Міністерство освіти і науки, молоді та спорту України
Тернопільський національний економічний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

До захисту допущено
Завідувач кафедри
комп'ютерної інженерії
к.т.н., доц. О.М.Березький
_____
"___" _____ 20___ р.

ДИПЛОМНА РОБОТА
освітньо-кваліфікаційного рівня "Магістр"
зі спеціальності 8.05010201 "Комп'ютерні системи та мережі"
на тему:

АЛГОРИТМИ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ ЕЛЕКТРОННОЇ ПОШТИ НА ОСНОВІ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ

Студент групи КСМзм - 51
Попов В.А.

_____
підпис
Науковий керівник
д.т.н., професор Саченко А.О.

_____
підпис
Консультант з нормоконтролю
Палій І.О.
_____
Прізвище, ініціали Підпис

Тернопіль – 2012

Міністерство освіти і науки, молоді та спорту України
Тернопільський національний економічний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

"Затверджую"
Зав. кафедри
комп'ютерної інженерії
к.т.н., доц. О.М. Березький
_____
"____"_____20__ р.

З А В Д А Н Н Я

НА ДИПЛОМНУ РОБОТУ СТУДЕНТА

Попова Віталія Андрійовича

1. **Тема дипломної роботи** "Алгоритми класифікації повідомлень електронної пошти на основі штучних нейронних мереж" затверджена наказом університету №_____ від „____" _____20__ р

2.**Термін здачі** закінченої дипломної роботи _____

3. **Об'єкт дослідження:** процеси фільтрації несанкціонованих масових розсилок рівня поштового сервера, що використовують алгоритми машинного навчання, а також методи класифікації електронної пошти, що засновані на таких алгоритмах.

4. **Предмет дослідження**: архітектурні рішення системи класифікації повідомлень електронної пошти.

5. **Перелік задач, які мають бути вирішені**:
- розглянути задачу класифікації повідомлень електронної пошти;
- розглянути моделі представлення об'єктів для задач класифікації;
- провести аналіз відомих методів та засобів класифікації повідомлень електронної пошти;
- зробити постановку задачі дослідження;
- провести вибір архітектури навчальної системи фільтрації пошти серверного рівня;
- здійснити вибір моделі представлення даних;
- експериментально дослідити метод скорочення простору ознак;
- розробити нейромережевий класифікатор повідомлень електронної пошти;
- розробити архітектуру серверної системи класифікації електронної пошти;
- розробити користувацький інтерфейс системи;
- провести програмну реалізацію модулів;
- здійснити інтеграцію системи класифікації повідомлень електронної пошти з поштовими системами.

6. **Перелік ілюстративного матеріалу:**
- архітектура серверної системи фільтрації пошти, яка заснована на традиційних методах,
- архітектура серверної системи фільтрації пошти, навчальний агент,
- архітектура серверної системи фільтрації пошти, класифікуючий агент,

- результати порівняльного тестування методів скорочення простору ознак,
- архітектура багаторівневого персептрона,
- архітектура рекурентної нейронної мережі,
- алгоритм навчання нейтронної мережі,
- архітектура експериментальної серверної системи фільтрації електронної пошти, яка заснована на навчальних методах класифікації,
- стадії навчання системи фільтрації пошти,
- стадії класифікації системи фільтрації пошти,
- архітектура експериментальної системи.

7. **Консультанти по роботі**

Розділ	Консультант	Підпис
1		
2	Комар М.П.	
3	Комар М.П.	

КАЛЕНДАРНИЙ ПЛАН

№	Назва структурних частин ДР	Термін виконання	Примітка
1	Аналіз відомих рішень класифікації повідомлень електронної пошти	15.09.2011 – 5.11.2011	
2	Архітектурні рішення системи класифікації повідомлень електронної пошти	6.11.2011 – 31.01.2012	
3	Реалізація серверної системи класифікації електронної пошти	1.02.2012 – 23.04.2012	

Завдання прийняв до виконання _____
(підпис)

Керівник дипломної роботи _____
(підпис)

РЕФЕРАТ

Дипломна робота на тему “Алгоритми класифікації повідомлень електронної пошти на основі штучних нейронних мереж” на здобуття освітньо-кваліфікаційного рівня “Магістр” зі спеціальності “Комп’ютерні системи та мережі” написана обсягом 81 сторінка і містить 18 ілюстрацій, 1 таблицю, 1 додаток та 27 джерел за переліком посилань.

Метою роботи є комплексне рішення для системи класифікації поштових повідомлень серверного рівня, яке засноване на навчальних методах класифікації, що володіє такими перевагами методів штучного інтелекту, як персоніфікація, автономність, висока точність виявлення спаму при низькій кількості помилково-позитивних помилок.

Методи досліджень. В дипломній роботі використовуються методи системного аналізу процесів фільтрації повідомлень електронної пошти, нейромережеві методи класифікації, багатоагентний підхід, методи чисельного експерименту.

Запропонована персоніфікована модель класифікації повідомлень електронної пошти на рівні поштового сервера. Розроблені нейромережеві алгоритми класифікації електронних повідомлень.

На основі запропонованих алгоритмів розроблена і апробована експериментальна багатоагентна серверна система класифікації електронних повідомлень, що використовує персоніфіковану модель класифікації. Розроблені програмні засоби інтеграції системи класифікації повідомлень електронної пошти з поштовими системами.

Ключові слова: ЕЛЕКТРОННА ПОШТА, ПОШТОВИЙ СЕРВЕР, НЕЙРОННА МЕРЕЖА, ПЕРСОНІФІКОВАНА МОДЕЛЬ, БАГАТОАГЕНТНА СИСТЕМА.

ABSTRACT

Diploma work «Algorithms for classification of email messages based on artificial neural networks» on acquiring of educationally-qualification «Master» degree, from speciality «Computer systems and networks» with total volume 81 pages that contains 18 illustrations, 1 table, 1 addition and 27 sources of information according to the list of references.

The object is a complex solution for a system of classification of server level email messages, which is based on the educational methods of classification, that has the following advantages of artificial intelligence methods as: personification, autonomy, high accuracy of spam detection with low number of false-positive errors.

Research Methods. In the thesis work are used methods of system analysis of e-mail filtering, neural network classification methods multi-agent approach, methods of numerical experiment.

The classification model of personalized emails at the mail server is proposed. The neural network algorithms for classification electronic massages are developed.

Based on the proposed algorithms is developed and tested an experimental multiagent server system of e-mail massages classification with the help of personalized classification model. The software for integration of e-mail classification system with mail systems are developed.

Keywords: E–MAIL, MAIL SERVER, NEURAL NETWORKS, PERSONALIZED MODEL, MULTI–AGENT SYSTEMS.

ЗМІСТ

ВСТУП

Актуальність роботи. Швидке зростання популярності електронних засобів комунікації, зокрема електронної пошти, а також низька вартість їх використання приводить до збільшення потоку несанкціонованих масових розсилок. По різних оцінках в даний час від 40 до 90% всіх електронних повідомлень в Інтернет є спамом [1].

Несанкціоновані розсилки завдають очевидного збитку – це реальні матеріальні втрати компаній і користувачів мережі. Зайве навантаження на мережі і устаткування, згаяний час на сортування і видалення листів, витрачені гроші на оплату трафіку – це неповний перелік проблем, які приносить спам. Часто спам використовується як канал для розповсюдження вірусів.

Отже, актуальною задачею є дослідження, розробка і створення нових алгоритмів, методів, засобів і систем, що забезпечують захист користувачів від несанкціонованих масових розсилок електронної пошти і мінімізують втрати, що завдаються такими розсилками.

В даний час розроблені різні засоби, методи і підходи для боротьби із спамом. Їх можна розділити на дві категорії:

- запобігання розповсюдженню спаму;
- запобігання отриманню спаму або фільтрація.

Перша категорія – це різні адміністративні і технічні методи, направлені на запобігання розсилки спаму. Сюди відносяться такі рішення як:

- законодавчі заходи по обмеженню розсилки спаму;
- нові протоколи електронної пошти, що використовують аутентифікацію відправника;
- блокування поштових серверів, користувачі яких розсилають спам.

Існує ряд підходів, що пропонують зробити розсилку спаму економічно невигідною. Одним з таких є пропозиція зробити відправку кожного електронного повідомлення платною. Плата за одне повідомлення повинна бути незначною. В цьому випадку для звичайного користувача це буде непомітним. Для тих, хто займається розсилкою спаму, щоб досягти помітного результату, необхідно

розіслати одночасно сотні тисяч і мільйони повідомлень. В цьому випадку вартість такої розсилки стає значною, що робить її економічно невигідною.

Друга категорія засобів боротьби з несанкціонованими масовими розсилками – це методи, направлені на запобігання отриманню спаму користувачами, так звані методи фільтрації спаму.

Задачу фільтрації спаму, можна розглядати як задачу класифікації [2] – визначення приналежності об'єкту до одного із заздалегідь виділених класів на підставі аналізу сукупності ознак, що характеризують даний об'єкт. Можна виділити дві основні групи методів, які використовуються при вирішенні задачі фільтрації спаму:

- традиційні методи – це ті методи, для яких модель класифікації, або іншими словами ті дані, на підставі яких класифікуються листи, визначається експертом.
- навчальні методи – це ті методи, для яких модель класифікації будується за допомогою методів інтелектуального аналізу даних.

Таким чином, для фільтрації електронної пошти можна застосувати широкий спектр методів, які використовуються для вирішення задачі класифікації. Зважаючи на специфіку предметної області, основну складність представляє вибір базового методу класифікації і його адаптація до умов застосування в задачі фільтрації спаму.

