з 16 мов та дозволяє з легкістю ідентифікувати їх та видалити з вхідних даних.

2.2.2 Токенізація речень

Токенізація - це процес, за допомогою якого велика кількість тексту ділиться на менші частини, що називаються лексемами.

Обробка натуральних мов використовується для побудови таких додатків, як класифікація тексту, створення інтелектуальних чат-ботів, сентиментальний аналіз, переклад мови тощо. Для досягнення вищезазначеної мети важливо зрозуміти закономірність тексту. Ці лексеми дуже корисні для пошуку таких шаблонів, а також вважаються базовим кроком для стримування та лематизації.

Для препроцесінгу тексту було вирішено розділити речення на слова. Вихід токенізованих слів може бути перетворений для кращого розуміння тексту в додатках машинного навчання. Він також може бути наданий як вхід для подальших етапів очищення тексту, таких як вилучення пунктуації, видалення чи виведення символів із числа. Моделям машинного навчання потрібні чисельні дані для навчання та прогнозування. Токенізація слів стає важливою частиною тексту (рядка) для цифрового перетворення даних.

Прикладом роботи токенізатора речень на слова в Python показано на рисунку 2.1.

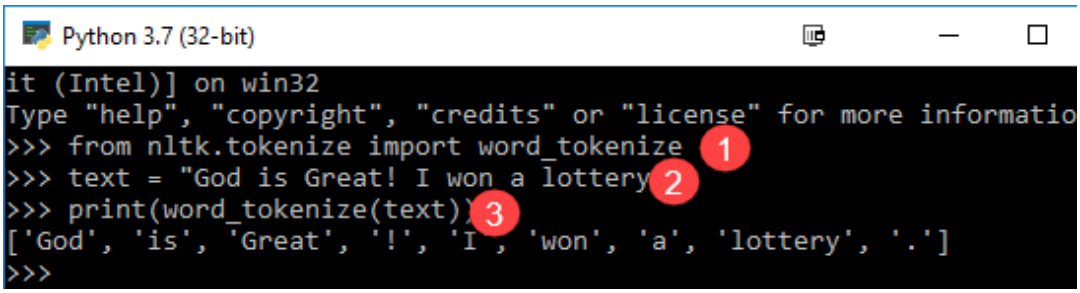

Рисунок 2.1 – Приклад роботи токенізатора речень

Спочатку імпортується сам модуль токенізації із бібліотеки Natural Language Toolkit. Потім ініціюється змінна text з власне текстом для розбиття його на токени. Далі змінна з текстом передається на вхід методу word_tokenize і виводиться результат роботи. Цей модуль розбиває кожне слово через розділові знаки, які можна побачити в результаті.

Нейронні мережі можуть лише навчитися знаходити шаблони в числових даних, і тому, перш ніж подавати на вхід коментар в нейронну мережу, доведеться перетворити кожне токенізоване слово в числове значення. Типовий процес кодування такий:

1. Для всіх текстових даних - у нашому випадку коментарі - ми беремо кожне з унікальних слів, що з’являються у цьому наборі даних, і записуємо їх як лексику для нашої моделі.
2. Ми кодуємо кожне словникове слово як унікальне ціле число, яке називається лексемою. Часто ці лексеми присвоюються залежно від частоти появи слова в наборі даних. Отже, слово, яке найчастіше з’являється в наборі даних, матиме пов’язаний маркер: 0. Наприклад, якщо найпоширенішим словом було "apple", воно матиме відповідне значення лексеми 0. Тоді наступне найпоширеніше слово буде буде позначено як 1, і так далі.
3. У коді ця асоціація слово-лексема представлена в словнику, що пов’язує кожне унікальне слово до його токена – цілого числа.
4. В кінця, після віднесення цих лексем до окремих слів, ми можемо токенізувати всі слова з кожного речення коментаря.

Будь-який один об’єкт у наборі даних, як один коментар, ми розглядаємо як список послідовності слів. Тоді ми використовуємо словник токенів для перетворення цього списку слів у список цілих значень.

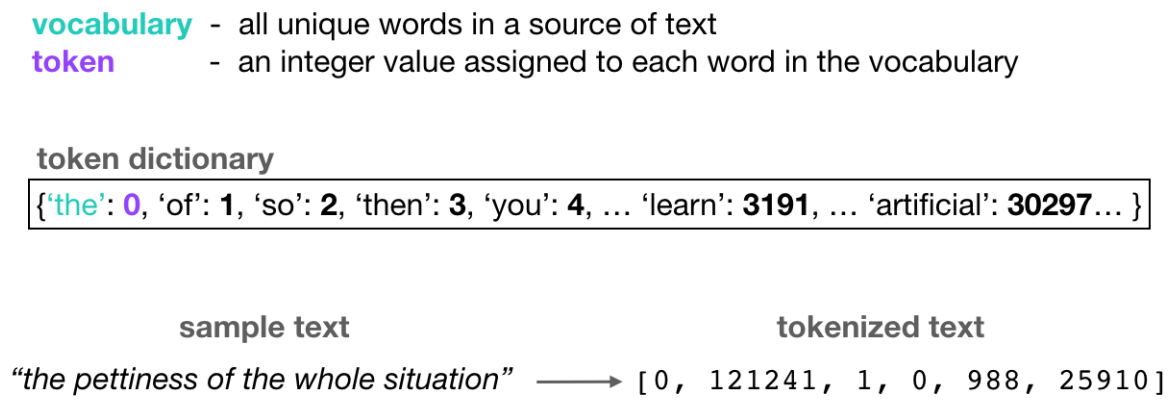

Рисунок 2.2 – Словник токенізованих слів

Важливо зазначити, що ці значення лексем не мають особливого умовного значення. Тобто ми зазвичай думаємо, що значення 1 близьке до 2 і так далі до 1000. Як інший приклад, ми вважаємо значення 10 як середнє значення 2 і 18. Однак слово, що токенізоване як 1 не обов’язково є подібним до слова, що токенізоване як 2, а це як слово, що токенізоване як 1000. Типові поняття числової відстані не говорять нам нічого про зв’язки між окремими словами.

2.3 Згорткова нейронна мережа для класифікації коментарів

З самого початку згорткові нейронні мережі були розроблені у спільноті обробки зображень нейронними мережами, де вони досягли проривних результатів у розпізнаванні об'єкта із заздалегідь визначеної категорії, наприклад, кішка, велосипед тощо.

Згорткова нейронна мережа, як правило, включає в собі дві основні операції, які можуть бути представлені в якості екстракторів функцій: згортання та об'єднання [16].

Результат такої послідовності операцій, як правило, з'єднується з повнозв’язним шаром, який в принципі по своїй суті такий же, як і традиційна багатошарова перцептронна нейронна мережа (MLP). На рисунку 2.3 зображений приклад роботи згорткової нейронної мережі для розпізнавання об’єктів на фото.

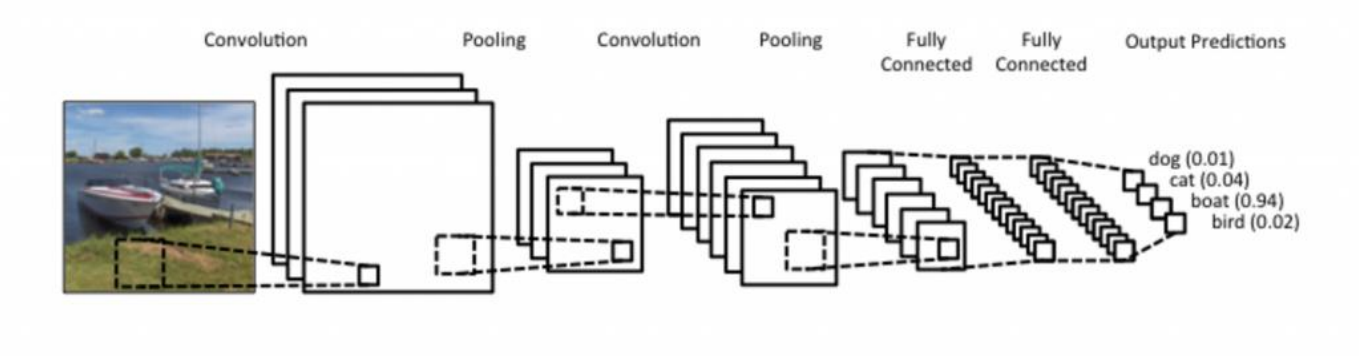

Рисунок 2.3 – Приклад роботи згорткової нейронної мережі

Ми можемо розглядати вхідне зображення як матрицю, де кожен запис представляє кожен піксель, і значення від 0 і 255, що представляє інтенсивність яскравості. Припустимо, що це чорно-біле зображення із лише одним каналом.

Один із способів зрозуміти операцію згортання - це уявити розміщення фільтра згортання або ядра у верхній частині вхідного зображення, розташованого таким чином, щоб ядро та верхній лівий кут зображення збігалися, а потім множили значення матриці вхідного зображення з відповідними значеннями у фільтрі згортки.

Всі помножені значення потім додаються разом, у результаті чого утворюється один скаляр, який розміщується в першому положенні результуючої матриці.

Потім ядро переміщує x-кількість пікселів праворуч, де x позначається довжиною кроку і є параметром структури згорткової нейронної мережі. Процес множення повторюється, щоб наступне значення в матриці результатів було обчислено і заповнено.

Потім цей процес повторюється, спочатку охоплюючи цілий ряд, а потім зміщуючи стовпчики вниз на однакову довжину кроку, поки всі записи у вхідному зображенні не будуть охоплені.

Результатом цього процесу є матриця з усіма заповненими записами, що називається згорнутою ознакою або картою функцій введення.

Вхідне зображення може бути об'єднано декількома ядрами згортки одночасно, створюючи один вихід для кожного ядра. Приклад операції згортання зображено на рисунку 2.4.

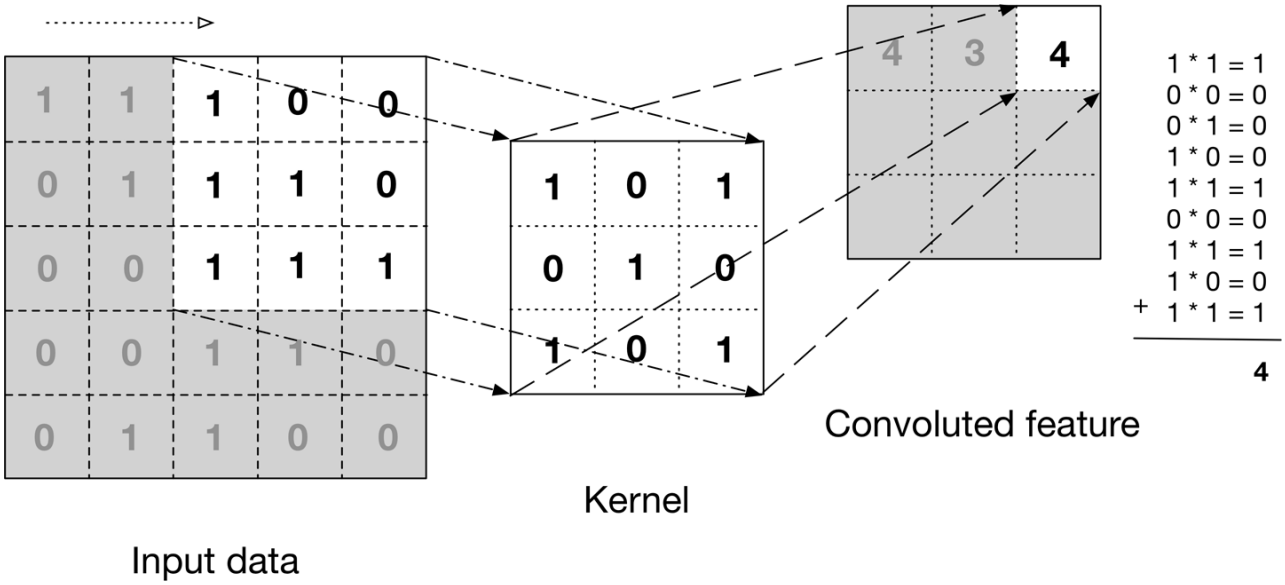

Рисунок 2.4 – Операція згортання згорткової нейронної мережі

Далі йде шар об'єднання чи зменшення розміру, який полягає у застосуванні деякої операції над областю / патчами на карті вхідних функцій та витягування деякого репрезентативного значення для кожної з проаналізованих областей / патчів.

Цей процес якимось чином схожий на описану раніше згортку, але замість трансформації локальних патчів за допомогою вивченого лінійного перетворення (тобто фільтра згортки) вони трансформуються за допомогою жорстко кодованої операції.

Дві найпоширеніші операції об’єднання - це максимальне та середнє об'єднання. Максимальне об'єднання вибирає максимум значень у області вхідної мапи картки кожного кроку та середнє об'єднання середнього значення в області значень. Отже, вихід на кожному кроці є одиночним скаляром, що призводить до значного зменшення розміру виходу. Приклад операції об’єднання зображено на рисунку 2.5.

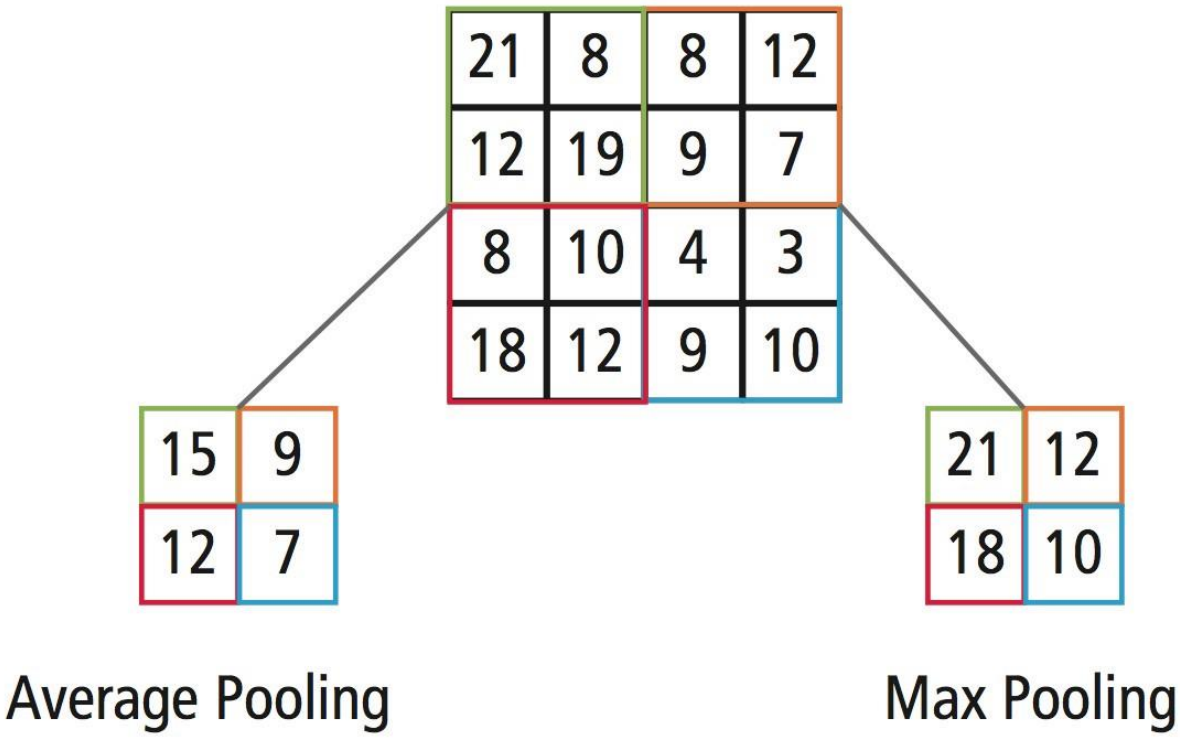

Рисунок 2.5 – Операція об’єднання згорткової нейронної мережі

Дві описані вище операції згортання та об'єднання можна розглядати як екстрактори функцій, тоді ми передаємо ці функції, як перероблений вектор одного ряду, на вхід нейронній мережі, наприклад, багатошаровому перцептрону, який був побудований для вирішення задачі класифікації. Приклад зображений на рисунку 2.6.