Мета і завдання дослідження. Метою даного дослідження є комплексне рішення для системи класифікації поштових повідомлень серверного рівня, яке засноване на навчальних методах класифікації, що володіє такими перевагами методів штучного інтелекту, як персоніфікація, автономність, висока точність виявлення спаму при низькій кількості помилково-позитивних помилок. В той же час система класифікації повідомлень повинна забезпечувати продуктивність, достатню для її використання на рівні поштового сервера.

Для досягнення поставленої мети в дипломній роботі необхідно вирішити наступні завдання:

- розглянути задачу класифікації повідомлень електронної пошти;
- розглянути моделі представлення об'єктів для задач класифікації;

– провести аналіз відомих методів та засобів класифікації повідомлень електронної пошти;

– зробити постановку задачі дослідження;

– провести вибір архітектури навчальної системи фільтрації пошти серверного рівня;

– здійснити вибір моделі представлення даних;

– експериментально дослідити метод скорочення простору ознак;

– розробити нейромережевий класифікатор повідомлень електронної пошти;

– розробити архітектуру серверної системи класифікації електронної пошти;

– розробити користувацький інтерфейс системи;

– провести програмну реалізацію модулів;

– здійснити інтеграцію системи класифікації повідомлень електронної пошти з поштовими системами.

Об'єкт дослідження – процеси фільтрації несанкціонованих масових розсилок рівня поштового сервера, що використовують алгоритми машинного навчання, а також методи класифікації електронної пошти, що засновані на таких алгоритмах.

Предмет дослідження – архітектурні рішення системи класифікації повідомлень електронної пошти.

Методи досліджень. В дипломній роботі використовуються методи системного аналізу процесів фільтрації повідомлень електронної пошти, нейромережеві методи класифікації, багатоагентний підхід, методи чисельного експерименту.

Наукова новизна одержаних результатів. Запропонована персоніфікована модель класифікації повідомлень електронної пошти на рівні поштового сервера. Розроблені нейромережеві алгоритми класифікації електронних повідомлень.

Практичне значення отриманих результатів. На основі запропонованих алгоритмів розроблена і апробована експериментальна багатоагентна серверна

система класифікації електронних повідомлень, що використовує персоніфіковану модель класифікації. Розроблені програмні засоби інтеграції системи класифікації повідомлень електронної пошти з поштовими системами.

Перший розділ дипломної роботи присвячений розробці і обгрунтуванню основних концепцій серверної системи класифікації пошти, заснованої на навчальних методах класифікації і використовує персональну модель. Представлений аналіз навчальних методів класифікації і їх характеристик. Обгрунтовується вибір базового алгоритму класифікації, формулюються вимоги по його оптимізації і представленню структур даних.

В другому розділі проведений вибір архітектури системи, яка забезпечує застосування навчального алгоритму класифікації за допомогою ефективного розподілу навантаження і забезпечує масштабованість системи і здатність працювати в гетерогенному середовищі. Розглядається вибір моделі представлення даних та результати експериментів, що обгрунтовують ефективність даного підходу. Проведений вибір нейронної мережі та розроблений алгоритм її навчання для класифікації повідомлень електронної пошти.

В третьому розділі дипломної роботи проведена реалізація серверної системи класифікації електронної пошти, зокрема розроблена архітектури системи, користувацький інтерфейс, який дозволяє редагувати настройки навчання, настройки класифікації, проводити навчання і до навчання системи, формувати «чорні»/«білі» списки адрес відправників, зберігати статистику навчання. Також здійснена програмна реалізація модулів системи та інтеграції системи класифікації з поштовими системами.

1 АНАЛІЗ ВІДОМИХ РІШЕНЬ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ ЕЛЕКТРОННОЇ ПОШТИ

1.1 Фільтрація пошти, як задача класифікації

Розглядатимемо задачу фільтрації спаму як класичну задачу класифікації – визначення приналежності об'єкту до одного із заздалегідь виділених класів на підставі аналізу сукупності ознак, що характеризують даний об'єкт. Формально задача класифікації – побудова такого відображення, де кожній парі $< d_j, c_i > \in D \times C$ ставиться у відповідність булеве значення. Тут D – простір об'єктів, що класифікуються, $C = \{c_1, ..., c_{|C|}\}$ – множина заздалегідь визначених категорій. Відображення будується таким чином, що значення *True* для пари $< d_j, c_i >$ означає, що об'єкт d_j належить класу c_i, а значення *False* – не належить. Більш формально, цей процес – побудова наближеної функції $\tilde{\Phi} : D \times C \to \{True, False\}$. Така функція називається класифікатором [3].

Можна виділити два підходи до вирішення задачі класифікації. Першим підходом – класифікатор є набір правил, складених вручну, на підставі інформації, наданої експертом предметної області (knowledge engineering). Об'єкти класифікуються відповідно до даних правил, що визначають їх категорію.

Для системи фільтрації електронної пошти такий підхід – це традиційні методи фільтрації спаму. Наприклад, сигнатурні методи, де сигнатури спам-листів виділяються і створюються експертами, системи RBL.

В минулому велику популярність отримав підхід до побудови класифікаторів на основі алгоритмів машинного навчання. Цей підхід полягає в побудові класифікатора на підставі аналізу характеристик набору об'єктів, заздалегідь класифікованих експертом предметної області. В результаті будується класифікатор, який виділяє характеристики нового некласифікованого документа і на їх підставі відносить об'єкт до того або іншого класу. Заздалегідь класифікований експертом набір об'єктів, на якому проводиться навчання, називається навчальною вибіркою або тренувальним набором.

Далі в цьому розділі розглядаються наступні питання задачі класифікації:

- методи представлення об'єктів – відображення об'єктів, що класифікуються, в простір, в якому функціонують алгоритми класифікації;
- методи і алгоритми побудови класифікаторів;
- порівняння методів класифікації.

1.2 Моделі представлення об'єктів в задачах класифікації

Як правило, об'єкти класифікації є набором істотно неструктурованих і різнорідних даних. Алгоритми класифікації працюють з деякими наборами ознак об'єктів. Звичайно це вектори фіксованої розмірності в деякому просторі. Таким чином, необхідно вибрати будь–яку модель представлення об'єктів – відображення об'єктів класифікації в даний простір.

При виборі моделі представлення розглядаються два суперечливі аспекти, від яких залежить ефективність алгоритму класифікації. По–перше, це точність класифікації, і, по–друге – ресурси обчислювальної системи, що використовуються. Чим складніше представлення використовується, тим вища якість їх класифікації, але в теж час це приводить до використання великих обчислювальних ресурсів.

Найбільш поширена модель представлення об'єктів – векторна модель [4]. Визначається множина T – набір ознак, які характеризують кожен об'єкт з навчального набору. Тоді будь–який об'єкт представляється у вигляді вектора, координатами якого є вагові коефіцієнти ознак об'єкту: $d_j = (w_{1j},...,w_{|T|j})$. Тут ваговий коефіцієнт визначає значущість відповідної ознаки в семантиці документа d_j. Далі різні представлення документів розрізняються в підходах до наступних задач:

- у способах визначення того, що є ознака об'єкту і визначення множини ознак;
- у способах визначення вагових коефіцієнтів ознак.

Виділення ознак об'єктів.

Набір ознак, що характеризують об'єкт, повною мірою визначається типом і суттю об'єкту. Наприклад, якщо, як в нашому у випадку, об'єктом є електронний лист, ознаки можна розділити на текстові і нетекстові.

Нетекстові ознаки визначаються емпірично на підставі експертних оцінок [5]. Їх кількість невелика і фіксована. Зокрема, можливі нетекстові ознаки це:

- наявність, тип, розмір прикріплених файлів;
- кодування, використання форматування;
- різна інформація із заголовка листа (про відправника, одержувачів листа, поштові сервери);
- статистичні дані про текстову частину листа (розмір, середня кількість слів, розділові знаки, використання регістра).

Текстову частину листа можна розглядати як звичайний текстовий документ. Існує декілька різних по складності методів виділення ознак текстових документів. Найбільш поширений підхід в якості ознаки документа пропонує використовувати всі слова, що входять в документ. Зазвичай використовуються деякі заходи по зменшенню розмірності простору таким чином виділених ознак, які будуть далі розглянуті в цьому розділі у відповідному пункті.

Застосовуються також складніші моделі виділення ознак текстових документів. Такі моделі враховують взаємне розташування слів в документі, використовують багатослівні характеристики. Існують різні методики формування багатослівних характеристик. В якості прикладу можна привести виділення всіх пар або всіх трійок слів, що стоять в тексті поряд. Поза сумнівом, такі моделі надають більше інформації про структуру і склад документа, ніж модель «Bag of words», в якій втрачаються семантичні зв'язки між словами. Але в той же час, значно зростає простір ознак і відповідно ростуть витрати на їх зберігання і обробку. Різні експерименти показують, що використання складніших моделей виділення ознак текстових документів не приводять до значного підвищення ефективності моделі [6].

Визначення вагових коефіцієнтів ознак.

Методи визначення вагових коефіцієнтів ознак грунтуються на двох

очевидних твердженнях: по–перше, чим частіше ознака зустрічається в об'єкті, тим він більш значуща (релевантна) для тієї категорії, якій належить об'єкт, і по–друге чим частіше ознака зустрічається у всіх об'єктах набору, тим менше інформації вона несе про відмінність різних об'єктів набору.

Найбільш проста бінарна модель визначення вагових коефіцієнтів: $w_{ij} = 1$, якщо відповідна ознака присутня в даному об'єкті і $w_{ij} = 0$ в іншому випадку. Більшість же моделей використовують складніші методи, що враховують «вагу» ознак в об'єкті.

Розглянемо декілька підходів визначення ваги ознак на прикладі об'єктів – текстових документів. Як було розглянуто раніше, одним з можливих визначень набору ознак для текстових документів є набір всіх слів, що входять у всі документи навчального набору. Найбільш поширений метод визначення ваги слова заснований на статистичних характеристиках його появи в тексті документа і всіх документах навчального набору:

– $w_{ij} = f_{ij}$ – кількість входжень слова в документ. Недолік цього методу визначення ваги – ознаки довших документів отримуватимуть більшу вагу.

$w_{ij} = tf_{ij} = \frac{f_{ij}}{|d_j|}$ – частота входження слова в документ. Де d_j – кількість слів в документі. Тут, навпаки, ознаки слів тих документів, в які входить велика кількість ознак, отримуватимуть меншу вагу.

– $w_{ij} = 1 + \log(tf_{ij})$. В порівнянні з попередньою, ця формула стійкіша до переоцінки документів, де дане слово зустрічається дуже часто.

– $w_{ij} = tf \times idf_{ij} = f_{ij} \cdot \log\left(\frac{N}{n_i}\right)$, де N – загальна кількість документів в наборі, n_i – кількість документів в наборі, в яку хоч би раз входить дане слово. На відміну від попередніх, цей метод визначення ваги враховує також частоту появи слова у всій колекції документів.

– $w_{ij} = tfc_{ij} = \frac{tf \times idf_{ij}}{\sqrt{\sum_{k=1}^{|d_j|}\left[f_{kj} \cdot \log\left(\frac{N}{n_k}\right)\right]^2}}$. Цей метод, на відміну від попередніх враховує різну довжину документів, використовуючи нормалізацію.

Існують різні модифікації представлених методів, хоча, швидше за все найбільш поширеним є останній з розглянутих. Ваги, обчислені по будь–якому з даних методів, монотонно зростають із зростанням числа входжень відповідної ознаки в документ.

Як правило, виконують нормування вагових коефіцієнтів так, щоб $0 \le w_{ij} \le 1$. Таким чином будь–який об'єкт $d_j \in I^{|T|}$. Достатньо часто використовується наприклад така норма:

$$\tilde{w}_{ij} = \frac{w_{ij}}{\sqrt{\sum_{s=1}^{|T|} (w_{sj})^2}} \qquad (1.1)$$

1.3 Аналіз відомих методів та засобів класифікації повідомлень електронної пошти

1.3.1 Методи класифікації

Оцінці і порівнянню алгоритмів класифікації, у тому числі і для задачі класифікації текстових документів, присвячений ряд публікацій [2, 3, 6]. Задача фільтрації електронної пошти на серверному рівні накладає певні обмеження на алгоритми класифікації. Тут буде розглянуто декілька алгоритмів, які або застосовуються в даний час для вирішення цієї задачі, або мають найбільші шанси бути застосовними.

Naive Bayes.

Поширений алгоритм класифікації, який використовується в даний час в системах фільтрації пошти (як правило, клієнтських), що реально діють, – це Naive Bayes. Класифікатор Байєса сконструйований, використовуючи тренувальні

набори для оцінки ймовірності того, що новий документ *d* належить до категорії c_j, використовуючи теорему умовної ймовірності Байєса:

$$P(c_j|d) = \frac{P(c_j)P(d|c_j)}{P(d)}, \tag{1.2}$$

Дільник в цій формулі не міняється від категорії до категорії і тому може не враховуватися. Наївна частина даної моделі полягає в тому припущенні, що слова є незалежними, тобто вважається, що поява ознак (слів) в документі випадковою, і отже, при розрахунку умовної ймовірності спрощений вираз приймає вигляд: $P(c_j|d) = P(c_j)\prod_{i=1}^{M} P(d_i|c_j)$.

Оцінка $\hat{P}(c_j)$ для $P(c_j)$ може бути підрахована, грунтуючись на числі тренувальних документів, співставлених категорії c_j : $\hat{P}(C = c_j) = \frac{N_j}{N}$.

Далі, оцінка $\hat{P}(d_i|c_j)$ для ймовірності $P(d_i|c_j)$ обчислюється:

$$\hat{P}(d_i|c_j) = \frac{1 + N_{ij}}{M + \sum_{k=1}^{M} N_{kj}}, \tag{1.3}$$

де N_{ij} – це кількість разів, яка слово *i* зустрілося в документі з категорії c_j в тренувальному наборі.

Ймовірність $P(c_j|d)$ розраховується для кожної категорії c_j і новому документу приписується та категорія, чия ймовірність більша.

Причиною популярності цього алгоритму є простота і достатньо низька ресурсоємність. Час навчання лінійно залежить від розміру тренувального набору. Процес навчання полягає в перерахунку статистики для вектора ознак. Швидкість класифікації також велика, оскільки лінійно залежить від кількості ознак, що зустрілися в новому документі. Проте точність класифікації методу Байєса по

порівнянню з іншими алгоритмами, судячи по результатах різних порівнянь [3] достатньо невелика.

Memory–Based (*k* найближчих сусідів).

Сімейство методів класифікації, для яких відсутня процедура навчання. Для класифікації нових документів вони прямо використовують тренувальний набір, що зберігається повністю. У англомовній літературі ця група методів називається memory–based. Проста форма пам'яті – це багатовимірний простір, визначений атрибутами з векторів в тренувальних прикладах. Кожен тренувальний приклад представляється точкою. в цьому просторі. Процедура класифікації зазвичай є простим варіантом алгоритму «*k* найближчих сусідів» (*k–nn*). *k–nn* співвідносить з кожним новим документом переважаючий клас серед *k* тренувальних прикладів, найближчих до оброблюваного прикладу (його *k* найближчих сусідів). Зазвичай використовується евклідова норма для визначення відстані між двома векторами, тобто для двох даних векторів $x_i = \{x_{i_1}, x_{i_2}, ..., x_{i_n}\}$ і $x_j = \{x_{j_1}, x_{j_2}, ..., x_{j_n}\}$ відстань між ними буде рівною $d(x_i, x_j) = \sqrt{\sum_{h=1}^{n}(x_{i_h} - x_{j_h})^2}$.

У деяких реалізаціях методу використовується варіація алгоритму *k–nn*. Одна важлива відмінність полягає у визначенні *k*–сосідніх. Варіація алгоритму розглядає всі тренувальні приклади на *k* найближчих відстанях від оброблюваного прикладу. Якщо є більш ніж один сусідній набір на кожній відстані, то алгоритм досліджує значно більше, чим *k* сусідів. У таких випадках потрібне невелике значення *k*, щоб уникнути розгляду прикладів дуже відмінних від оброблюваного прикладу.

Перевагою даного методу є відсутність етапу навчання. Моделлю, на підставі якої ухвалюється рішення про приналежність нового документа тому або іншому класу, є весь тренувальний набір. Отже, його необхідно зберігати постійно, що вимагає додаткових витрат пам'яті. Складність класифікації даного методу складає $O(N \cdot n)$, де N – кількість прикладів в тренувальному наборі і n – розмірність вектора ознак.

Лінійний дискримінант Фішера.

Лінійний дискримінант Фішера (Fisher's Linear Discriminant, FLD) був

спочатку розроблений для вирішення проблем класифікації для випадку двох класів. Основна ідея методу полягає в проекції векторів ознак на пряму, що еквівалентно обчисленню лінійної комбінації їх компонент (рисунок 1.1).

Допустимо, є набір *M* прикладів з *N*–мspрного простору $d = d_1,...,d_N$, N_1 з яких належать класу c_1, а N_2 класу c_2 . Для визначення класу нового прикладу *d* потрібно отримати скаляр *y*, проектуючи приклад на лінію: $y = w^T d$. Зі всіх можливих ліній *w* слід вибрати таку, яка максимально розділяє проекції прикладів з двох класів c_1 і c_2.

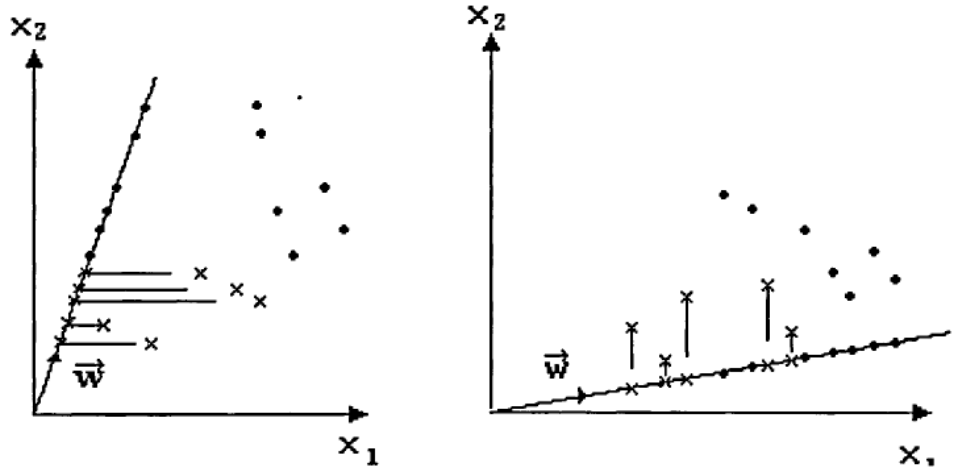

Рисунок 1.1 – Лінійний дискримінант Фішера: різні проекції точок *d* на *w*

Для того, щоб знайти оптимальний для класифікації вектор проекції, потрібно визначити міру вимірювання відстані між проекціями. Середній вектор для кожного класу в просторі ознак *x* і *y* визначається по формулах:

$$\mu_i = \frac{1}{N_i}\sum_{d\in c_i} d \text{ і } \tilde{\mu}_i = \frac{1}{N_i}\sum_{y\in c_i} y = \frac{1}{N_i}\sum_{x\in c_i} w^T d = w^T \mu_i . \qquad (1.4)$$

Для розрахунку відстані між середніми спроектованими векторами можна використовувати функцію $J(w) = |\tilde{\mu}_1 - \tilde{\mu}_2| = |w^T(\mu_1 - \mu_2|$. Проте, відстань між середніми спроектованими векторами не дуже хороший показник, оскільки вона не враховує стандартне відхилення всередині класів.