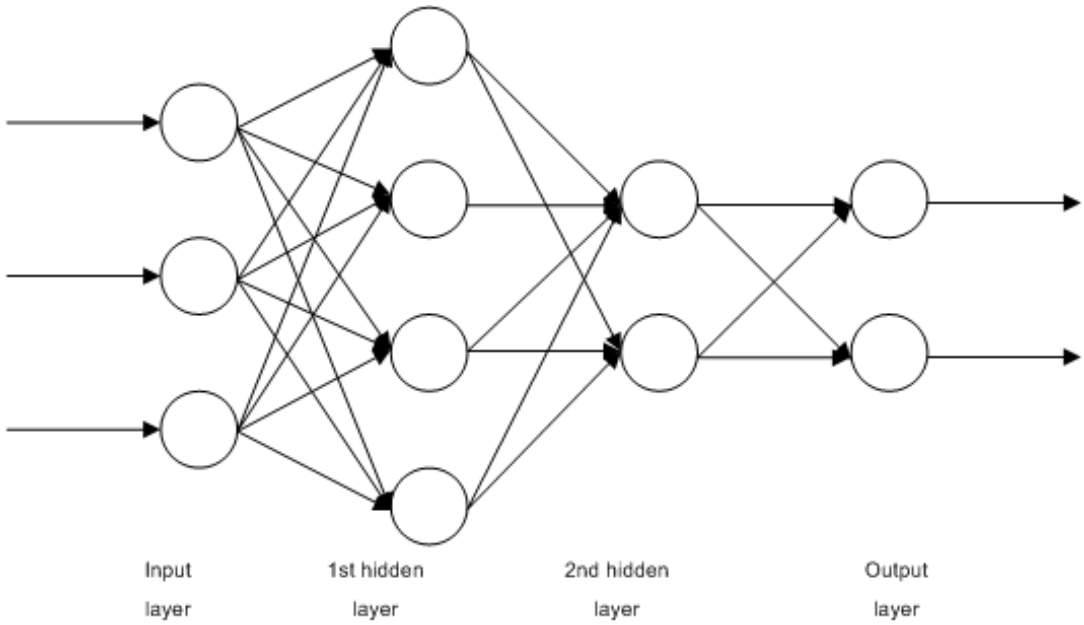

Рисунок 2.6 – Приклад багатошарового перцептрона для задач класифікації.

У випадку із завданнями класифікації тексту, тобто при застосуванні згорткової нейронної мережі до тексту замість зображень, ми маємо 1 розмірний масив, що представляє текст. Тут архітектура згорткової нейронної мережі змінюється на 1D операції конвертування та об'єднання.

Однією з найбільш типових завдань класифікації тексту, де використовується згорткової нейронної мережі, є класифікація речень, тобто класифікація речення на набір заздалегідь визначених категорій шляхом врахування n-грамів, тобто це слова або послідовність слів, а також символи чи послідовність символів.

Дано послідовність слів $w_{1:n} = w_1, \dots, w_n$, де кожне слово асоціюється з внутрішнім вектором розмірності d. 1D згортка шириною k є результатом переміщення розсувного вікна розміром k над реченням та застосування одного і того ж фільтра згортки або ядра до кожного вікна в послідовності, тобто крапкового добутку між конкатенацією вбудовування векторів у заданому вікні та ваговому векторі u, після чого часто слідує нелінійна функція активації g.

Розглядаючи вікно слів wi,…, wi + k, з'єднаний вектор і-го вікна є:

$$x_i = [w_i, w_{i+1}, \dots, w_{i+k}] \in R^{k \times d},$$

Фільтр згортки застосовується до кожного вікна, в результаті чого отримуються скалярні значення r_i, кожне для і-го вікна:

$$r_i = g(x_i \cdot u) \in R,$$

На практиці зазвичай застосовується більше фільтрів, $u_1, \dots, u_l$, які потім можуть бути представлені у вигляді вектора, помноженого на матрицю U та з додаванням терміна зміщення b:

$$r_i = g(x_i \cdot U + b),$$

Де $r_i \in R^l,\ x_i \in R^{k \times d}, U \in R^{k \cdot d \times l}, b \in R^l.$

Приклад згортки речень у позначенні вектора-конкатенації зображено на рисунку 2.7.

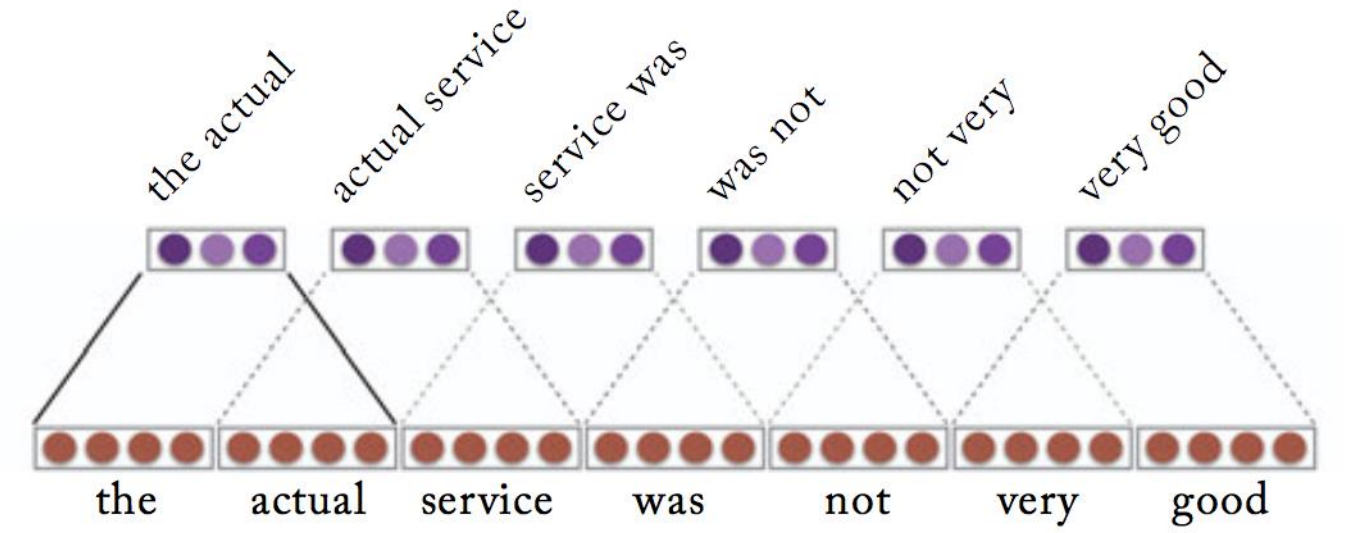

Рисунок 2.7 – Приклад згортання речення з k = 2 та розмірним виходом l = 3.

За основу створення власної реалізації згорткової нейронної мережі була взята нейронна мережа, розроблена для класифікації відгуків про фільми [17], що працює за наступним алгоритмом:

1. Обробка усіх відгуків про фільми та їх міток про настрій відгука, щоб видалити не потрібну інформацію та закодувати мітки (позитивна = 1, негативна = 0);
2. Завантаження тренованої моделі та використання її для токенізації кожного відгуку;
3. Стандартизація кожного відгуку для приведення їх до однакової довжини;
4. Створення навчального, підтверджувального та тестового наборів даних;
5. Визначити та навчити модель;
6. Тестування моделі на позитивних та негативних відгуках.

На рисунку 2.8 зображена архітектура такої моделі.

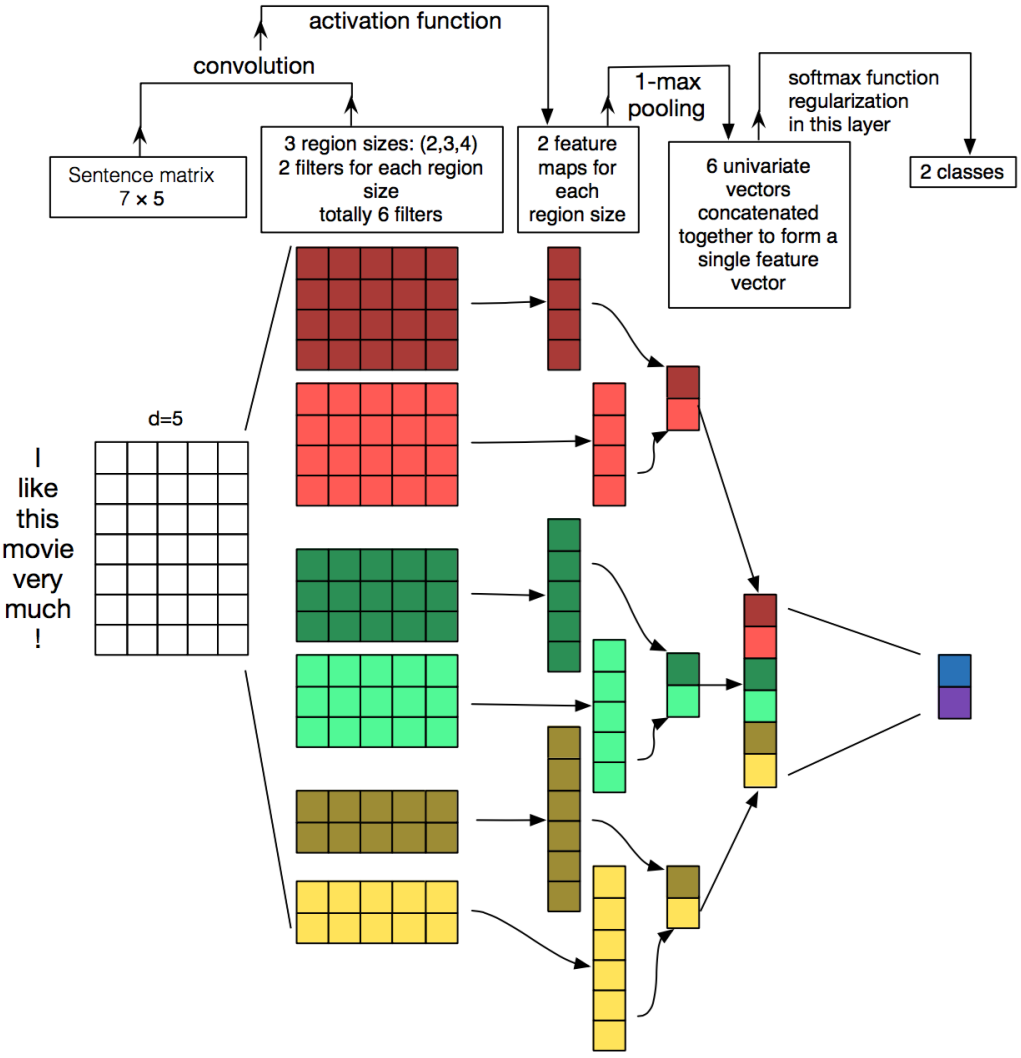

Рисунок 2.8 – Приклад архітектури згорткової нейронної мережі для класифікації відгуків про фільми.

Результат кожної згортки буде спрацьовувати, коли буде виявлено спеціальний шаблон (паттерн). Змінюючи розмір ядер і об'єднуючи їх вихідні дані, можна виявити шаблони кратних розмірів (2, 3 або 5 суміжних слів).

Шаблони можуть бути виразами, на кшталт "я ненавиджу", "дуже добре", і тому згорткова нейронна мережа може ідентифікувати їх у реченні незалежно від їх позиції.

Беручи за основу описану вище модель нейронної мережі, мною була побудована модель згорткової нейронної мережі, що націлена на оцінку вхідних даних, а саме коментарів за такими ознаками як: токсичність, нецензурна лексика, погроза, образа, та персональна ненависть. Структурна схема результуючої моделі показана на рисунку 2.9.

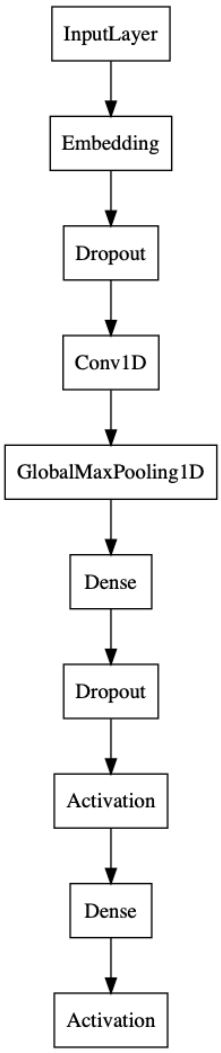

Рисунок 2.9 – Структурна схема створеної моделі машинного навчання

2.4 Висновки до розділу

В розділі було сформульоване завдання для розробки моделі машинного навчання, описані етапи препроцесингу тексту для його подальшого аналізу нейронною мережою. Основними етапами препроцесингу можна виділити видалення стоп-слів з тексту та подальшу токенізацію тексту. Для аналізу коментарів було обрано використовувати згорткову нейронну мережу, так як вона рахується одним з лідерів в області класифікації текстових даних серед нейронних мереж. Створена та описана розроблена модель машинного навчання.

3 РОЗРОБКА СИСТЕМИ

Діаграма сценаріїв із функціональними вимогами наведена у додатку Г, детальні описи кожного із сценаріїв наведено у таблицях нижче.