Фішер запропонував максимізувати функцію, яка представляє різницю

середніх, нормалізовану у міру розкиду вибірок всередині класу. Для кожного класу визначається розкид, еквівалент дисперсії: $\tilde{s}_i^2 = \sum_{y \in c_i} (y - \tilde{\mu}_i)^2$, причому $(\tilde{s}_1^2 + \tilde{s}_2^2)$ називається розкидом вибірок всередині класу. Лінійний дискримінант Фішера визначається як лінійна функція $w^T d$, яка максимізувала функцію

$$J(w) + \frac{\left|\tilde{\mu}_1 - \tilde{\mu}_2\right|^2}{(\tilde{s}_1^2 + s_2^2)}. \qquad (1.5)$$

Таким чином, потрібно знайти таку проекцію, де приклади з одного класу проектуються дуже близько один до одного, і в той же час середні спроектовані вектори проектуються якнайдалі один від одного.

Для того, щоб знайти оптимальну проекцію w^*, потрібно виразити *J(w)* як явну функцію від w . Для цього визначаються матиці розкиду всередині і між класами:

$$S_i = \sum_{d \in C_i} (d - \mu_i)(d - \mu_i)^T \text{ і } S_1 + S_2 = S_W \qquad (1.6)$$

Тоді розкид проекцій у може бути виражений як функція від матриці розкиду в просторі ознак *x*:

$$\tilde{S}_i^2 = \sum_{y \in c_i} (y - \tilde{\mu}_i)^2 = \sum_{d \in c_i} (w^T d - w^T \mu_i)^2 = \sum_{d \in c_i} w^T (d - \mu_i)(d - \mu_i)^T w = w \qquad (1.7)$$

$$(\tilde{s}_1^2 + \tilde{s}_2^2) = w^T S_W w \qquad (1.8)$$

Аналогічно в проекціях отримаємо:

$$(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = (w^T \mu_1 - w^T \mu_2)^2 = w^T \xleftarrow{(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T} w = w^T S_B w \qquad (1.9)$$

S_W називається матрицею розкиду всередині вибірки, вона пропорційна коваріаційній матриці класу, а S_B – матриця розкиду між класами. І таким чином, ми можемо виразити критерій Фішера через S_W і S_B як

$$J(w) = \frac{w^T S_B w}{w^T S_W w}. \tag{1.10}$$

Вирішуючи задачу максимізації функції, отримуємо $S_W{}^{-1} S_B w - Jw = 0$.

І остаточно:

$$w^* = \arg\max_w \left\{ \frac{w^T S_B w}{w^T S_W w} \right\} = S_w{}^{-1}(\mu_1 - \mu_2) \tag{1.11}$$

Порівнюючи значення $w^*\mu_1$ і $w^*\mu_2$ із значенням проекції того, що класифікується прикладу w^*d, документу d приписується клас c_1 або c_2 залежно від близькості до відповідної проекції класу.

Навчання методу лінійного дискримінанта Фішера вимагає обчислення зворотної матриці порядку M, для чого необхідно M^3 операцій і займає основний час і пам'ять. Результатом навчання є вектор, на який проектуються приклади. Метод працює довше за решту всіх досліджуваних методів і вимагає багато пам'яті на проміжні обчислення при перетворені матриці. Для перенавчання методу потрібно буде знову перераховувати пряму, що займає значний час. Але для класифікації потрібне перемножування M чисел і їх складання, що виконується швидше, ніж в методі Байєса і тим більше в Memory–Based методі.

Метод опорних векторів.

Метод опорних векторів (Support Vector Machines – SVM) – один з популярних методів класифікації, грунтується на теорії статистичного навчання [7–9].

Ідея методу опорних векторів полягає у відшукуванні гіперплощини в

просторі ознак, яка розділяла б об'єкти двох класів з максимальною межею між ними.

У геометричній інтерпретації метод виглядає таким чином (рисунок 1.2).

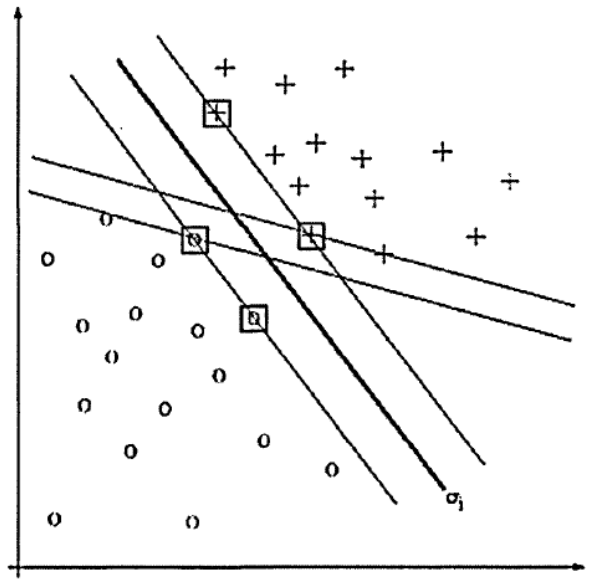

Рисунок 1.2 – Метод опорних векторів

Припустимо, що є тренувальний набір $D = \{(x, y) : x \in R^n, y \in \{-1;1\}\}$, тоді побудова класифікатора – це процес пошуку в просторі R^n гіперплощини, яка розділяє позитивні і негативні приклади з тренувального набору так, щоб межа між ними була максимальна. У випадку, якщо об'єкти тренувального набору лінійно розділяються, то пошук такої гіперплощини не викликає труднощів. В цьому випадку умови, які визначають розділення об'єктів гіперплощиною (*w*,*b*), виглядають так:

$(w, x_i) + b \geq 1$, при $y_i = +1$

$(w, x_i) + b \leq -1$, при $y_i = -1$

Вирішальна функція для класифікації нових об'єктів, очевидно, виглядає так: $f(x) = \text{sgn}((w, x) + b)$.

Умови оптимальності гіперплощини – тобто такої, що максимізує межу між об'єктами різних класів, визначається так:

$$\rho = \frac{2}{\|w\|}. \qquad 1.12)$$

Таким чином, задача пошуку оптимальної гіперплощини зводиться до наступної задачі оптимізації:

$$\min \Phi(w) = \frac{1}{2}(w,w) \text{ при умові } y((w,x_i)+b) \geq 1, \forall i \in [1,n] \qquad (1.13)$$

Але, як правило, неможливо побудувати таку гіперплощину, яка лінійно розділяла б об'єкти різних класів. Тобто завжди знаходяться об'єкти тренувального набору, які порушують умови – викиди (outliers). У такому разі вводиться набір фіктивних змінних $\xi_i \geq 0, i = 1..N : y_i((w,x_i)+b) \geq 1-\xi_i, \forall i \in [1,N]$.

І задача пошуку оптимальної гіперплощини в лінійно нероздільному випадку зводиться до наступної задачі оптимізації:

$$\min \Phi(w) = \frac{1}{2}(w,w) + C\sum_{i=1}^{N} \xi_i$$

$$\text{за умови } y_i((w,x_i)+b) \geq 1-\xi_i,\ \xi_i \geq 0, \forall i \in [1,N]. \qquad (1.14)$$

Права частина в цій формулі – штраф за невірно класифіковані об'єкти тренувального набору. Параметр C визначає компроміс між властивістю класифікатора до узагальнення (тобто максимізації межі між категоріями) і мінімізації помилки класифікації. Це єдиний параметр, який підбирається емпірично. Якщо він дуже малий – це зменшує штраф за невірно класифіковані об'єкти тренувального набору і таким чином приводить до спрощення моделі і зменшення точності. Якщо параметр дуже великий – це приводить до «перенавчання», тобто модель стає дуже складною і втрачає властивість до узагальнення (добре класифікує тільки приклади з тренувального набору, але погано – будь–які нові приклади).

Ця задача оптимізації у свою чергу зводиться до задачі квадратичного програмування:

$$\max_{\alpha \in \Re^n} W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j (x_i, x_j),$$

за умови $\sum_{i=1}^{N} y_i \alpha_i = 0, C \geq \alpha_i \geq 0, i = 1,..., N$ (1.15)

Тут коефіцієнти α_i відмінні від нуля тільки для частини об'єктів, які називаються опорними векторами $0 < \alpha_i < C$, для outliers $\alpha_i = C$ і для останніх елементів $\alpha_i = 0$.

З умов Куна–Такера для задачі квадратичного програмування:

$$\alpha_i [y_i (\langle w, x_i \rangle + b) - 1 + \xi_i] = 0, i = 1,..., N \quad (1.16)$$

$$(C - \alpha_i)\xi_i = 0, i = 1,..., N \quad (1.17)$$

отримуємо $w_0 = \sum_{i=1}^{N} \alpha_i y_i x_i$,

тоді остаточно вирішальна функція виглядає так:

$$f(x) = \text{sgn}\left(\sum_{i}^{n} \alpha_i y_i (x_i, x) + b \right) \quad (1.18)$$

Таким чином, для класифікації нового об'єкту необхідно зберігати тільки частину навчального набору. Як правило, число опорних векторів істотно менше числа елементів в початковому тренувальному наборі.

Перевага методу опорних векторів в тому, що як в задачі квадратичного програмування, так і у вирішальній функції використовується тільки скалярний добуток векторів, що представляють об'єкти тренувального набору.

З іншого боку, можна припустити, що одну і ту ж кількість об'єктів тренувального набору в просторі більшої розмірності буде простіше розділити гіперплощиною. Таким чином, якщо існує деяка нелінійна функція, що відображає вектора тренувального набору в простір більшої розмірності $\varphi(x) \in R^m, m > n$, то оптимальну гіперплощину можна шукати в цьому новому

просторі.

Завдяки тому, що при побудові рішення і в найвирішальнішій функції використовується тільки скалярний добуток, немає необхідності задавати це нелінійне відображення в явному вигляді. Необхідно лише вибрати так звану kernel–функцію, яка неявно відображає вектора початкового простору в простір більшої розмірності: $K(x_i, x_j) = (\varphi(x_i), \varphi(x_j))$. Kernel–функція або потенційна функція – це суть міри схожості об'єктів в просторі характеристик, обмеження, що накладаються на неї – задоволення умовам теореми Мерсера [10].

Таким чином, в термінах kernel–функцій, задача перетворюється до вигляду:

$$\max_{\alpha \in \Re^n} W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j K(x_i, x_j)$$

$$\text{при умові } \sum_{i=1}^{N} y_i \alpha_i = 0,\ C \geq \alpha_i \geq 0,\ i = 1,...,N \qquad (1.19)$$

А вирішальна функція має вигляд:

$$f(x) = \operatorname{sgn}\left(\sum_{i}^{n} \alpha_i y_i K(x_i, x) + b \right) \qquad (1.20)$$

Декілька прикладів найбільш поширених kernel–функцій це:

- поліноміальна $K(x_i, x_j) = (x_i, x_j)^p, p \in N$;
- RBF $K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2} \left\| x_i - x_j \right\|^2 \right)$

Навчання методу опорних векторів достатньо ресурсоємне і має складність порядку $O(N^2 \cdot n)$. Але для класифікації нових об'єктів необхідно зберігати лише частину тренувального набору – ті елементи, для яких α_i відмінні від нуля (опорні вектори). Класифікація має складність $O(SV \cdot n)$, де SV – кількість опорних векторів. У порівняльних експериментах метод опорних векторів показує добрі результати для задачі класифікації в різних прикладних областях [3, 11].

Нейронні мережі.

Нейронні мережі – потужний механізм аналізу даних, який використовується для вирішення задач класифікації і прогнозування [12, 13].

Існує декілька публікацій, в яких описуються експерименти по застосуванню механізму нейронних мереж для задач класифікації електронної пошти, але в них декларуються результати, що дуже відрізняються один від одного. У оглядах, присвячених дослідженню різних методів для задач класифікації текстових документів, нейронні мережі показують не кращі результати [3] наприклад, в порівнянні з методом опорних векторів.

Безумовно, нейронні мережі є цікавим і сильним механізмом для задачі класифікації. Але існують проблеми, які необхідно усунути для ефективного застосування в задачах фільтрації електронної пошти:

– велика складність стадії навчання, яка залежна від топології мережі і вибору методу представлення даних;

- висока чутливість до коректності тренувального набору;
- залежність якості навчання від кількості прикладів тренувального набору (у реальності для даної задачі розмір тренувального набору може відрізнятися в сотні разів).

1.3.2 Оцінка методів класифікації

Для оцінки алгоритмів класифікації, які використовуються в системі фільтрації пошти серверного рівня, були визначені наступні критерії:

Час навчання (побудови моделі). Цей критерій має не таку високу важливість завдяки тому, що, взагалі кажучи, не вимагається миттєвої побудови моделі. Вона може бути, зокрема відкладеною.

Час класифікації нового листа. Найбільш критичний параметр для системи фільтрації пошти серверного рівня, що характеризує продуктивність системи.

Точність класифікації. Основний критерій, що визначає якість класифікації.

Розмір моделі. Оскільки серверна система персональної фільтрації припускає зберігання моделей фільтрації на сервері для кожного користувача системи, то розмір середньої моделі є важливою характеристикою методу

класифікації.

Швидкість донавчання моделі. У системі фільтрації пошти, в якій користувач може регулярно перебудовувати свою модель, донавчаючи її на нових прикладах, цей показник достатньо важливий. Для деяких алгоритмів процес донавчання еквівалентний повному перенавчанню.

Схильність алгоритму до надмірного навчання (overfitting). Виникнення ситуації, коли в результаті багатократного донавчання моделі якість класифікації різко погіршується. Ця проблема, наприклад, актуальна для найбільш поширеного алгоритму класифікації для задачі фільтрації спаму – Naive Bayes. На основі проведеного аналізу розглянутих алгоритмів класифікації, представленого вище, в таблиці 1.1 узагальнені оцінки даних критеріїв:

Таблиця 1.1 – Оцінки алгоритмів класифікації

Критерій	Naive Bayes	FLD	k–nn	SVM	Neural Networks
Час навчання	+	+	–	+	+
Час класифікації	–	–	–	+	+
Точність	+	+	–	+	+
Розмір моделі	+	–	+	+	+
Донавчання моделі	–	–	–	+	+
Overfitting	+	+	–	+	+

1.4 Постановка задачі дослідження

Актуальним є дослідження і розробка алгоритмів і методів, що забезпечують використання персональної моделі класифікації для фільтрації спаму на рівні поштового сервера.

Перспективним напрямком дослідження є розвиток серверних персоніфікованих систем фільтрації пошти, що використовують навчальні алгоритми класифікації на основі методів інтелектуального аналізу даних. Це

твердження будується на наступних фактах:

– серверні системи фільтрації пошти переважають клієнтські, насамперед через універсальність доступу до пошти, скорочення витрат, однаковості рішення, що найбільш важливо для корпоративних користувачів;

– персональна модель фільтрації більш краща ніж загальна модель внаслідок більшої точності і меншої кількості помилок;

– навчальні алгоритми класифікації перевершують традиційні по ряду принципових якостей, таких як якість фільтрації, відсутність оновлень, автономність, тобто незалежність від зовнішніх баз знань, складність «обходу» системи розповсюджувачами спаму.

Алгоритм класифікації повідомлень електронної пошти на основі методу нейронних мереж найбільш збалансований по необхідних сформульованих вище критеріях, і з існуючих алгоритмів класифікації він є найбільш ефективним для використання в серверній системі класифікації пошти. Для реального практичного застосування методу нейронних мереж необхідне вирішення ряду завдань:

– вибір моделі представлення даних;

– скорочення тренувального набору;

– підвищення стійкості методу нейронних мереж до помилок в тренувальному наборі.

Розглянемо їх докладніше.

1. Вибір моделі представлення даних, що забезпечує якість класифікації і швидкість роботи алгоритму. Для алгоритму класифікації однієї з найбільш важливих задач є вибір моделі представлення об'єктів реального світу (електронних листів). Це впливає на більшість важливих критеріїв оцінки алгоритму: швидкість навчання і класифікації, точність, розмір моделі.

2. Скорочення тренувального набору. Обчислювальна складність алгоритму класифікації на основі методу нейронних мереж квадратично залежить від розміру тренувального набору. З іншого боку, розмір і склад тренувального набору визначає розмір і якість побудованої класифікаційної моделі і таким чином впливає на точність і швидкість класифікації. Задача полягає в пошуку компромісу з одного боку між швидкістю навчання, класифікації і розміром

тренувального набору, і з іншого боку – точністю класифікації.

3. Підвищення стійкості методу нейронних мереж до помилок в тренувальному наборі. Однією з умов високого рівня точності навчальних алгоритмів класифікації є коректність тренувального набору. Готуючи тренувальний набір для навчання системи, користувач завжди класифікує листи однозначно – або як спам, або як легальний лист. При цьому листи, по–перше, можуть бути не зовсім типовими для тієї категорії, до якої їх відніс користувач, і, по–друге, користувач може допускати помилки в класифікації. Такі помилки в тренувальному наборі погано впливають на якість моделі і у результаті на якість класифікації нової пошти.

Метою даного дослідження є комплексне рішення для системи класифікації поштових повідомлень серверного рівня, яке засноване на навчальних методах класифікації, що володіє такими перевагами інтелектуальних методів як персоніфікація, автономність, висока точність виявлення спаму при низькій кількості помилково-позитивних помилок. В той же час система класифікації повідомлень повинна забезпечувати продуктивність, достатню для її використання на рівні поштового сервера.

Для досягнення поставленої мети в дипломній роботі необхідно вирішити наступні завдання:

- розглянути задачу класифікації повідомлень електронної пошти;
- розглянути моделі представлення об'єктів для задач класифікації;
- провести аналіз відомих методів та засобів класифікації повідомлень електронної пошти;
- зробити постановку задачі дослідження;
- провести вибір архітектури навчальної системи фільтрації пошти серверного рівня;
- здійснити вибір моделі представлення даних;
- експериментально дослідити метод скорочення простору ознак;
- розробити нейромережевий класифікатор повідомлень електронної пошти;
- розробити архітектуру серверної системи класифікації електронної

пошти;

- розробити користувацький інтерфейс системи;
- провести програмну реалізацію модулів;
- здійснити інтеграцію системи класифікації повідомлень електронної пошти з поштовими системами.

Отже, перший розділ дипломної роботи присвячений розробці і обгрунтуванню основних концепцій серверної системи класифікації пошти, заснованої на навчальних методах класифікації і використовує персональну модель. Представлений аналіз навчальних методів класифікації і їх характеристик. Обгрунтовується вибір базового алгоритму класифікації, формулюються вимоги по його оптимізації і представленню структур даних.