3.1 Сценарії використання системи

Таблиця 3.1 – Сценарій використання

Назва	Реєстрація користувача
ID	1
Опис	Для отримання доступу до функціоналу системи новий користувач має зареєструватися.
Актори	Користувач
Вигоди компанії	Отримання нових користувачів, що можуть приносити дохід
Частота користування	Часто
Тригери	Новий користувач виконує запит
Передумови	Новий користувач, з наявним логіном, паролем та email. Або реєстрація через соц. мережі.
Постумови	Створено нового користувача в системі
Основний розвиток	Виконується запит і передає дані про користувача, відбувається реєстрація, користувач отримує доступ до користування системою.
Альтернативні розвитки	-
Винятки	-

Таблиця 3.2 – Сценарій використання

Назва	Авторизація користувача
ID	2
Опис	Для отримання доступу до функціоналу системи користувач має авторизуватися
Актори	Не авторизований користувач
Вигоди компанії	Доступ до системи мають тільки авторизовані користувачі
Частота користування	Постійно
Тригери	Не авторизований користувач виконує запит
Передумови	Наявність логіну та пароля користувача
Постумови	Надано доступ до системи
Основний розвиток	Виконується запит і передаються дані для авторизації користувача
Альтернативні розвитки	-
Винятки	-

Таблиця 3.3 – Сценарій використання

Назва	Відновлення пароля користувача
ID	3
Опис	Для відновлення доступу користувача. Неавторизованому користувачу необхідно отримати доступ до функцій системи
Актори	Користувач
Вигоди компанії	Не втрачати зареєстрованих користувачів
Частота користування	Рідко
Тригери	Користувач виконує запит
Передумови	Наявність електронної пошти користувача та доступу до неї
Постумови	Дані про користувача змінено в системі
Основний розвиток	Виконується запит і передаються дані про користувача, змінюється його пароль доступу до системи
Альтернативні розвитки	-
Винятки	-

Таблиця 3.4 – Сценарій використання

Назва	Вийти з акаунта системи
ID	4
Опис	Для завершення доступу до системи
Актори	Авторизований користувач
Вигоди компанії	Контроль за одночасно авторизованими користувачами
Частота користування	Часто
Тригери	Користувач виконує запит
Передумови	Авторизований користувач
Постумови	Користувача деавторизовано із системи
Основний розвиток	Користувач виконує запит, передаються дані, користувач більше немає можливості користуватися функціоналом системи
Альтернативні розвитки	-
Винятки	-

Таблиця 3.5 – Сценарій використання

Назва	Отримати інформацію про користувача системи
ID	5
Опис	Для перегляду інформації про користувача та його активності в системі
Актори	Користувач, адміністратор
Вигоди компанії	Контроль і аналіз користування системою
Частота користування	Постійно
Тригери	Користувач виконує запит
Передумови	Авторизований користувач або адміністратор
Постумови	Отримується інформація про користувача та його активність в системі, адміністратор бачить загальну інформацію по всіх користувачах
Основний розвиток	Користувач або адміністратор виконує запит, після система повтертає інформацію про користувача та його активність в системі
Альтернативні розвитки	-
Винятки	-

Таблиця 3.6 – Сценарій використання

Назва	Додати нове відео для фільтрації коментарів
ID	6
Опис	Для початку фільтрації коментарів
Актори	Користувач
Вигоди компанії	Користувач отримує доступ до основного функціоналу системи
Частота користування	Часто
Тригери	Користувач виконує запит
Передумови	Авторизований користувач, знаходиться на сторінці додавання нового відео в систему
Постумови	Розпочинається автоматична фільтрація коментарів на відео, зменшення кількості токсичних коментарів
Основний розвиток	Користувач виконує запит, передаються дані, додається нове відео в систему, розпочинається фільтрація коментарів
Альтернативні розвитки	Додавання нового відео може не відбутися, якщо тип відео не є прямим ефіром
Винятки	-

Таблиця 3.7 – Сценарій використання

Назва	Перегляд доданих відео в системі
ID	7
Опис	Для перегляду доданих відео користувачем та керування ними
Актори	Користувач, адміністратор
Вигоди компанії	Користувач отримує доступ до власних доданих відео в системі та отримує можливість до керування ними
Частота користування	Часто
Тригери	Користувач натискає відповідну кнопку
Передумови	Авторизований користувач, знаходиться на будь-якій сторінці системи
Постумови	Користувач отримує доступ до перегляду власних доданих відео в систему та має змогу керувати ними. Адміністратор бачить відео всіх користувачів в системі.
Основний розвиток	Користувач виконує запит, натискаючи відповідну кнопку, передаються дані, отримується інформація про відео в системі
Альтернативні розвитки	-
Винятки	-

Таблиця 3.8 – Сценарій використання

Назва	Редагування доданого відео
ID	8
Опис	Для редагування доданого користувачем відео
Актори	Користувач
Вигоди компанії	Користувач отримує доступ до редагування доданих відео: назва, активність моніторингу за коментарями, параметри фільтрації коментарів
Частота користування	Рідко
Тригери	Користувач вибирає, що він хоче відредагувати
Передумови	Користувач знаходиться на сторінці з його відео та натискає на відповідний ID відео
Постумови	Зміна інформації про відео
Основний розвиток	Користувач знаходиться на сторінці з його відео та натискає на відповідний ID відео, вибирає те, що він бажає редагувати та зберігає зміни
Альтернативні розвитки	-
Винятки	-

Таблиця 3.9 – Сценарій використання

Назва	Видалення відео з системи
ID	9
Опис	Користувач видаляє відео із системи
Актори	Користувач
Вигоди компанії	Користувач отримує можливість керувати відео, які він додав до системи, можливість видалити неактивні відео або ті, які йому вже не потрібні
Частота користування	Рідко
Тригери	Користувач натискає відповідну кнопку
Передумови	Авторизований користувач, знаходиться на сторінці редагування відео
Постумови	Відео видаляється з системи
Основний розвиток	Користувач виконує запит, натискаючи відповідну кнопку на сторінці редагування відео, передаються дані, відео видаляється з системи
Альтернативні розвитки	-
Винятки	-

Таблиця 3.10 – Сценарій використання

Назва	Перегляд відфільтрованих коментарів на відео
ID	10
Опис	Для перегляду відфільтрованих коментарів на вибраному відео
Актори	Користувач
Вигоди компанії	Користувач отримує доступ до сторінки відфільтрованих коментарів, бачить результат роботи системи
Частота користування	Постійно
Тригери	Користувач натискає на назву відео на сторінці з його відео
Передумови	Авторизований користувач, знаходиться на сторінці з його відео
Постумови	Користувач отримує доступ до перегляду відфільтрованих системою коментарів
Основний розвиток	Користувач виконує запит, натискаючи на назву відео зі сторінки з усіма його відео, передаються дані, отримується інформація про відфільтровані коментарі на вибраному відео
Альтернативні розвитки	-
Винятки	-

Таблиця 3.11 – Сценарій використання

Назва	Помітка коментаря як помилково відфільтрованого
ID	11
Опис	Для подальшого донавчання моделі машинного навчання
Актори	Користувач
Вигоди компанії	Користувач допомагає зрозуміти, які коментарі були помилково відфільтровані для подальшого донавчання моделі та досягнення більш високої точності
Частота користування	Рідко
Тригери	Користувач натискає відповідну кнопку
Передумови	Користувач знаходиться на сторінці з коментарями вибраного відео
Постумови	Система отримує інформацію про помилково відфільтроване повідомлення
Основний розвиток	Користувач виконує запит, натискаючи відповідну кнопку, передаються дані, записується інформація про коментар в систему
Альтернативні розвитки	-
Винятки	-

Таблиця 3.12 – Сценарій використання

Назва	Блокування коментатора в чаті
ID	12
Опис	Для блокування особливо агресивних коментаторів в чаті
Актори	Користувач
Вигоди компанії	Користувач блокує агресивного коментатора в чаті для уникнення його подальшої активності в майбутніх чатах каналу
Частота користування	Часто
Тригери	Користувач натискає відповідну кнопку
Передумови	Користувач знаходиться на сторінці з коментарями вибраного відео
Постумови	Система блокує коментатора в чаті та на каналі користувача
Основний розвиток	Користувач виконує запит, натискаючи відповідну кнопку, передаються дані, блокується користувач, записується інформація
Альтернативні розвитки	-
Винятки	-

Таблиця 3.13 – Сценарій використання

Назва	Оплата платної підписки
ID	13
Опис	Підписка для розширення можливостей користувача
Актори	Користувач, преміум-користувач
Вигоди компанії	За рахунок платних підписок компанія отримує дохід
Частота користування	Часто
Тригери	Користувач натискає відповідну кнопку
Передумови	Авторизований користувач, знаходиться на будь-якій сторінці системи
Постумови	Користувач стає преміум-користувачем або продовжує підписку, якщо він є існуючим преміум-користувачем.
Основний розвиток	Користувач виконує запит, натискаючи відповідну кнопку, здійснюється оплата на платіжному сервісі
Альтернативні розвитки	-
Винятки	Оплата відхиляється банком, користувач передумав здійснювати покупку преміум-користувача

3.2 Розробка структурної схеми

Розроблену структурну схему можна побачити в додатку Г. Система складається з таких модулів:

- Модуль інтерфейса користувача;
- Модуль роботи з коментарями;
- Модуль оцінки коментарів;
- База даних.

Модуль інтерфейса користувача використовується для зручного доступу та використанням можливостями інтелектуальної системи для користувачів. Модуль інтерфейса є входом для роботи зі системою.

Модуль роботи з коментарями є головним модулем для роботи з системою та контентом всередині неї. Цей модуль відповідає за роботу по додаванню, редагуванню, видаленню відео, перегляду та роботі з коментарями тощо. Цей модуль взаємодіє з модулем оцінки коментарів та базою даних.

Модуль оцінки коментарів складається з сервісу оцінки коментарів на їх токсичність, що працює з використанням алгоритмів машинного навчання

Модуль бази даних зберігає інформацію про користувачів системи, додані в систему відео, коментарі тощо.

Модуль інтерфейсу користувача спілкується з іншими модулями засобами протоколів TCP/IP.

3.3 Розробка функціональної схеми

Розроблена функціональна схема зображена на додатку Д. Система має такі основні модулі:

- модуль інтерфейса користувача;
- модуль роботи з коментарями;
- модуль оцінки коментарів;
- база даних.

Модуль інтерфейса користувача є інтерфейсом, з яким взаємодіє користувач системи. Модуль складається із таких підстистем:

- управління відео;

- відображення коментарів;
- перегляд інформації про активність користувача;
- авторизація/реєстрація.

Модуль роботи з коментарями взаємодіє з базою даних для запису відфільтрованих коментарів та взаємодіє з модулем оцінки коментарів. Цей модуль складається з наступних підсистем:

- параметризація фільтрування;
- отримання всіх коментарів з чату відео;
- отримання відфільтрованих коментарів.

Модуль оцінки коментарів взаємодіє з модулем роботи з коментарями. Йому передаються коментарі, які необхідно оцінити на їх токсичність, після цього оцінки відправляються в модуль роботи з коментарями, де, якщо коментар задовільняє критеріям фільтрації, він видаляється з чату і записується в базу даних.

База даних слугує для збереження інформації про зареєстрованих користувачів для їх аутентифікації та надання доступу до роботи з системою. Також зберігається інформація про додані в системі відео, відфільтровані коментарі тощо.

3.4 Вибір та обґрунтування елементів та технологій

Для вирішення поставлених задач для серверної частини інтелектуальної системи було вирішено застосувати фреймворк для розробки веб-додатків на мові Java Spring Framework та інтегроване середовище розробки Intellij IDEA 2019. Для інтелектуальної системи фільтрації коментарів з використанням машинного навчання даний інструмент ідеально підходить, оскільки він поєднує просте написання серверної частини на мові Java з використанням фреймворку Spring Framework, створення скриптів на мові JavaScript і верстку сторінок веб-сайту на HTML і CSS з використанням фреймворку Bootstrap Framework.

Spring Framework – це універсальний відкритий фреймворк для розробки веб-додатків на мові Java. Всі вихідні файли фреймворку доступні на хостингу репозиторіїв

GitHub. Його було обрано, тому що Spring дозволяє вирішувати багато задач, з якими зіштовхуються розробники веб-додатків. Цей фреймфорк пропонує послідовну модель, що дозволяє використовувати її для великої кількості типів додатків [18].

Завдяки модульності фреймворку, всі необхідні компоненти веб-додатка можуть бути завантажені у вигляді окремих модулів через Maven, фреймворк для автоматизації складання проектів на основі опису їх структури у конфігураційному файлі з xml-подібною структурою. Maven забезпечує декларативну збірку проекту, тобто це значить, що файли проекту містять його специфікацію, а не окремі виконуючі команди. Такий підхід полегшує проектування проекту і використання залежних для нього модулів.

Головною частиною Spring фреймворку є контейнер інверсії управління, що надає засоби для конфігурації і керування Java-об'єктами за допомогою рефлексії. Контейнер інверсії управління керує життєвим циклом Java-об'єкту, викликом методів ініціалізації і конфігурації об'єктів шляхом їх зв'язки між собою. Крім цього контейнера, для реалізації системи були також використані такі модулі Spring Framework:

а) модуль Spring Data – є прошарком для роботи з базою даних і підтримує багато основних систем керування базами даних. Модуль забезпечує керування ресурсами, а саме слугує для отримання та закриття ресурсів бази даних. Також модуль дає можливість переводу винятків при доступі до даних у винятки Spring, відповідає за транзакційність в операціях з даними та дозволяє отримувати ресурси бази даних [19];

б) модуль MVC – каркас, основою якого є HTTP і сервлети і який має багато можливостей для розширення і налаштування роботи. У ньому визначені інтерфейси для роботи з функціями сучасної веб-системи. Кожен інтерфейс розроблений таким чином, щоб розробникам було просто його заново реалізовувати;

в) модуль Spring Security – відповідає за процеси аутентифікації і авторизації, який можна легко налаштовувати, а також він підтримує багато популярних протоколів та засобів авторизації в системи [20].

г) модуль тестування – цей модуль підтримує класи для написання модульних та інтеграційних тестів.

Для пришвидшення і спрощення розгортання системи було використано проект Spring Boot для спрощеного створення веб-додатків на Spring Framework. Він дозволяє створювати самостійні додатки, які зразу можна запускати та використовувати, має вбудований web-сервер Apache Tomcat, що позбавляє необхідності окремо завантажувати зібраний файл з додатком. Також Spring Boot автоматично налаштовує Spring в багатьох ситуаціях, що потребує менше роботи для розробника та власне спрощує роботу з даним фреймворком.

Інтегроване середовище розробки Intellij IDEA чудово використовується для розробки web-додатків, так як воно підтримує багато мов програмування та бібліотеки, такі як JavaScript, Spring Framework, VueJS тощо[21]. В IDEA наявне інтелектуальне автозаповнення коду, інструменти для аналізу коду, розширний функціонал покращення коду, і форматування для Java, HTML та CSS, що достатньо спрощує роботу розробника. У Intellij IDEA також присутній функціонал для роботи з web-серверами, наприклад з Apache Tomcat, присутні інструменти для роботи з базою даних та для розробки та запуску тестів.

Бібліотекою для логування модуля для роботи з коментарями було обрано Log4J, що входить до частини проекту «Apache Logging Project» [22]. Ця бібліотека дозволяє логувати декілька рівнів одночасно, залежно від їх пріоритетності:

- рівень «OFF» – максимальний можливий ранг, який виключає повністю функцію логування;
- рівень «FATAL» – призначений для фатальних помилок, які спричиняють зупинку роботу програми;
- рівень «ERROR» – використовується для критичних помилок виконання та непередбачених ситуацій в роботі;
- рівень «WARN» – використовується для показу попереджень, якщо використовується застаріле API або некоректне використання API, випадки виконання програми, які є не критичними;
- рівень «INFO» –використовується для відображення інформації, що несе інформацію про звичайний хід роботи додатку;

– рівень «DEBUG» – призначений для відображення детальної інформації під час виконання додатку;
– рівень «TRACE» – призначений для відображення найбільш детальної інформації, наприклад логування SQL запитів, під час роботи додатку.

Log4J також дозволяє створювати і настроювати власні рівні ведення логування, а також налаштовувати ведення логування в окремих файлах, зазначених у файлі конфігурації.

Slf4J обраний в якості фасаду забезпечення логування. Це бібліотека ведення логування, яка слугує для забезпечення найпростішого, але достатньо потужного фасаду для різних систем ведення логів в Java. Slf4J забезпечує простий, загальний інтерфейс для протоколів, які не залежать від конкретної реалізації, зокрема Slf4J прозоро інтегрується з обраної реалізацією Log4J. Крім того, він дозволяє інтегрувати компоненти, що мають залежність від системи ведення логів Log4J, шляхом заміни реалізації, що працює з логами цих систем в Slf4J.

3.5 Розробка бази даних

Розроблена ER-діаграма системи зображена на додатку К.

Для роботи з базою даних була вибрана система управління базами данних MySQL.

MySQL - це реляційна система управління базами даних із відкритим вихідним кодом. У теперішній час ця СУБД одна з найпопулярніших у веб-додатках – дуже багато систем використовують саме MySQL, а майже всі веб-фреймворки підтримують MySQL вже на рівні базової конфігурації без додаткових модулів.