2 АРХІТЕКТУРНІ РІШЕННЯ СИСТЕМИ КЛАСИФІКАЦІЇ ПОВІДОМЛЕНЬ ЕЛЕКТРОННОЇ ПОШТИ

2.1 Вибір архітектури системи

2.1.1 Архітектура серверних систем фільтрації спаму

Розглядаючи архітектуру традиційних систем фільтрації пошти рівня поштового сервера, можна виділити наступні основні компоненти (рисунок 2.1). Основу традиційної системи фільтрації складає загальна експертна база знань, що містить, залежно від типу використовуваних методів виявлення, характеристики, ознаки листів із спамом (IP–адреса, сигнатури, ключові слова і шаблони). Вона може бути як зовнішньою по відношенню до системи, так і внутрішньою. Для підтримки працездатності системи фільтрації бази знань регулярно оновлюються експертами.

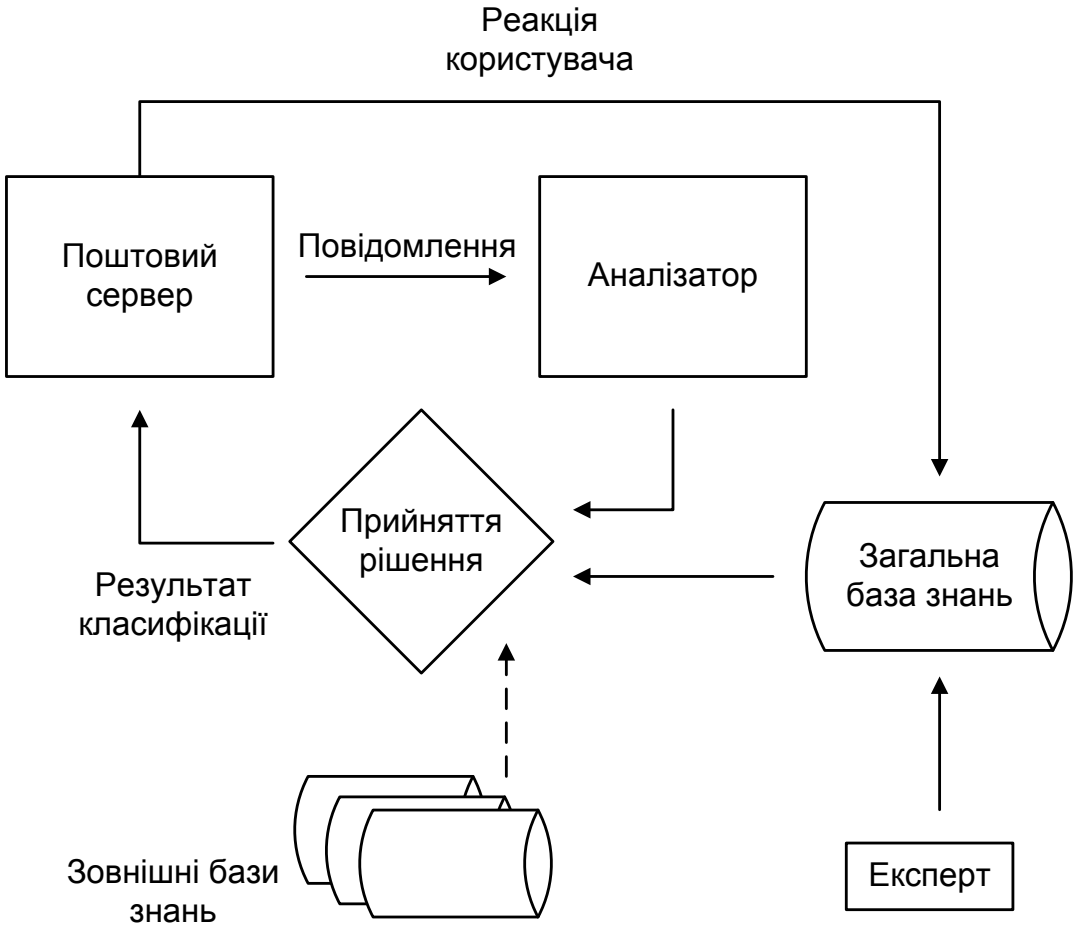

Рисунок 2.1 – Архітектура серверної системи фільтрації пошти, яка заснована на традиційних методах

Більшість поштових серверів мають інтерфейси, які дозволяють одержувати вхідні листи для аналізу сторонньою системою фільтрації пошти. Як правило, традиційні системи одночасно підтримують декілька різних методів виявлення. Таким чином, одержавши лист для класифікації, аналізатор системи виставляє оцінку по кожному методу окремо. Потім остаточне рішення про те, чи є лист спамом, виноситься з врахуванням значущості вагових коефіцієнтів для кожного методу і встановленого порогу загальної оцінки.

У архітектурі традиційних систем обчислювальне навантаження при класифікації листа залежить від методу, що використовується. Зокрема, найбільш ресурсоємний метод – пошук на відповідність евристичним правилам, шаблонам, пошук специфічних трюків і характеристик в листах із спамом.

2.1.2 Архітектура навчальної системи фільтрації пошти серверного рівня

Функціональні стадії навчальної системи класифікації.

Типові стадії роботи системи, заснованої на використанні навчального алгоритму класифікації – це процес первинного навчання системи, класифікація нової пошти і донавчання системи (рисунок 2.2).

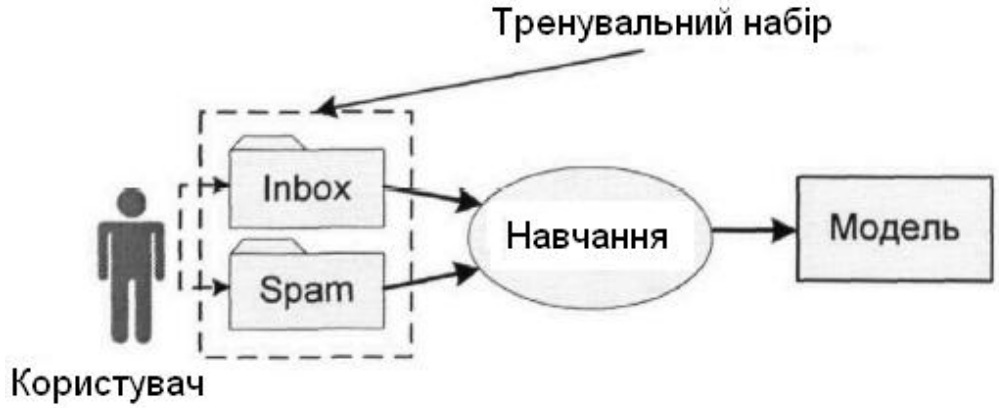

Рисунок 2.2 – Навчальна система фільтрації: первинне навчання системи

Для того, щоб система функціонувала, необхідно провести первинне навчання – побудувати модель класифікації. Для цього користувачем готується тренувальний набір, що складається з його листів – прикладів легального листування і прикладів листів із спамом. Потім ініціюється процес навчання. На підставі аналізу тренувального набору системою будується модель, яка

використовується потім для класифікації нової пошти (рисунок 2.3).

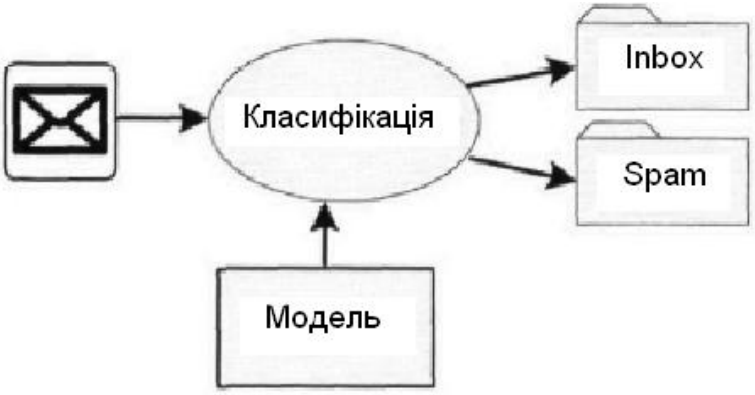

Рисунок 2.3 – Навчальна система фільтрації: класифікація нової пошти

Періодично користувач може проводити донавчання системи, для того, щоб її стан був адекватним поточному стану листування користувача. Нова модель будується на підставі старої моделі і нових листів користувача (рисунок 2.4).

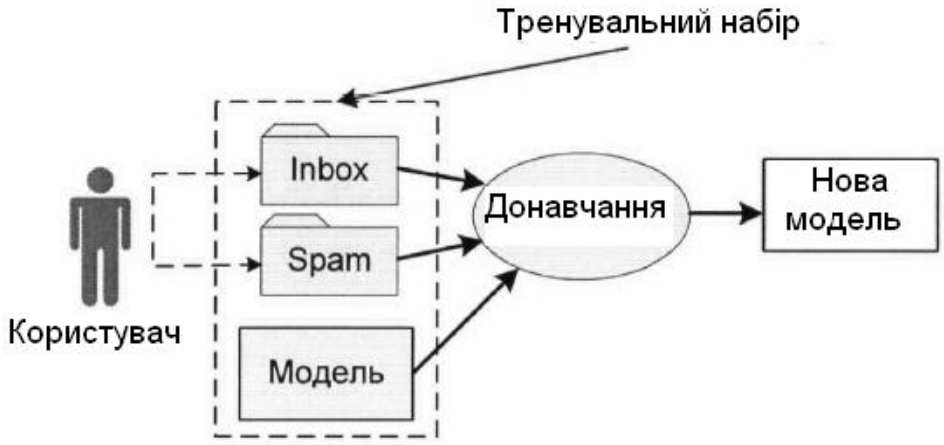

Рисунок 2.4 – Навчальна система фільтрації: донавчання системи

Особливості архітектури.