Із переваг СУБД MySQL є просте використання, гнучкість, низька вартість відносно платних СУБД, а також масшабованість і виробництво.

MySQL дозволяє зберігати цілочисельні значення зі знаком і беззнакові, довжиною в 1, 2, 3, 4 і 8 байтів, працює зі строковими і текстовими даними фіксованої і змінної довжини, дозволяє здійснювати SQL-команди SELECT, DELETE, INSERT, REPLACE і UPDATE, забезпечує повну підтримку операторів і функцій в SELECT- і WHERE-

частинах запитів, працює з GROUP BY і ORDER BY, підтримує групові функції COUNT (), AVG (), STD (), SUM (), MAX () і MIN (), дозволяє використовувати JOIN в запитах, в тому числі LEFT OUTER JOIN і RIGHT OUTER JOIN, підтримує реплікацію, транзакції, роботу з зовнішніми ключами і каскадні зміни на їх основі, а також забезпечує багато інших функціональні можливості.

Зручність системи управління базами даних MySQL також досягається підтримкою роботи багатьох типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що мають підтримку транзакцій на рівні окремих записів. Також, тут присутні й інші типи таблиць, розроблені спільнотою розробників та користувачів.

Далі у таблиці 3.14 наведено опис таблиць розробленої бази даних.

Таблиця 3.14 — Опис таблиць бази даних

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
	id	bigint(20)	Ідентифікаційний номер користувача
	email	varchar(255)	Email користувача
	password	varchar(255)	Пароль користувача
	user_role	varchar(255)	Роль користувача в системі
	username	varchar(255)	Юзернейм користувача
	id	bigint(20)	Ідентифікаційний номер відео
	hate	double	Показник ознаки hate для фільтрації коментарів на відео

	toxic	double	Показник ознаки toxic для фільтрації коментарів на відео
	threat	double	Показник ознаки threat для фільтрації коментарів на відео
	obscene	double	Показник ознаки obscene для фільтрації коментарів на відео
	insult	double	Показник ознаки insult для фільтрації коментарів на відео
	name	varchar(255)	Назва відео
	url	varchar(255)	Посилання на відео
	is_active		Показник активності фільтрації коментарів на відео
	user_id	bigint(20)	Ідентифікаційний номер користувача, що додав відео в систему
	id	bigint(20)	Ідентифікаційний номер коментара
	content	varchar(255)	Зміст коментаря
	comment_user	varchar(255)	Юзернейм коментатора

	video_id	bigint(20)	Ідентифікаційний номер відео
	id	bigint(20)	Ідентифікаційний номер
	comments_analyzed	int(11)	Кількість проаналізованих відео

Для зв’язки розробленої моделі з базою даних у проекті вирішено використано ORM Hibernate.

Hibernate - це бібліотека, яка призначена для завдань об'єктно-реляційного відображення. Hibernate дозволяє розробнику працювати з базою даних не безпосередньо напряму, як це робиться за допомогою бібліотеки JDBC з базою даних, а за допомогою представлення таблиць баз даних у вигляді класів Java.

JPA - специфікація, яка дає можливість зберігати в зручному вигляді Java-об'єкти в базі даних. Hibernate - це одна з найпопулярніших реалізацій цієї специфікації.

Приклад зв’язку Java об’єкту Video з базою даних за допомогою використання Hibernate представлено на рисунку 3.1.

```java
@Entity
@Table(name = "videos")
public class Video {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    private String name;

    private String url;

    private Double toxic;

    private Double obscene;

    private Double threat;

    private Double insult;

    private Double hate;

    @OneToMany(mappedBy = "video", fetch = FetchType.LAZY)
    private List<Comment> comments;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "user_id")
    private User user;

    @Enumerated(value = EnumType.STRING)
    private VideoFilterStatus isActive;
}
```

Рисунок 3.1 — Приклад налаштування ORM

3.6 Діаграма послідовності

Діаграма послідовності представлена в додатку Е.

Об’єктами на діаграмі представлені:

- користувач;
- front-end;
- модуль роботи з коментарями;
- база даних;
- модуль оцінки коментарів.

Робота з інтелектуальною системою починається з ініціативи користувача – він додає відео в систему через веб-інтерфейс. Веб-інтерфейс робить запит в систему на додавання відео. Якщо не приходить відповідь з помилкою і відео активне – розпочинається фільтрація коментарів за допомогою модуля оцінки коментарів шляхом надсилання запитів з коментарями, та отриманням відповідей з їх оцінками. Якщо оцінка задовільняє критерії фільтрації – коментар видаляється з чату та дані про нього записуються в базу даних.

3.7 Розгортання системи

Діаграму розгортання системи можна побачити в додатку К.

Перший вузол діаграми – пристрій користувача. Саме на пристрої користувача, комп’ютері, телефоні або на планшеті, розгортається доступ до веб-сайту системи фільтрації коментарів, який спілкується з сервером системи. В свою чергу веб-сервер системи відповідає прийом, обробку запитів до сервісу оцінки коментарів, запису відфільтрованих коментарів в базу даних та відправку відповіді до клієнта. Спілкування з сервером сервісу оцінки коментарів відбувається шляхом відправки HTTP запитів по TCP/IP. На сервері оцінки коментарів розгорнуто Python та Tensorflow Framework для повноцінної та коректної роботи сервісу. Крім цього веб-сервер системи спілкується з сервером бази даних, на якому розгорнуто базу даних MySQL шляхом встановлення JDBC з’єднання.

3.8 Архітектура системи

Діаграма класів системи зображена на додатку Л. На рисунку 3.2 зображена архітектура проекту системи фільтрації коментарів.

Рисунок 3.2 — Архітектура проекту системи фільтрації коментарів

Щоб підвищити продуктивність системи і спростити архітектуру, був реалізований REST, архітектурний стиль взаємодії з компонентним розподіленим додатком через Інтернет. REST - це набір погоджених обмежень, що завжди враховуються при розробці розподіленої системи. Для такої системи цей підхід забезпечує простоту використання стандартизованого інтерфейсу, відкритість компонентів для можливих змін для вирішення проблем по ходу використання системи, прозорість з'єднань між компонентами системи для обслуговування клієнтів,

надійність, яка виражається в стійкості системи до системних помилок на окремих компонентах, помилок при з'єднані або даними.

По-перше, застосування RESTful підходу накладає на розробника певні обмеження. Архітектура системи має відповідати моделі «клієнт-сервер». Відділення потреб клієнтського інтерфейсу від потреб серверної частини збільшує ймовірність переносу коду клієнтського інтерфейсу на інші платформи, а спрощення серверної компоненти підвищує масштабованість. Тож цей розподіл також дозволяє частинам розвиватися незалежно.

По-друге, необхідно, щоб під час між клієнтськими запитами на сервері не зберігалася інформація про стан клієнта, тобто всі клієнтські запити мають бути побудовані так, щоб сервер отримував всю необхідну інформацію для виконання запиту. Стан сеансу зберігається на стороні клієнта. Інформація про стан сеансу може бути передана сервером в іншу службу для підтримки стабільного стану системи.

Головною умовою проектування RESTful сервісів є єдиність інтерфейсу. Таке правило дає кожному сервісу право розвиватися незалежно один від одного. В такому випадку клієнти можуть змінювати стан системи лише за допомогою дій, які динамічно визначаються на сервері [23]. Крім того, всі ресурси повинні бути ідентифіковані в запитах і бути концептуально відокремленими від даних, що повертаються клієнту.

Також RESTful сервіси мають підтримувати роботу через проміжні сервери, але клієнт не має мати можливості відрізнити роботу сервісу через окремий або центральний сервери. Використання таких проміжних серверів дозволяє реалізувати масштабованість системи за рахунок балансування навантаження запитів між різними серверами системи, а також забезпечити можливість кешування однакових запитів.

HTTP найкраще підходить для управління службовою інформацією та її передачею. Це протокол прикладного рівня, орієнтований на клієнт-серверну технологію. Протокол HTTP підтримує роботу з багатьма методами для роботи з основними методами, але для розробленого модуля роботи з коментарями використовувалися тільки такі методи як GET, POST і DELETE.

Метод GET використовується для того, щоб зробити запит на отримання даних за вхідними параметрами, що передаються в URL адресі. Цей метод використовувався для запитів для повернення системою даних, таких як повернення списку коментарів або доданих відео в систему.

Метод DELETE використовується для передачі запиту на сервер про видалення вказаного ресурсу, зазвичай по його унікальному ідентифікатору. У системі він викликається для відправки запитів для видалення відео, користувачів або коментарів.

Метод POST використовується для того, щоб зробити запит з передачою даних для збереження або відправки даних для проходження реєстрації нового користувача або для входу в систему, отримання інформації про відфільтровані коментарі, додавання нових відео, тобто для передачі великого обсягу інформації. Також метод POST використовується для отримання оцінки коментаря з сервісу оцінювання, для цього у тілі запиту вказується зміст коментаря та відправляється на сервер оцінки коментарів.

Відправки даних на сервер модуля роботи з коментарями відбувається через HTML-форми. В ці форми заповнюються текстові дані або вибираються з уже готових елементів, таких як вибір дати або радіо-кнопки. Після заповнення полей форм дані відправляються на сервер для подальшого оброблення та повернення результату. В залежності, що саме було введено або натиснуте на формах до них може бути прив’язана різна логіка роботи та поведінка веб-інтерфейсу.

Класи і функції, які відповідають за створення нових користувачів, авторизацію та інші дії з ними зображені у таблиці 3.15.

Таблиця 3.15 – Функції керування класом «User»

Клас	Функція	Опис
LoginController	login	Приймає запит з клієнту з інформацією для авторизації зареєстрованих користувачів

LoginController	register	Приймає запит з клієнту з інформацією для реєстрації нових користувачів
LoginController	successRegistration	Приймає запит переадресації на форму успішної реєстрації нових користувачів
UserService	save	Зберігання нового користувача в баз
UserService	getUserByUsername	Отримання інформації про користувача по його логіну
UserService	getUserByEmail	Отримання інформації про користувача по
UserService	getUserById	Отримання інформації про користувача по його ідентифікатору
UserService	delete	Видалення користувача із системи

Таблиця 3.16 – Функції керування класом «Video»

Клас	Функція	Опис
VideoController	addVideo	Приймає запит з клієнту з інформацією для додавання нового відео в систему

VideoController	editVideo	Приймає запит з клієнту з інформацією для редагування існуючого відео в системі
VideoController	deleteVideo	Приймає запит з клієнту з інформацією для видалення існуючого відео з системи
VideoController	videos	Приймає запит з клієнту з інформацією для отримання списку всіх відео
VideoService	save	Зберігання інформації про нове відео в системі
VideoService	update	Редагування існуючого відео в системі

Продовження таблиці 3.16

Клас	Функція	Опис
VideoService	delete	Видалення існуючого відео з системи
VideoService	getVideoById	Отримання інформації про

		існуюче відео в системі по його ідентифікатору
VideoService	getVideosByUserId	Отримання списку всіх відео, доданих користувачем по його ідентифікатору
VideoService	getAll	Отримання списку всіх відео, що додані в систему
VideoService	countActiveVideosByUserId	Отримання інформації щодо кількості активних відео з активною фільтрацією коментарів по ідентифікатору користувача

Таблиця 3.17 – Функції керування класом «Comment»

Клас	Функція	Опис
CommentController	comments	Приймає запит з клієнту з інформацією для отримання всіх

		відфільтрованих коментарів на відео
CommentController	markAsOk	Приймає запит з клієнту з інформацією для відмітки коментаря як помилково відфільтрованого
CommentController	ban	Приймає запит з клієнту для блокування коментатора в чаті
CommentController	delete	Приймає запит з клієнту для видалення коментаря з чату
CommentService	save	Зберігання нового відфільтрованого коментаря в систему
CommentService	getComments	Отримати всі відфільтровані коментарі по ідентифікатору відео

Продовження таблиці 3.17

Клас	Функція	Опис
CommentService	delete	Видалити коментар з чату
CommentService	getCommentsCountByUserId	Отримати кількість всіх відфільтрованих

		коментарів по ідентифікатору користувача
CommentService	incrementAnalyzedComments	Інкрементація лічильника проаналізованих коментарів в системі
CommentService	getAnalyzedCommentsCount	Отримати кількість проаналізованих коментарів в системі
CommentService	startCommentsFiltering	Розпочати фільтрацію коментарів на відео
CommentService	banUser	Заблокувати користувача в чаті
CommentService	markAsOk	Відмітити коментар як помилково відфільтрований

3.9 Тестування системи

Для підтвердження коректної роботи розробленої інтелектуальної системи фільтрації коментарів з використанням машинного навчання було проведено тестування системи.

У таблиці 3.15 наведено тестовий сценарій управління користувачами, авторизації та менеджменту ролями в системі.

Результатами тестових сценаріїв можуть бути такі варіанти як: «Пройшов», «Не пройшов» або «Заблокований».

Таблиця 3.15 — Тестовий сценарій 1.

Назва:	Тестовий сценарій 1	
Функція:	Керування користувачами, авторизація та менеджмент ролей	
Дія	**Очікуваний результат**	**Результат тесту:**
Передумова:	Передумов немає	
Кроки тестування:		
Додати роль	Роль додано до бази даних	Пройшов
Зареєструвати користувача	Користувача додано до БД	Пройшов
Авторизувати користувача	Користувачу надано доступ до функцій системи	Пройшов
Вийти	Користувач втрачає доступ	Пройшов
Видалити користувача	Користувача видалено з БД	Пройшов
Видалити роль	Роль видалено з БД	Пройшов
Постумова:	Постумов немає	

Тестовий сценарій 1 пройшов успішно. Був протестований такий функціонал як реєстрація і авторизація користувача, додавання первісних ролей в системі, керування користувачами.

Постумов в даному сценарії не передбачається.

У таблиці 3.16 наведено тестовий сценарій додавання відео, управління доданими відео в системі тощо.

Таблиця 3.16 — Тестовий сценарій 2.

Назва:	Тестовий сценарій 2	
Функція:	Керування відео в системі	
Дія	**Очікуваний результат**	**Результат тесту:**
Передумова:		
Користувач авторизується	Користувача надано доступ	Пройшов
Кроки тестування:		

Додати відео	Відео додано до бази даних	Пройшов
Редагувати відео	Відео відредаговано в БД	Пройшов
Видалити відео	Відео видалене з бази даних	Пройшов
Отримати список всіх доданих відео	Виводиться список всіх відео користувача	Пройшов
Додати відео, яке вже є в системі	Виводиться повідомлення про повторне додавання відео в систему	Пройшов
Постумова:	Постумов немає	

Тестовий сценарій 2 пройшов успішно. Був протестований такий функціонал як додавання нових відео, редагування та видалення відео з системи, а також було здійснена спроба повторного додавання відео в систему, яке вже там присутнє.

Постумов в даному сценарії не передбачається.

У таблиці 3.17 наведено тестовий сценарій роботи з коментарями в інтелектуальній системі фільтрації коментарів.

Таблиця 3.17 — Тестовий сценарій 3.

Назва:	Тестовий сценарій 3	
Функція:	Робота з коментарями в системі	
Дія	**Очікуваний результат**	**Результат тесту:**
Передумова:		
Користувач авторизується	Користувача надано доступ	Пройшов
Додається відео в систему	Відео додано в систему	Пройшов
Кроки тестування:		
Перегляд відфільтрованих коментарів на відео	Отримання списку всіх відфільтрованих коментарів на відео	Пройшов
Видалення коментаря з чату	Коментар видалено з чату	Пройшов
Блокування коментатора в чаті	Коментатор заблокований в чаті	Пройшов
Помітити коментар як помилково відфільтроване	Зміна інформації про коментар в базі даних	Пройшов

Постумова:	Постумов немає

Тестовий сценарій 3 пройшов успішно. Був протестований такий функціонал як перегляд коментарів на відео, видалення коментаря з чату, блокування коментатора.

Постумов в даному сценарії не передбачається.

Покриття тестовими сценаріями варіантів використання системи наведене в таблиці 3.18. З даної таблиці видно, що деякі тести покривають відразу декілька варіантів використання.

Таблиця 3.25 – Покриття тестовими сценаріями варіантів використання

Варіант / Тест	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	+	+	+	+	+	+	+	+						
2											+	+	+	+
3														
4									+	+				

3.10 Висновки до розділу

В даному розділі були розроблені та описані всі сценарії використання інтелектуальної системи, розроблена структурна та функціональна схеми. Для обраних технологій при розробці системи були описані їх переваги і чому саме вони були вибрані. Також, була описана реалізація схеми бази даних та власне архітектура всієї системи в цілому. Вся система була покрита юніт-тестами та було проведено інтеграційне тестування, що було підтверджено описаними тестовими сценаріями.

4 РОЗРОБЛЕННЯ ІНТЕРФЕЙСА КОРИСТУВАЧА

Для розробки інтерфейсу інтелектуальної системи фільтрації коментарів з використанням машинного навчання були використані Java Server Pages - технологія, яка дозволяє створювати динамічні веб-сторінки. Спочатку JSP (разом з сервлетами) на зорі розвитку Java EE були домінуючим підходом до веб-розробки на мові Java, проте JSP продовжують широко використовуватися й зараз.

По суті Java Server Page або JSP є html-кодом зі вставками коду Java. У той же час сторінки JSP - це не зовсім стандартні html-сторінки. Коли приходить запит до певної JSP-сторінки, то сервер обробляє її, генерує з неї код html код і відправляє його клієнту. В результаті користувач після звернення до сторінки JSP бачить в своєму браузері звичайну html-сторінку.

Як і звичайні статичні веб-сторінки, файли JSP необхідно розміщувати на веб-сервері, до якого звичайні користувачі можуть звертатися по протоколу http або https, наприклад, набираючи в адресному рядку браузера потрібну адресу. Однак щоб сервер міг обробляти файли JSP, він повинен використовувати движок JSP (JSP engine), який також називають JSP-контейнером. Є безліч движків JSP, і всі вони реалізують одну й ту ж специфікацію і в цілому працюють однаково. Однак тим не менше при перенесенні коду з одного веб-сервера на інший іноді можуть знадобитися невеликі зміни.

Для власної системи було вирішено використати Apache Tomcat, який одначасно виконує роль і веб-сервера, і контейнера сервлетів та JSP.

Для дизайну сторінок був використаний Bootstrap Framework, що дозволив значно спростить та скоротить час на верстку веб-сторінок. Bootstrap - це фреймворк, який допомагає швидше і простіше розробляти веб-сайти. Він включає в себе шаблони дизайну на основі HTML та CSS для шрифтів, форм, кнопок, таблиць, навігації, модалів, каруселей зображень тощо. Він також надає підтримку плагінів JavaScript. Використовуючи базові стилі та скрипти Bootstrap Framework були розроблені інтерфейси основних сторінок сервісу та імплементовані в JSP.

На рисунках 4.1-4.8 зображені розроблені інтерфейси системи фільтрації коментарів.

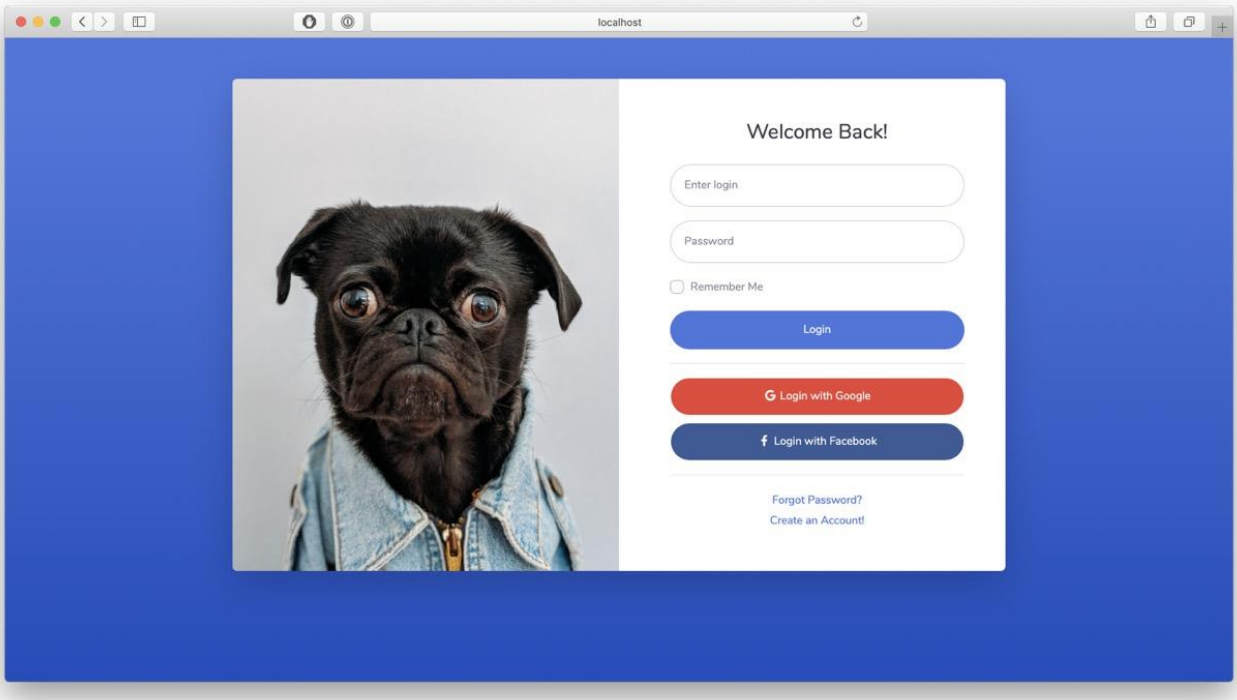

Рисунок 4.1 — Інтерфейс сторінки авторизації.

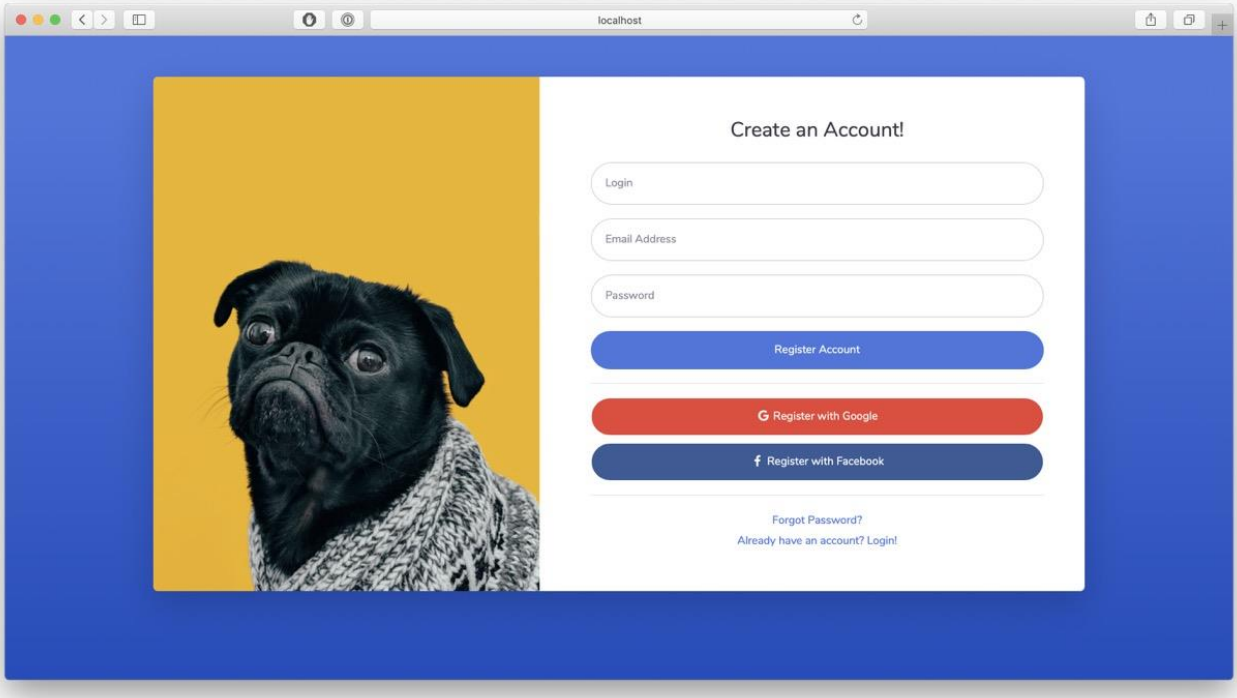

Рисунок 4.2 — Інтерфейс сторінки реєстрації.

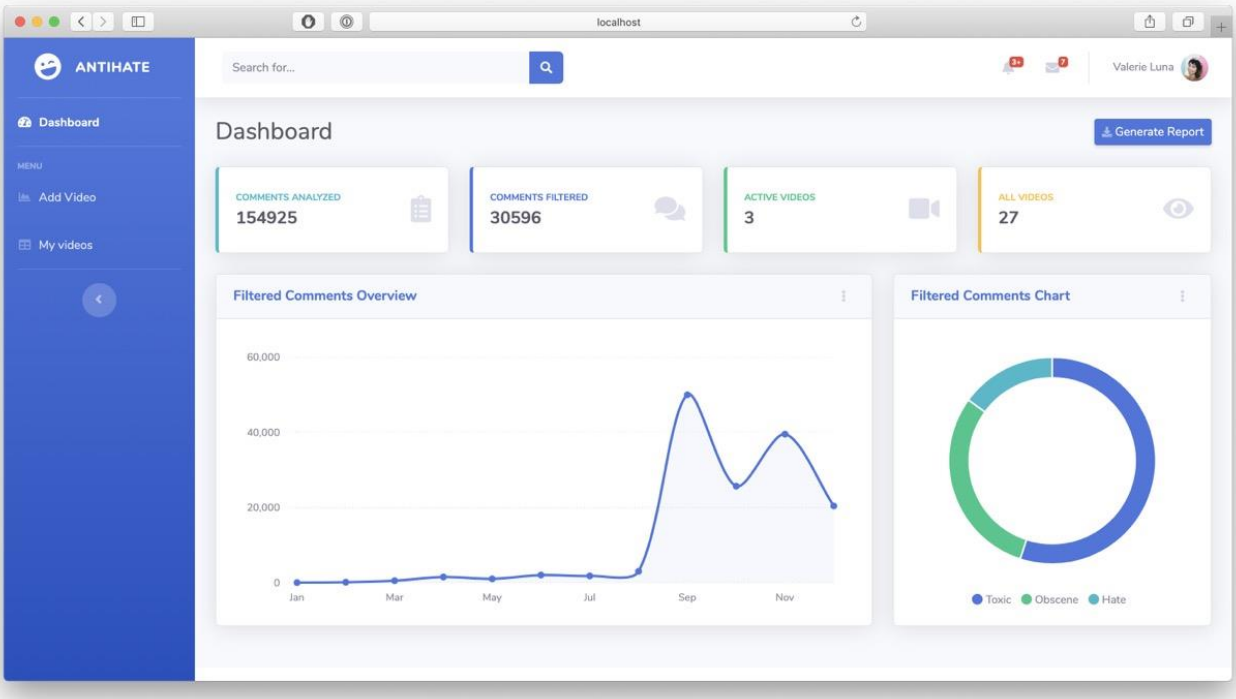

Рисунок 4.3 — Інтерфейс головної сторінки.

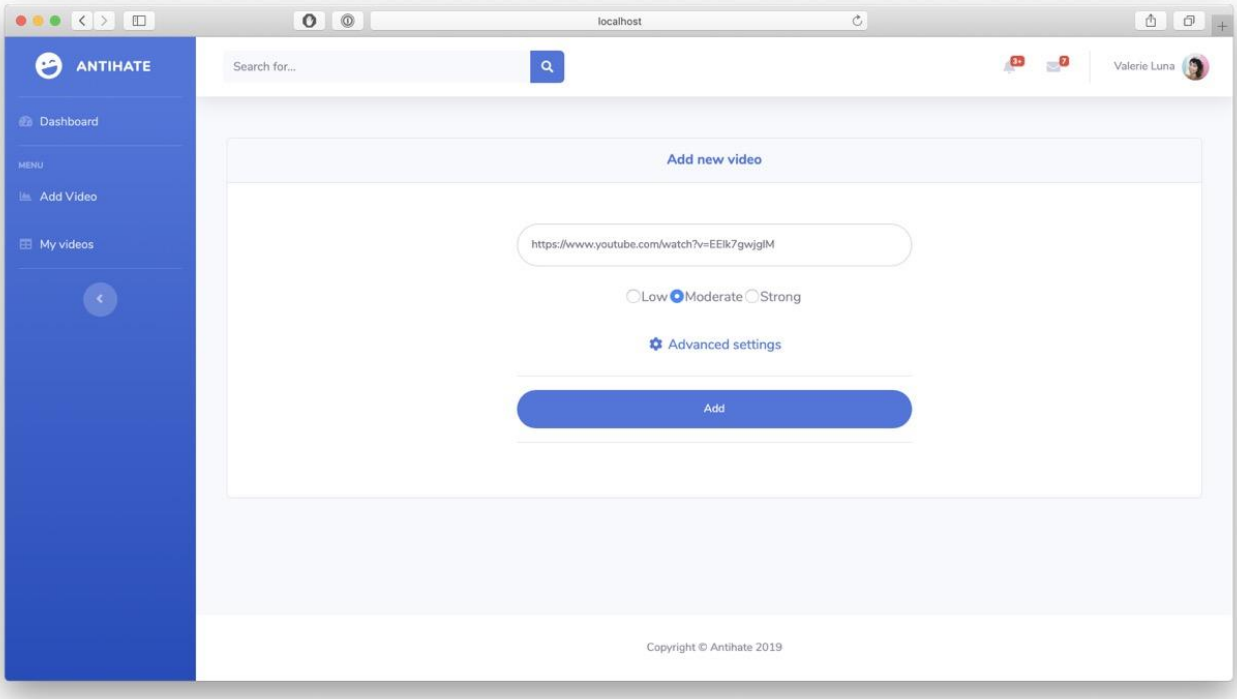

Рисунок 4.4 — Інтерфейс додавання нового відео для фільтрації коментарів.