Для навчальної системи фільтрації електронної пошти рівня поштового сервера були визначені наступні основні вимоги, що пред'являються до її архітектури:

- можливість застосування ресурсоємного алгоритму класифікації на основі алгоритмів машинного навчання;
- необхідний рівень продуктивності системи фільтрації пошти в порівнянні з традиційними системами фільтрації;
- можливість масштабування системи;

- незалежність від платформи реалізації, переносимість;
- можливість інтеграції з різними поштовими серверами;
- необхідний рівень безпеки і конфіденційності.

Враховуючи вимоги, що пред'являються, а перш за все продуктивність і масштабованість, пропонується використання багатоагентної серверної архітектури системи, яка забезпечувала б застосовність щодо ресурсоємного навчального алгоритму класифікації за допомогою ефективного розподілу навантаження, забезпечувала б масштабованість системи і здатність працювати в гетерогенному середовищі.

Архітектурне рішення, що задовольняє представленим концепціям є багатоагентною, багатопотоковою, розподіленою системою, що складається з наступних основних функціональних блоків:

1. Навчальний агент. Витягує із зовнішніх джерел електронні повідомлення і завантажує їх спеціальне представлення в базу даних для подальшого аналізу інтелектуальним агентом і побудови моделі, на підставі якої класифікуються нові листи (рисунок 2.5).

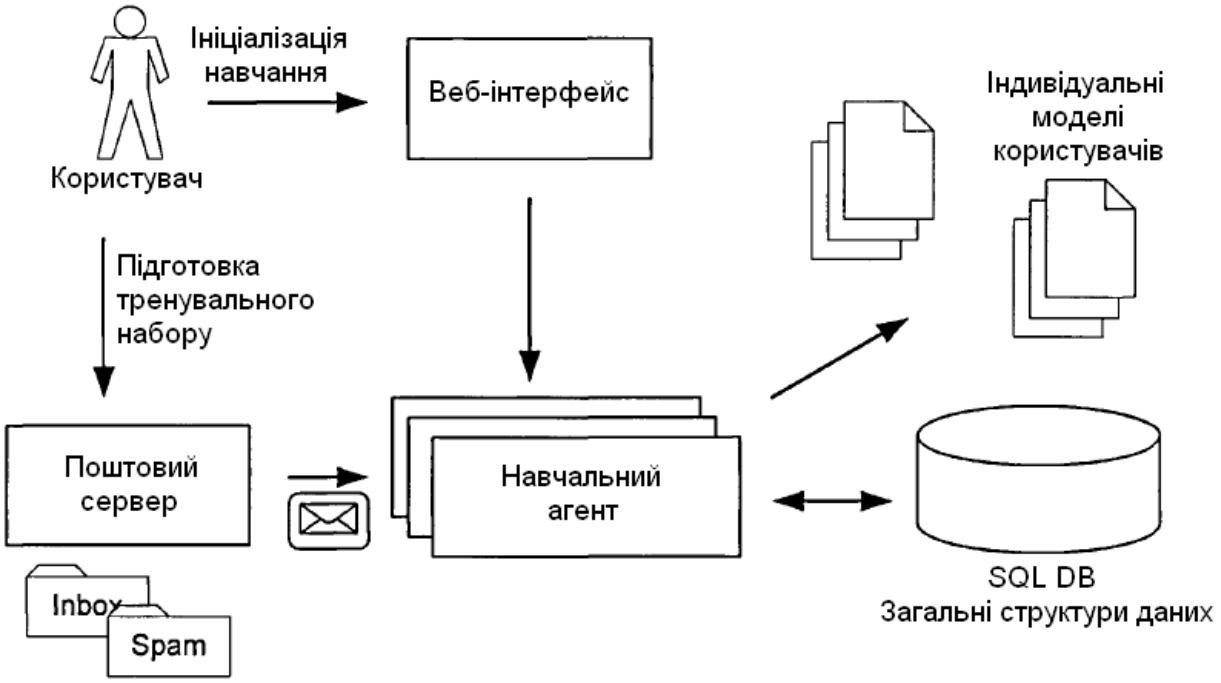

Рисунок 2.5 – Архітектура серверної системи фільтрації пошти, навчальний агент

Класифікуючий агент. Цей компонент системи прив'язується до поштового

сервера і перехоплює листи, адресовані до користувачів, зареєстрованих в системі виявлення спаму. Далі проводиться розбір перехоплених листів і в спеціальному представлені вони передаються для класифікації до інтелектуального агента. Залежно від результатів класифікуючий агент проводить необхідні дії з перехопленим листом.

Сервер бази даних. База даних використовується для зберігання призначених для користувача налаштувань, допоміжних структур даних, які використовуються для аналізу листів, проміжного представлення моделей користувачів.

Завдяки виділенню окремих логічних блоків системи в незалежні агенти спрощується інтеграція системи з різними поштовими серверами. Як правило, поштові сервери мають інтерфейси для підключення зовнішніх фільтрів пошти. Специфіка взаємодії з конкретним поштовим сервером інкапсульована в класифікуючому агентові. Таким чином, можлива інтеграція системи фільтрації з декількома різними поштовими серверами одночасно. При цьому для кожного сервера використовуватиметься свій власний класифікуючий агент (рисунок 2.6).

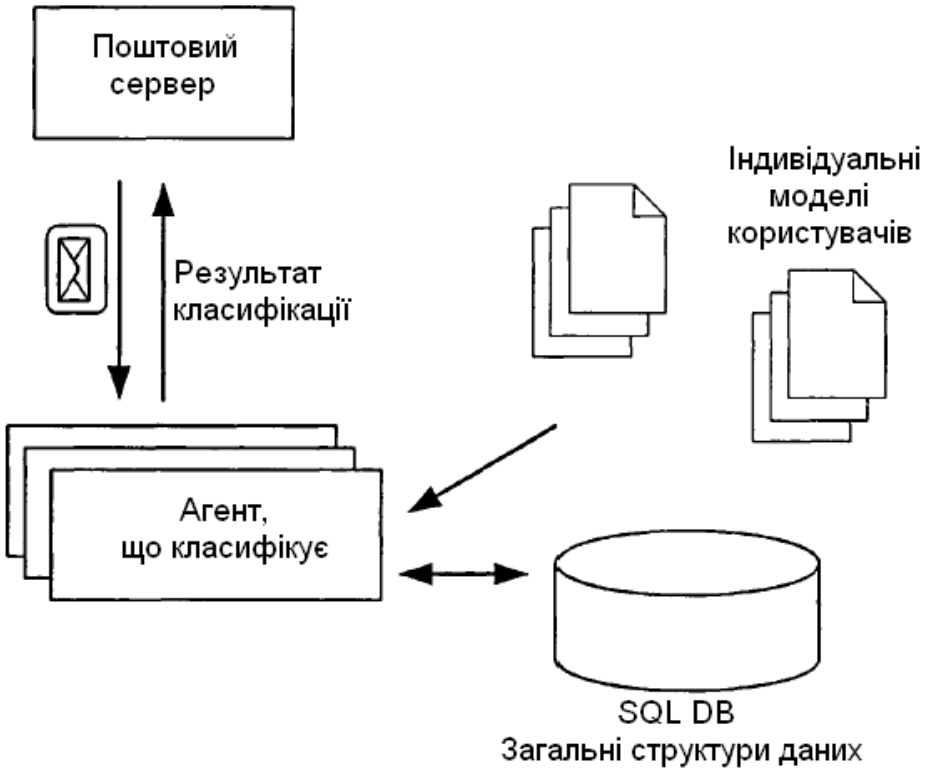

Рисунок 2.6 – Архітектура серверної системи фільтрації пошти, класифікуючий агент

Потенційно різні агенти можуть розташовуватися на різних серверах, що забезпечує масштабованість системи. Взаємодія між агентами здійснюється за допомогою захищеного протоколу обміну даних. Додатково в системі можна виділити комунікаційний агент, який управляє взаємодією, планує задачі і розподіляє навантаження між робочими агентами (класифікуючим і навчальним).

2.2 Модель представлення даних

Об'єкти класифікації – електронні листи є істотно неструктурованими і різнорідними даних. Алгоритми класифікації працюють з формальнішими об'єктами, наприклад з векторами в просторі фіксованої розмірності. Таким чином, необхідно знайти формальне представлення об'єктів класифікації або іншими словами знайти модель представлення об'єктів. Модель представлення об'єкту – формальне представлення сукупності його характеристик, які враховуються алгоритмом класифікації при його обробці.

2.2.1 Вибір моделі представлення даних

При виборі моделі представлення електронних листів розглядаються два суперечливі аспекти, від яких залежить ефективність алгоритму класифікації. По–перше, це точність класифікації, і, по–друге, ресурси обчислювальної системи, що використовуються. Чим більш інформаційно-ємке (і відповідно складніше) представлення електронних листів використовується, тим вища якість їх класифікації, але в той же час це приводить до використання великих обчислювальних ресурсів.

Для представлення листів був вибраний модифікований варіант найбільш поширеної векторної моделі [14]. В термінах цієї моделі набір листів представляється у вигляді матриці: $D=a_{ij}$, в якій a_{ij} це ваговий коефіцієнт i–ї ознаки, що входить в лист k. Розмірність цієї матриці визначається кількістю листів в наборі і кількістю вибраних ознак листа.

Використано два типи ознак, що характеризують електронні листи: текстові ознаки і ознаки, не пов'язані з текстовим вмістом. Текстові ознаки – це всі

текстові лексеми, що входять в лист. Відповідним елементом вектора представлення листа є нормована частота входження даної лексеми в лист. Нетекстові ознаки – це статистичні характеристики для листа, такі як, наприклад середня довжина слова, а також тип, розмір прикріплених файлів.