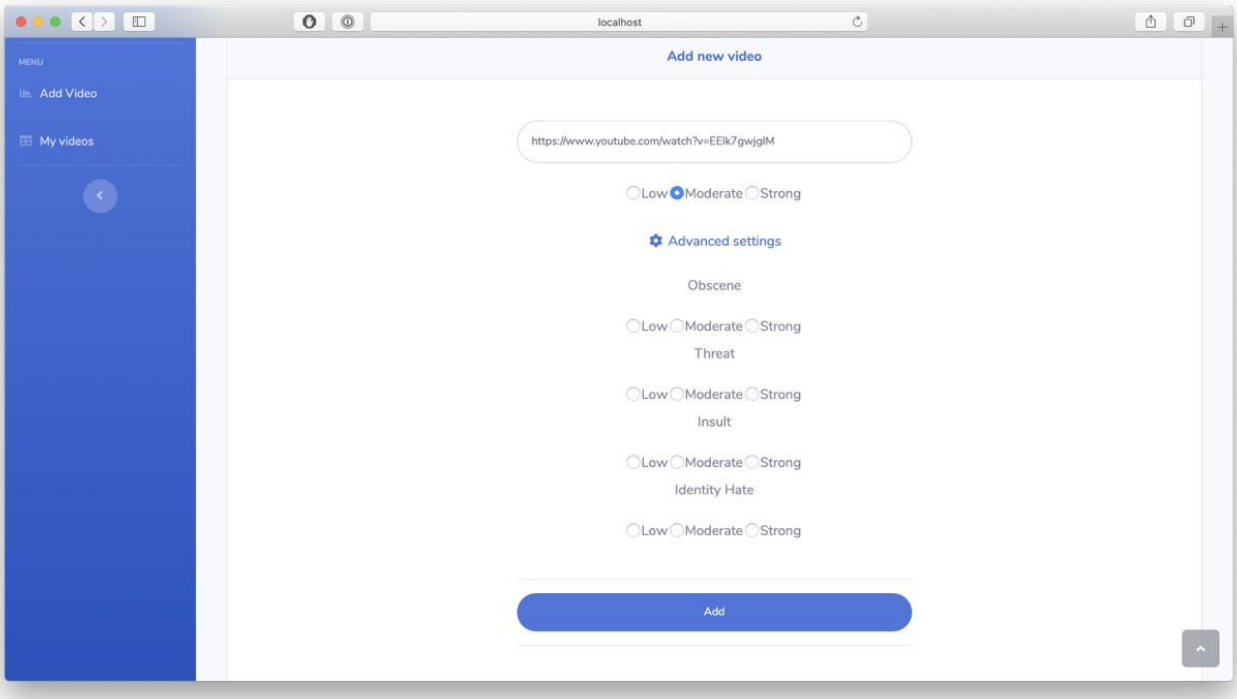

Рисунок 4.5 — Розширений інтерфейс додавання нового відео для фільтрації коментарів.

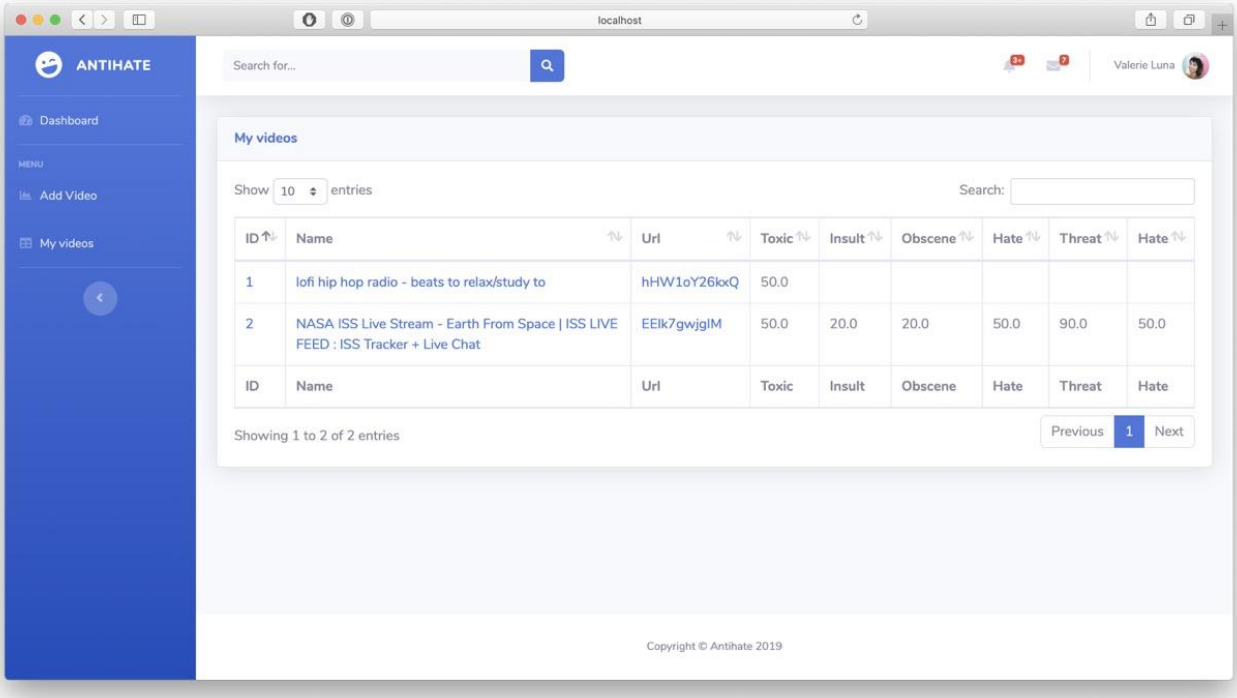

Рисунок 4.6 — Інтерфейс зі списком всіх відео, на яких фільтруються коментарі.

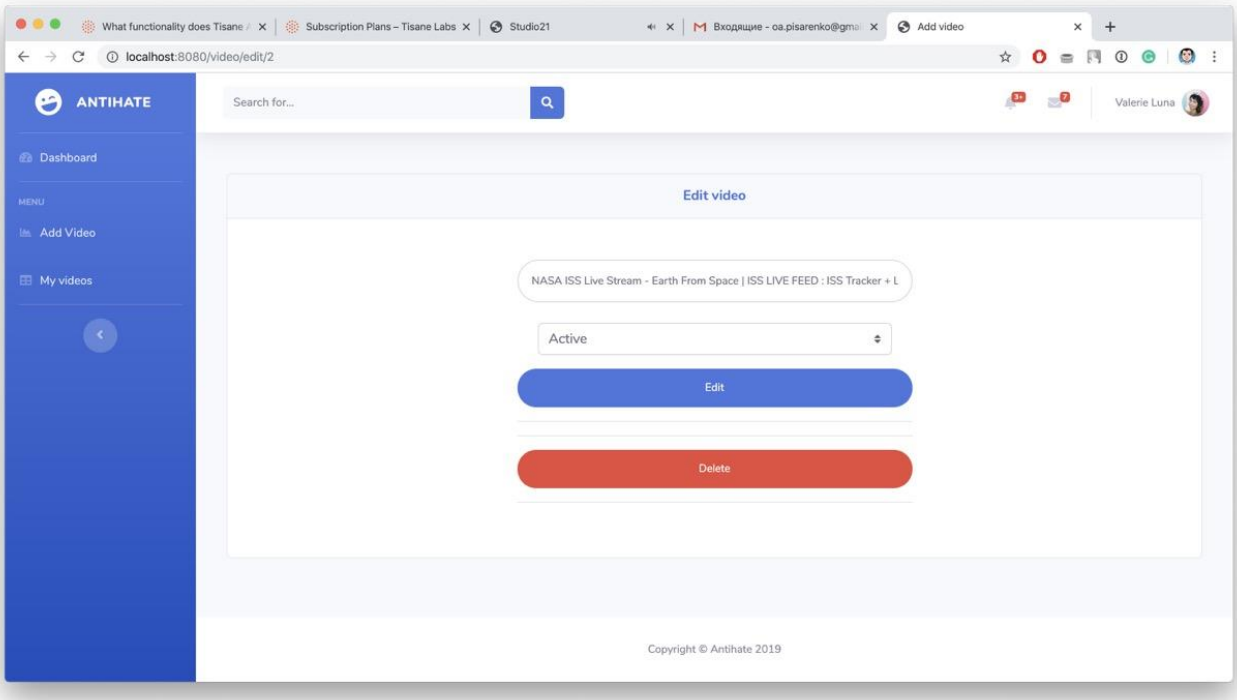

Рисунок 4.7 — Інтерфейс редагування та видалення відео.

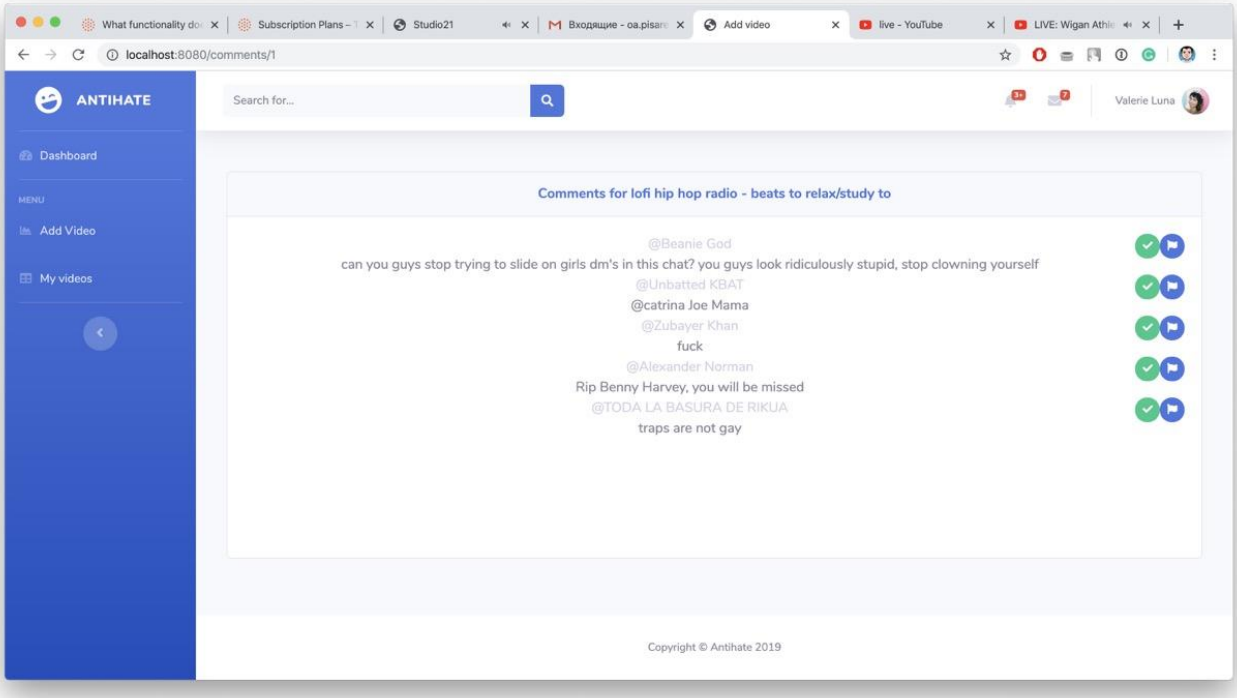

Рисунок 4.8 — Інтерфейс редагування та видалення відео.

4.1 Висновки до розділу

В розділі було описані технології, використані при розробці інтерфейсу користувача для інтелектуальної системи фільтрації коментарів з використанням машинного навчання. Для розбробки інтерфейсу було вирішено використати фреймворк Bootstrap Framework для роботи з візуальними компонентами сторінок та Java Servlets для серверного рендерингу сторінок. Обрані інструменти та технології дозволили розробити сучасний та зручний веб-інтерфейс сторінок системи. Результати розробки інтерфейсу були представлені на скриншотах.

5 ОЦІНКА ЯКОСТІ МОДЕЛІ

Для оцінки якості розробленої моделі нейронної мережі для оцінки коментарів на вміст токсичності було проведено оцінку тестової вибірки коментарів, що складалась з понад 150 тисяч коментарів різного змісту. Для оцінки точності оцінки коментарів розробленої моделі нейронної мережі були вибрані розглянуті раніше класичні алгоритми машинного навчання: логістична регресія, градієнтний бустинг та алгоритм Random Forest.

5.1 Порівняння роботи алгоритмів машинного навчання

Для кожного з алгоритмів навчання були побудовані ROC криві – графік, що дозволяє оцінити якість бінарної класифікації, відображає співвідношення між часткою об'єктів від загальної кількості носіїв ознаки, вірно класифікованих до загальної кількості об'єктів, що не несуть ознаки, помилково класифікованих, як такі, що мають ознаку.

Результат оцінки тестової вибірки коментарів алгоритмом Random Forest та його графік ROC кривої зображений на рисунку 5.1.

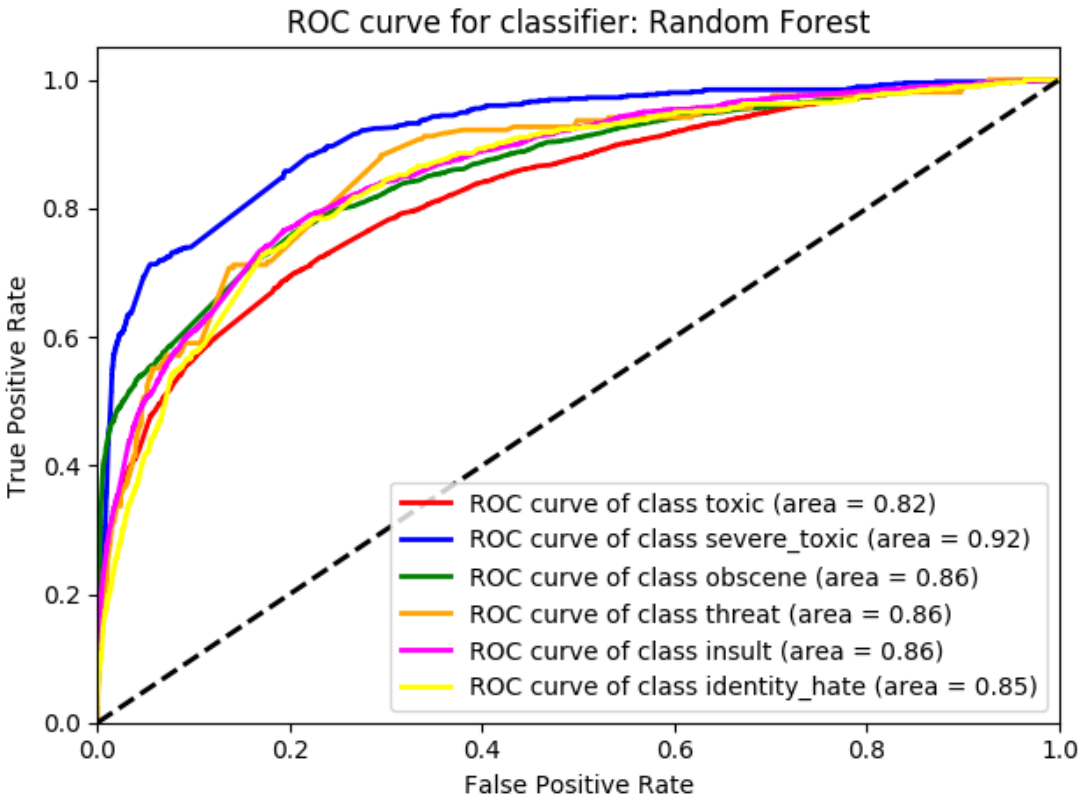

Рисунок 5.1 — Графік ROC кривої для алгоритму Random Forest.

Результат оцінки тестової вибірки коментарів алгоритмом градієнтний бустинг та його графік ROC кривої зображений на рисунку 5.2.

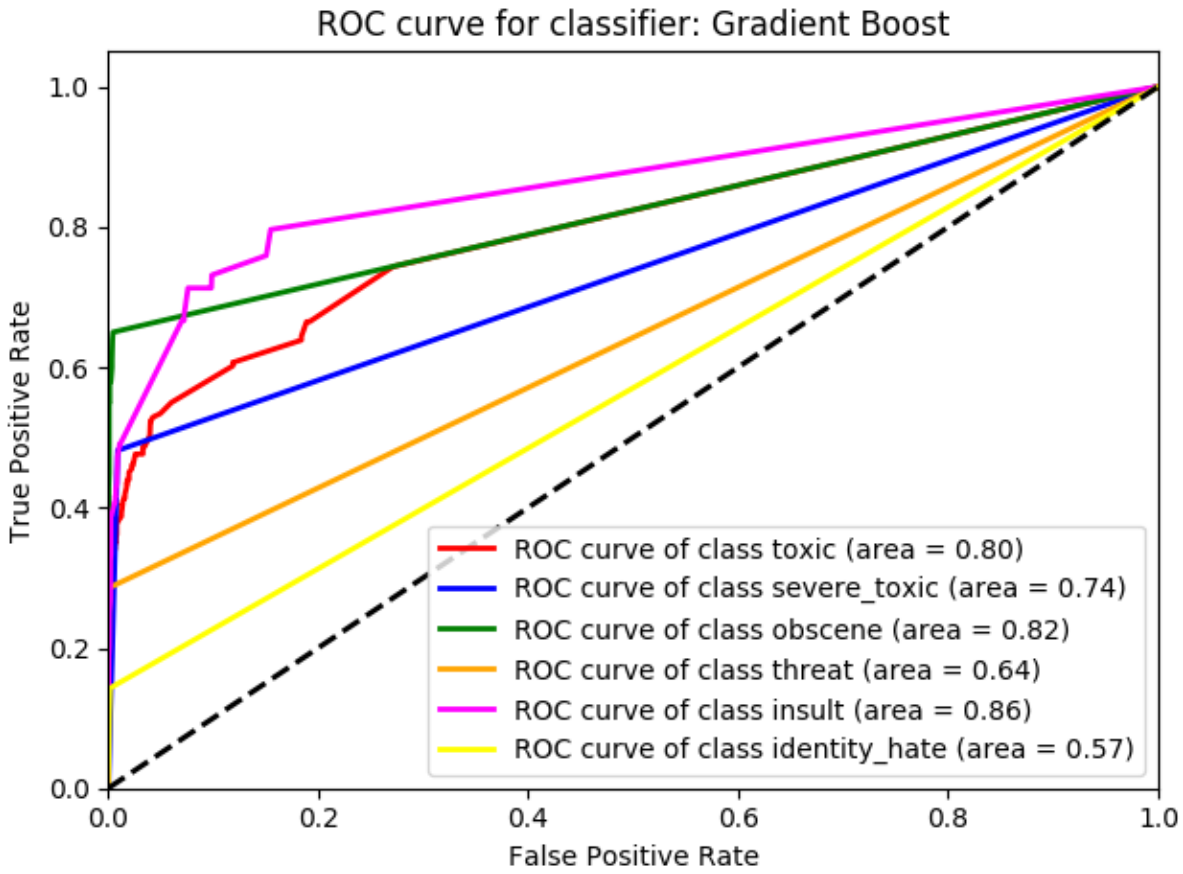

Рисунок 5.2 — Графік ROC кривої для алгоритму градієнтний бустинг.

Результат оцінки тестової вибірки коментарів алгоритмом логістична регресія та його графік ROC кривої зображений на рисунку 5.3.

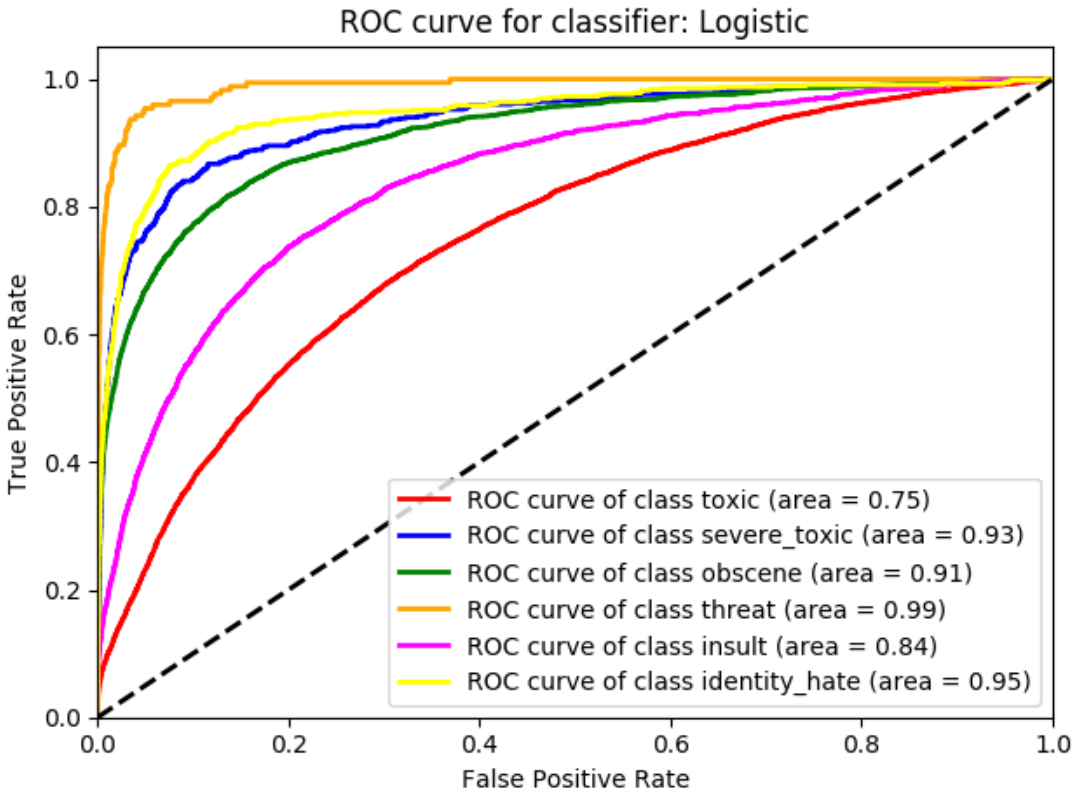

Рисунок 5.3 — Графік ROC кривої для алгоритму логістична регресія.

Розроблена власна модель нейронної мережі була також протестована на тестовій вибірці коментарів. Результат роботи оцінки коментарів можна побачити на рисунку 5.4.

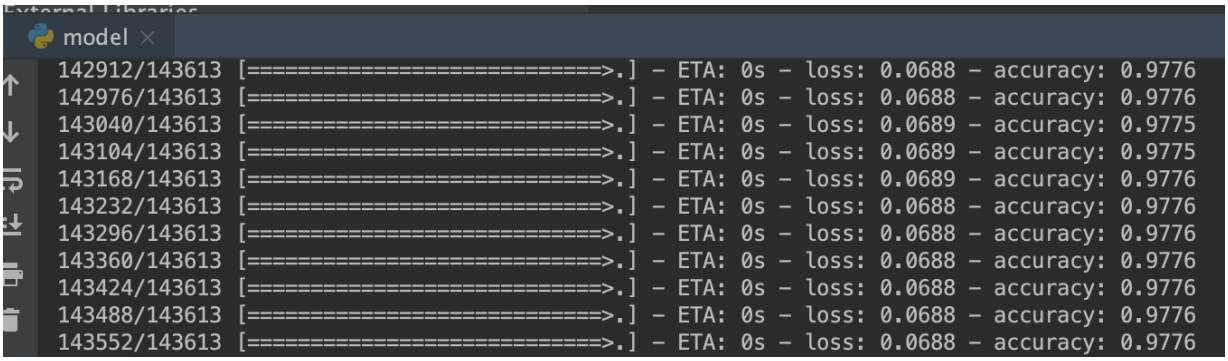

Рисунок 5.4 — Результат оцінки коментарів розробленою моделлю нейронної мережі.

5.2 Висновки до розділу

Приклад роботи кожного з алгоритмів був продемонстрований на графіках ROC кривих для кожної з ознак, за якими класифікується коментар.

Як видно з графіків, кожен з алгоритмів показав різну точність на тих чи інших ознаках, але середнє значення точності залишається на рівні 80-85%.

Розроблена модель згорткової нейронної мережі для оцінки коментарів на токсичність показала точність в 97.76%, що є набагато кращим результатом за роботу класичних алгоритмів машинного навчання.

6 РОЗРОБКА СТАРТАП ПРОЕКТУ

Стартап – це форма підприємства, яка шукає бізнес-модель, що задовольняє потреби, як споживачів, так і самої компанії. Бізнес-моделлю є процес створення цінності компанії шляхом продажу продуктів чи послуг. В стартапі розрізняють масштабовану відновлювану бізнес-модель, оскільки стартап – це невизначене підприємство і тому бізнес-модель має відповідати вказаним критеріям.

6.1 Опис ідеї проекту

Інтелектуальна система фільтрації коментарів з використанням машинного навчання функціонує як самостійна одиниця у вигляді сервісу, що інтегрується з існуючими відео-платформами, так і у вигляді доступу до API, що дає змогу отримати оцінку текстового коментаря про його токсичність. Опис ідеї стартапу наведено в таблиці 5.1.

Таблиця 5.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрями застосування	Вигоди для користувача
Інтелектуальна система фільтрації коментарів з використанням машинного навчання використовується авторами каналів на відео-сервісах для покращення онлайн-спілкування їх підписників.	Фільтрація коментарів	Покращення змістовності онлайн-спілкування, фільтрація агресивних та токсичних коментарів в чатах, що можуть призвести до образ з боку певних осіб або соціальних груп.

На ринку присутні декілька конкурентів, що пропонують схожий функціонал. Основним недоліком їх впровадження є відсутність інтеграції з існуючими популярними відео-сервісами.

Аналіз потенційних техніко-економічних переваг ідеї наведено в таблиці 5.2.

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		IC Antihate	Perspective API	Tisane API			
1	Вартість	Середня вартість	Безкоштовно	Висока вартість			+
2	Аналіз тексту за ознаками	Так	Так	Так		+	
3	Інтеграція з відео-сервісами	Так	Ні	Ні			+
4	Автоматична фільтрація коментарів	Так	Ні	Ні			+
5	Наявність клієнтської бази	Ні	Так	Так	+		

5.2 Технологічний аудит ідеї проекту

Для реалізації системи необхідно провести технічний аудит, що наведено в таблиці 5.3.

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Back-end частина системи фільтрації коментарів	Java, Spring Boot, Spring Security, Spring Data, YouTube API	Наявні, дороблювати не потрібно.	Доступні, відкритий доступ
2	Сервіс оцінки коментарів з використанням машинного навчання	Keras, Tensorflow, Python, Flask	Наявні, дороблювати не потрібно.	Доступні, відкритий доступ
3	Front-end частина системи фільтрації коментарів	Javascript, jQuery, Bootstrap Framework	Наявні, дороблювати не потрібно.	Доступні, відкритий доступ
4	База даних	MySQL	Наявні, дороблювати не потрібно.	Доступні, відкритий доступ
Для реалізації системи наявні та доступні всі необхідні технології та реалізації: Java, Spring Framework, Python, Tensorflow, MySQL.				

5.3 Аналіз ринкових можливостей запуску

Аналіз попиту наведений в таблиці 5.4.

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	Видимі конкуренти на ринку відсутні, наявні послуги для часткових рішень, що не задовольняють повністю проблеми клієнтів. На світовому ринку їх до 5
2	Загальний обсяг продаж, грн/ум.од	1 млрд. ум. од.
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Без обмежень
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	Дуже висока, під п’ятидесяти до восьмидесяти відсотів за одну одиницю товару.

Визначення потенційних клієнтів наведено в таблиці 5.5.

Таблиця 5.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Фільтрація коментарів на відео-сервісах	Автори онлайн відео-трансляцій, що хочуть отримати конструктивне спілкування в чаті	Потребують фільтрацію коментарів за різними ознаками	Легкість користування, інтуїтивний інтерфейс, вартість, автоматизація
2	Фільтрація коментарів на онлайн-ресурсах	Автори онлайн-ресурсів, що хочуть отримати конструктивне спілкування в коментарях	Потребують фільтрацію коментарів за різними ознаками, доступ по API	Зручна, інтуїтивна система, легкість інтеграції на власний ресурс

Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища: складаються таблиця 5.7 та факторів, що сприяють ринковому впровадженню проекту, та таблиця 5.6 факторів, що йому перешкоджають.

Таблиця 5.6 – Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Поява нових конкурентів	Поява конкурентів з подібними рішеннями	зміна ціни; вирішення інших функцій;
2	Стан економіки	Економіка України є нестабільною і постійно падає, що знижує платоспроможність компаній	Оптимізація витрат та перевезення бізнесу за кордон

До позитивних факторів можна віднести незайнятий ринок аналізу даних клієнтів.

Таблиця 5.7 – Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1.	Вільний ринок України	На ринку немає готових рішень	Простота виходу на ринок
2.	Неусвідомленість компаній	Багато компаній до сих пір не усвідомлює значущість інтелектуальних систем фільтрації коментарів	Реклама, прямі продажі

Надалі проводиться аналіз пропозиції: визначаються загальні риси конкуренції на ринку, що показано в таблиці 5.8.

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
Тип конкуренції: ринок України – монополістична, світовий – чиста	На ринку України відсутні рішення ІС фільтрації коментарів	Максимально розширювати межі та захоплювати сегменти компаній, що потребують даний продукт
Рівень боротьби: національний	Ринок фільтрації коментарів є по всій території України	Використання продукту на території України
За галузевою ознакою: внутрішньо-галузева	Конкуренція тільки в системах фільтрації коментарів	Моніторинг ринку
За характером пере-ваг: нецінова	ІС фільтрації коментарів має перевагу в автоматизації та інтеграції з відео-сервісами	Розширення інтеграцій з іншими популярними сервісами для роботи з ними «з коробки»

Після аналізу конкуренції проводиться більш детальний аналіз умов конкуренції в галузі (за моделлю 5 сил М. Портера), що наведено у таблиці 5.9, з якої випливає що є велика можливість виходу на ринок, пропонуючи унікальні продукти та послуги, а оскільки продуктом є система з інноваційним рішенням, конкурувати буде значно легше, а також беручи до уваги те, що система є простою для впровадження, підтримує масштабованість, вона уде задовольняти потреби більшості компаній.

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Прямі конкуренти на ринку України відсутні, на світовому частково – Perspective, Tisane. Непрямі конкуренти: Системи аналізу текстових даних в соц. мережах	Вихід світових лідерів на ринок України	Постачальники відсутні	Обмеження платоспроможністю	Товарозамінники відсутні
Висновки	Конкуренція несформована	Perspective API, Tisane API	Постачальники відсутні	Можлива відмова від рішення	Товарозамінники відсутні.

Далі необхідно проаналізувати та визначити фактори конкурентоспроможності компанії, що наведені в таблиці 5.10.

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспро-можності	Обґрунтування (наведення чинників, що роб-лять фактор для порівняння конкурентних проектів значущим)
1	Універсальність	Система може бути впроваджена в будь-який веб-ресурс з коментарями через API, так і використовуватись самостійно через веб-інтерфейс автоматично для роботи з відео-сервісами.
2	Ціна	По відношенню до конкурентів, дана система маю меншу собівартість, а це означає, що вартість використання системи є безкоштовною або мінімальною
3	Інновації та технології	Система має інновації в таким місцях, як аналіз коментарів з використанням машинного навчання, автоматизація фільтрування коментів, інтеграція з відео-сервісами.
4	Масштабованість	Для забезпечення конкурентоспроможності система має бути масштабована і готова до змін та розміру бізнесу клієнта.
5	Швидкість впровадження	Для бізнесу завжди важлива швидкість, щоб бути попереду конкурентів

За визначеними факторами конкурентоспроможності було проведено аналіз сильних та слабких сторін стартапу.

Таблиця 5.11 – Порівняльний аналіз сильних та слабких сторін «Інтелектуальна система фільтрації коментарів з використанням машинного навчання»

№	Фактор конкурентоспроможності	Рейтинг товарів-конкурентів у порівнянні						
		-3	*-2*	*-1*	*0*	*1*	*2*	*3*
1	Універсальність		✓					
2	Ціна			✓				
3	Інновації та технології				✓			
4	Масштабованість				✓			
5	Швидкість			✓				

Було розроблено альтернативи ринкової поведінки для виведення стартапу на ринок. На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. SWOT аналіз складається з аналізу сильних, слабких сторін, загроз та можливостей. SWOT наведений у таблиці 5.12.

Таблиця 5.12 – SWOT-аналіз стартап-проекту

Сильні сторони: 1. універсальність; 2. технологія, інновація	Слабкі сторони: 1. ІТ фахівці. 2. досвід
Можливості: 1. несформована конкуренція; 2. попит на системи фільтрації коментарів серед авторів контенту	Загрози: 1. поява нових конкурентів; 2. стан економіки середовища

Альтернативи ринкового впровадження наведені в таблиці 5.13.

З означених альтернатив обирається та, для якої:

а) отримання ресурсів є більш простим та ймовірним; б) строки реалізації – більш стислими.

Таблиця 5.13 – Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Спільні продажі	Дуже висока. У випадку партнера-гіганта	1 рік
2	Індивідуальні продажі	Середня	1,5-2 роки

5.4 Розроблення ринкової стратегії проекту

Для отримання продаж необхідно визначити цільову аудиторію та групу потенційних споживачів. Опис таких груп наведеного в таблиці 5.14. Аналіз груп дає можливість сформувати стратегію охоплення ринку для виходу на нього.

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Самостійні автори відео контенту	Готові	Зростаючий попит	Низька	Низький бар’єр входу

Продовження таблиці 5.14

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
2	Власники сайтів з можливістю коментування	Готові	Зростаючий попит	Низька	Низький бар'єр входу
3	Великі медіа текстового та відео контенту	Не готові	Низький попит	Низька	Високий бар'єр входу
Які цільові групи обрано: Самостійні автори відео контенту, власники сайтів з можливістю коментування					

Для виходу на ринок необхідно визначити базову стратегію розвитку, яка наведена в таблиці 5.15.

Таблиця 5.15 – Визначення базової стратегії розвитку

№	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Надання платформи малому та середньому бізнесу	Вибіркови й розподіл	Здатність протистояти прямим конкурентам Низькі витрати Ефективна співпраця	Стратегія диференціації

За вимогами споживачів у таблиці 5.16 розробляється стратегія конкурентної поведінки. В таблиці 5.17 визначена стратегія позиціонування.

Таблиця 5.16 – Визначення базової стратегії конкурентної поведінки

№	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
1.	Так	Шукати нових	Ні	Стратегія спрямованості

Таблиця 5.16 – Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	1. Простота використання 2. Ціна 3. Наявність бажаного функціоналу 4. Масштабованість 5. Інтеграція 6. Якість	Стратегія спрямованості	1. Цінова перевага 2. Іноваційне рішення 3. Простота розгортання 4. Простота впровадження 5. Інтеграція 6. Масшабованість	1. Підвищення прибутку 2. Збільшення клієнтів 3. Оптимізація витрат

5.5 Розроблення маркетингової програми

Маркетингова програма – це комплекс завдань, що пов’язані для планомірного здійснення, і адресних заходів соціального, економічного, науково-технічного, виробничого, організаційного характеру з визначенням ресурсів, що використовуються. Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у таблиці 5.18 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Збільшення прибутку	Спрямування ресурсів лише на ті групи клієнтів, які більше приносять прибутку	
2	Оптимізація витрат	Зменшення витрат на пошук прихованих груп клієнтів	
3	Підвищення якості оцінки коментарів	Підвищення якості роботи інтелектуальної систе ми	Якість надання послуг

Наступним кроком є розробка трирівневої маркетингової моділ товару, для чого уточнюється ідея послуг та продуктів, що наведено таблиця 5.19.

Таблиця 5.19 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>І. Товар за задумом</td><td colspan="3">Продукт у вигляді інтелектуальної системи фільтрації коментарів з використанням машинного навчання з інтеграцією з популярними відео-сервісами.</td></tr>
<tr><td rowspan="5"></td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>Кількість</td><td></td><td>1 шт.</td></tr>
<tr><td colspan="3">Якість: стандарти якості постачання програмних продуктів</td></tr>
<tr><td colspan="3">Пакування: через веб-інтерфейс</td></tr>
<tr><td colspan="3">Марка: Antihate</td></tr>
<tr><td rowspan="2"></td><td colspan="3">Програмний продукт</td></tr>
<tr><td colspan="3">Програмний продукт, технічна підтримка та підписка на розширений функціонал</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: захист інтелектуальної власності</td></tr>
</table>

Наступним кроком є визначення цінових меж на потенційний товар, яке передбачає аналіз ціни на товари-аналоги, а також аналіз рівня доходів цільової групи споживачів, що наведене в таблиці 5.20.

Таблиця 5.20 – Визначення меж встановлення ціни

№	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1.	Від 99$ в місяць	Від 99$ в місяць	Від 2000$ в місяць	Нижня:0$ Верхня: 200$

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення, таблиця 5.21:

− проводити збут власними силами або залучати сторонніх посеред-ників (власна або залучена система збуту);

− вибір та обґрунтування оптимальної глибини каналу збуту;

− вибір та обґрунтування виду посередників.

Таблиця 5.21 – Формування системи збуту

№	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має викону- вати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Оплата підписки іде через веб-сайт	Доступ до додаткових функцій сервісу	Канал одного рівня	Селективна з використанням комбінованого каналу збуту

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів, що наведено в таблиці 5.22.

Таблиця 5.22 – Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1.	Автоматичне фільтрування коментарів на прямих трансляціях	Прямі офіційні	Доступність та об’єктивність інформації про фірму і товар Унікальність послуги	Розширення аудиторії	Вказати на використання інновацій та сучасних технологій та кейси вирішення проблем

5.6 Висновки до розділу

В цьому розділі було проведено маркетинговий аналіз з метою визначення можливості та доцільності виходу на ринок з проектом інтелектуальної системи фільтрації коментарів з використанням машинного навчання.

В результаті дослідження було встановлено, що існує велика ймовірність успішної комерціалізації системи на ринку. Виходячи з того, що на ринку України прямих конкурентів немає, можна стверджувати, про легкий старт. Однак необхідно брати до уваги економічний стан в країні та можливість входу зарубіжних систем на ринок України. В результаті проведення маркетингового дослідження було обрано стратегії для того, щоб стартап був конкурентоспроможний, а також проаналізовано сильні та слабкі сторони, визначено цільову аудиторію та потенційних клієнтів.

Сильними сторонами продукту є іноваційність, простота впровадження та інтеграції та масштабованість. Виходячи з цього можна стверджувати про доцільність подальшої реалізації продукту.

ВИСНОВКИ

В роботі було описано розробку інтелектуальної системи фільтрації коментарів з використанням машинного навчання. Для цього був проведений огляд існуючих рішень в даній області, були виділені сильні та слабкі сторони систем по аналізу текстових даних. Було виявлено, що на ринку зараз немає систем фільтрації коментарів, оскільки цей ринок систем лише починає набирати популярність. Були розглянуті алгоритми машинного навчання для вирішення задачі аналізу текстових коментарів. Для реалізації інтелектуальної системи було вирішено використати згорткову нейронну мережу.

В результаті розробки інтелектуальної системи було:

- проаналізовано існуючі алгоритми машинного навчання для аналізу та класифікації текстових коментарів за ознаками, вибрано згорткову нейронну мережу як лідера в області вирішення данної задачі;
- розроблено та описано модель машинного навчання для згорткової нейронної мережі;
- розроблено інтелектуальну систему фільтрації коментарів в цілому, що також було супроводжено розробкою відповідних для неї діаграм;
- розроблено та описано інтерфейс користувача системи.

Розроблена модель нейронної мережі була протестована, а також результат роботи оцінки коментарів на вміст обраних ознак в них був порівняний з класичними алгоритмами машинного навчання. Розроблена модель згорткової нейронної мережі показала гарний результат та точність оцінки коментарів за вибраними ознаками для подальшої фільтрації, що досягла в понад 97% точності на тестовій вибірці даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Perspective [Електронний ресурс] – Режим доступу до ресурсу: https://perspective.com/
2. Tisane Labs [Електронний ресурс] – Режим доступу до ресурсу: https://tisane.ai/
3. Machine Learning: What it is and why it matters [Електронний ресурс] – Режим доступу до ресурсу: https://www.sas.com/en_us/insights/analytics/machine-learning.html
4. Logistic regression [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Logistic_regression
5. Hastie T. The Elements of Statistical Learning: Prediction, Inference and Data Mining (Second Edition) / T. Hastie, R. Tibshirani, J. Friedman. – New York, 2009.
6. Furey T. Support vector machine classification and validation of cancer tissue samples using microarray expression data / T. Furey. – Bioinformatics, 200. – 906-914 с.
7. Moisen G.G., Freeman E.A., Blackard J.A. Predicting tree species presence and basal area in Utah / G.G. Moisen, E.A. Freeman, J.A. Blackard. – Ecological Modelling, 2006. – 176-187 с.
8. Применение модели градиентного бустинга для прогнозирования развития диабета [Електронний ресурс] – Режим доступу до ресурсу: https://moluch.ru/archive/131/36581/
9. XGBoost: A Scalable Tree Boosting System [Електронний ресурс] – Режим доступу до ресурсу: https://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf
10. XGBoost Parameters. Python Package Introduction [Електронний ресурс] – Режим доступу до ресурсу: https://xgboost.readthedocs.io/en/latest/parameter.html
11. Neuroscience-Inspired Artificial Intelligence [Електронний ресурс] – Режим доступу до ресурсу: https://www.ncbi.nlm.nih.gov/pubmed/28728020

12. Artificial Intelligence: A Modern Approach [Електронний ресурс] – Режим доступу до ресурсу: https://faculty.psau.edu.sa/filedownload/doc-7-pdf-a154ffbcec538a4161a406abf62f5b76-original.pdf

13. Гудфеллоу, Я. Глубокое обучение / Я. Гудфеллоу. - М. : ДМК Пресс, 2017. - 652 с.

14. Searching for Activation Functions [Електронний ресурс] – Режим доступу до ресурсу: https://arxiv.org/abs/1710.05941

15. Keras: Sequential model [Електронний ресурс] – Режим доступу до ресурсу: https://keras.io/models/sequential/

16. Understanding of Convolutional Neural Network [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

17. Understanding Convolutional Neural Networks for NLP [Електронний ресурс] – Режим доступу до ресурсу: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

18. Agile Java Development with Spring 2006 p. - Hemrajani A.

19. Spring Data for Web Development 2012 p. – Risberg T., Brisbin J. – 316 c.

20. Spring Data JPA [Електронний ресурс] – Режим доступу до ресурсу: https://docs.spring.io/spring-data/jpa/docs/current/reference/html/

21. Spring Security – Roles and Privileges [Електронний ресурс] – Режим доступу до ресурсу: http://www.baeldung.com/role-and-privilege-for-spring-security-registration

22. Log4J Overview [Електронний ресурс] – Режим доступу до ресурсу: https://logging.apache.org/log4j/2.x/manual/index.html

23. Архитектура REST [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/post/38730/

ДОДАТОК А

Стаття за темою дисертації

Аналіз коментарів за допомогою машинного навчання

Писаренко Олег Анатолійович
КПІ ім. Ігоря Сікорського
Київ, Україна
oa.pisarenko@gmail.com

Дорошенко Анатолій Юхимович
КПІ ім. Ігоря Сікорського
Київ, Україна
a-y-doroshenko@ukr.net

Анотація. **В роботі демонструється результат реалізації сервісу для аналізу текстових коментарів за допомогою машинного навчання. Для цього був обраний підхід ансамблювання алгоритмів машинного навчання, до якого війшли алгоритми Gradient Boosting, Random Forest та логістична регресія. Була навчена прогнозуюча модель та оцінена її точність на основі тестових даних.**

Ключові слова: машинне навчання, Tensorflow, Gradient Boosting, Random Forest, логістична регресія.

Вступ

В час швидкого розвитку інтернету, соціальних мереж та стрімінгових сервісів виникла необхідність в реалізації інструментів та платформ для обговорення та спілкування між людьми. Не завжди таке спілкування є конструктивним та по суті, а тим більше коли спільнота стає досить великою або популярною, там з'являються люди із зовсім інакшою метою – не конструктивна критика авторів, агресивна та нецензурна дискусія з іншими коментаторами, приниження та образи певних груп людей за різними ознаками. Таким чином виникла ідея в створенні інтелектуальної системи, завдяки якій автори та модератори різних інтернет-майданчиків зможуть автоматизувати стеження за своєю спільнотою коментаторів та завчасно фільтрувати негативні коментарі. Частиною такої системи є сервіс оцінки коментарів за різними негативними ознаками.

В якості негативних ознак для класифікації коментарів було обрано ознаки: токсичність, висока токсичність, нецензурна лексика, погроза, образа та персональна ненависть.

Ансамбль методів

Для створення інтелектуальної системи фільтрації коментарів, а саме в частині сервісу аналізу коментарів було вирішено використати машинне навчання, як один із найбільш сучасних та ефективних інструментів аналізу текстових та графічних даних і знаходження закономірностей в проаналізованих даних для подальшого їх використання на реальних даних. Чим більш різноманітні вхідні дані, тим простіше алгоритмам знайти закономірності та завдяки використанню різних алгоритмів класифікації даних машинного навчання можна досягти швидкодії та дуже високої кінцевої точності необхідного результату.

Для побудови прогнозуючої моделі було вирішено обрати один із підходів в машинному навчанні – ансамблювання алгоритмів машинного навчання [1]. Ансамблі є більш точними в прогнозуванні кінцевого результату за окремо взяті методи навчання, так як такий підхід дозволяє взяти різні методи класифікації та навчити їх виправляти помилки один одного. Якість такої системи буде набагато вище, чим використання кожного із методів окремо.

Алгоритм Gradient Boosting

Класифікатор Gradient Boosting – це алгоритм машинного навчання, що відноситься до вирішення проблем регресії та класифікації, який продукує прогнозуючу модель у вигляді дерев рішень [2].

Алгоритми класифікації часто використовують логарифмічні втрати. Класифікатору Gradient Boosting не потрібно отримувати нову функцію втрат щоразу, коли додається алгоритм бустингу, скоріше будь-яка диференційована функція втрат може бути застосована до системи.

Gradient Boosting складається з двох ключових елементів: дерева рішень та адитивного компоненту. Для мінімізації похибки між заданими параметрами використовується процедура, схожа на спуск градієнта [3]. Це робиться шляхом взяття обчисленої втрати та виконання градієнтного спуску для зменшення цих втрат. Потім параметри дерева змінюються для зменшення залишкових втрат.

Вихід нового дерева потім додається до виходу попередніх дерев, використовуваних у моделі. Цей процес повторюється, поки не буде досягнуто раніше визначеної кількості дерев, або втрати не зменшаться нижче певного порогу.

В цьому алгоритмі машинного навчання, процес навчання послідовно пристосовує створені ним моделі, щоб забезпечити більш точну оцінку для шуканої ознаки. Під час кожної конкретної ітерації нове дерево рішень, яке отримується під час навчання на попередніх помилках, які були отримані на попередніх ітераціях роботи алгоритму.

Алгоритм Random Forest

Класифікатор Random Forest – алгоритм машинного навчання для класифікації, регресії та інших завдань, які можна вирішити за допомогою