Для визначення вагових коефіцієнтів ознак була проведена серія експериментів по порівнянню різних підходів, розглянутих в розділі 1. У зв'язку з особливостями реалізації системи класифікації пошти, для забезпечення можливості донавчання алгоритму необхідно використовувати такий метод визначення вагових коефіцієнтів, який використовував би тільки дані про конкретний лист. Тобто дані про всю колекцію, такі як наприклад кількість об'єктів в колекції, кількість повторів конкретної ознаки в колекції, недоступні.

У експериментах по вибору ефективного для даної задачі методу визначення вагових коефіцієнтів якості алгоритму класифікації використовувався метод нейронних мереж. Експерименти проводилися на еталонному наборі електронних листів [15].

У даних умовах тестування, при вибраному алгоритмі класифікації результати тестування відрізнялися трохи для різних методів визначення вагових коефіцієнтів незалежно від їх складності. Таким чином, для досягнення високої продуктивності була вибрана форма визначення вагових коефіцієнтів, що забезпечує необхідну точність класифікації:

$$a_{ij} = f_{ij} / \sqrt[2]{\sum_{i=1}^{m} f_{if}^{\,2}} \qquad (2.1)$$

де a_{ij} – ваговий коефіцієнт *i*–ї ознаки, що входить в лист *j*;

f_{ij} – частота повторення *i*–ї ознаки в листі *j*;

m – кількість ознак в даному листі.

2.2.2 Скорочення розмірності простору ознак

Основна проблема методів класифікації на основі машинного навчання – велика розмірність простору ознак. Методи автоматичного виділення ознак створюють простори розмірності в десятки і сотні тисяч елементів. Сучасні методи класифікації не можуть працювати з об'єктами такого розміру через надзвичайну обчислювальну складність задач, що виникають . Таким чином, виникає необхідність в зменшенні простору ознак.

По оцінках, приведених в роботі [16], за допомогою найбільш складних методів скорочення ознак їх кількість можна скоротити на два порядки без погіршення ефективності класифікації.

Іншим аргументом користь зменшення простору ознак, є проблема перенавчання або «надмірно близької підгонки» (overfitting, overtraining, overlearning). Сенс її полягає в тому, що побудований класифікатор добре задовольняє конкретним прикладам тренувального набору, враховує його незначні деталі, але погано моделює простір об'єктів в цілому завдяки впливу перешкод або шуму в тренувальному наборі. В результаті такий класифікатор добре класифікуватиме тільки об'єкти з тренувального набору, але погано – будь-які нові об'єкти, які відрізняються від тих, на яких він був навчений.

У дипломній роботі був вибраний наступний підхід для первинного формування набору ознак. Всі ознаки діляться на дві групи: текстові і нетекстові ознаки. Нетекстові ознаки визначалися емпірично на підставі експертних оцінок. Їх кількість невелика і фіксована. Зокрема, можливі нетекстові ознаки це:

- наявність, тип, розмір прикріплених файлів;
- кодування, використання форматування;
- статистика, отримана на підставі аналізу тексту листа.

Сенс зменшення розмірності простору ознак – залишити тільки ті, які найбільш інформативні, найбільш значущі для задачі розділення об'єктів по виділених категоріях. Емпірично можна припустити, що:

- якщо одна і та ж ознака зустрічається в об'єктах різних категорій, вона має невисоку інформаційну значущість;
- якщо ознака часто зустрічається в об'єктах певної категорії і при цьому

рідко серед об'єктів решти категорій, то вона має високу інформаційну значущість.

Незалежно від того, який з методів зменшення простору ознак застосовуватиметься надалі, заздалегідь його можна скоротити, використовуючи прості емпіричні правила, такі як:

- виділення основ слова на основі морфологічного аналізу;
- виключення підмножини найбільш часто вживаних слів;
- виключення дуже коротких або дуже довгих слів.

Що стосується виділення основ слова (stemming), то ресурсоємність, реалізація і ефективність цього методу зокрема залежать від природної мови документів набору. У зв'язку з цим застосовність його для задачі класифікації електронних повідомлень в багатомовному середовищі представляється сумнівною.

Видалення найбільш часто вживаних слів – простий і ефективний метод скорочення розмірності простору ознак. Він грунтується на твердженні, що такі слова як прийменники, артиклі, частини не несуть інформаційної значущості для ідентифікації документа. Цей метод також пов'язаний з природною мовою документів.

Видалення дуже довгих і дуже коротких слів так само грунтуються на емпіричному висновку про рівномірний розподіл таких слів серед документів, що належать будь–якій категорії.

Для попереднього скорочення ознак в роботі були застосовані наступні методи:

- виключення найбільш часто вживаних слів (на підставі заздалегідь певного словника);
- виключення слів, які довші і коротші заданих меж.

У формальних процедурах зменшення простору ознак застосовуються методи лінійної алгебри і теорії інформації. Їх можна розділити на дві категорії:

- методи вибору ознак з існуючого набору ознак;
- методи репараметризації або відображення існуючого простору ознак в інший, меншого розміру.

До першої категорії відносяться методи, які оцінюють «важливість» або значущість кожної ознаки для задачі класифікації. Найбільш простий, але в той же час ефективний метод – результуючий простір складають ознаки, які зустрічаються не менше чим в заздалегідь визначеній кількості документів навчальної вибірки.

Існує велика кількість методів, які на основі аналізу статистичних розподілів ознак в документах тренувального набору кожної категорії, в абсолютних одиницях оцінюють інформаційну значущість кожної ознаки. Отримавши таку оцінку і вибравши поріг відсікання, в результуючому просторі ознак можна залишити лише найбільш важливі, такі, що «розділяють» документи різних категорій.

Серед великої кількості таких методів, як найбільш ефективні, виділяють в [16] наступних: Information Gain, X^2–статистика, Mutal Information, Term Strength. Проте, результати у великій мірі залежать від алгоритму класифікації, що використовується і навчального набору.

Information Gain – характеристика кожної ознаки, що показує наскільки присутність або відсутність даної ознаки в документі навчального набору впливає на приналежність її до тієї або іншої категорії:

$$IG(x) = -\sum P(y_j)\log P(y_j) + P(x)\sum P(y_j|x)\log P(y_j|x) + P(\bar{x})\sum P(y_j|\bar{x})\log P(y_j|\bar{x}) \quad (2.2)$$

Хі–квадрат статистика оцінює ступінь залежності даної ознаки і категорії. Для нашої задачі з двома категоріями ця оцінка виявляється симетричною:

$$\chi^2 = \text{◐}w, y_i \text{◑} = \frac{(a+b+c+d)(ad-bc)^2}{(a+b)(c+d)(b+d)(a+c)}; \quad (2.3)$$

де a – кількість документів з категорії y_i, які містять ознаку w;

b – кількість документів які містять ознаку w, але не належать

категорії y_i;

c – кількість документів з категорії y_i, які не містять ознаку w;

d – кількість документів, які не належать категорії y_i і не містять ознаку w.

Були проведені експерименти з трьома найбільш ефективними [17] методами вибору набору оптимальних ознак на основі оцінок їх інформаційної значущості: Information Gain, хі–квадрат статистика і вибір ознак по частоті їх повторів. Було показано [17], що методи вибору оптимальних ознак мають схожі обчислювальну складність і ефективність. Проведені експерименти показують, що ці методи недостатньо точні для істотного скорочення ознак, але, проте, вельми ефективні для первинного грубого відбору значущих ознак. З розглянутих методах найкращих результатів вдалося досягти для методу вибору ознак по частоті повторів.

Окремий клас методів скорочення простору ознак – методи репараметризації. Це процес побудови нового набору ознак як комбінації або трансформації оригінального набору.

Один з таких методів, використаний в експериментах в даній роботі –скрите семантичне індексування (LSI, Latent Semantic Indexing). Він заснований на припущенні, що існують приховані залежності у використанні слів або конструкцій із слів в документах і статистичні методи дозволяють виявити ці структури [18]. Для побудови нового набору ознак використовується сингулярне розкладання матриці документів тренувального набору [19].

Загальна схема роботи складається з двох кроків – перший: побудова матриці по колекції тренувальних документів – $A=(a_{ik})$, другий: пониження розмірності. Для цього в LSI зазвичай використовується сингулярне розкладання матриці A.

Нехай є матриця, що представляє набір документів A, розміру (M*N), тоді вона розкладається:

$$A = U \sum V^T, \tag{2.4}$$

де *U(M*R)* і *V(N*R)* складаються з ортонормованих стовпців;

$\sum(R*R)$ – діагональна матриця, що складається з сингулярних значень.

$R \le \min(M, N)$ – ранг матриці *A*. Тоді якщо упорядкувати сингулярні числа за збільшенням і вибрати *K* максимальних, то матриця

$$A_K = U_{|K} \sum_K V_K{}^T, \tag{2.5}$$

де $\sum_K(K*K)$ містить вибрані сингулярні числа, а $U_{|K}$ і $V_K{}^T$ отримуються при видаленні стовпців і рядків, не відповідних числам стовпців і рядків з U і V^T.

У матриці A_K – більше виражені залежності, які є в *A*, в той же час віддаляється шум і багатозначність використання слів. Оскільки розмір *K* багато менший числа унікальних слів, незначні відмінності в термінології документів ігноруватимуться. Слова, які з'являлися в схожих документах, будуть знаходитися поряд в просторі розмірності *K*, навіть якщо в початковому документі вони не попадалися в одному документі одночасно. Більш того, документи, множини використовуваних яких слів не перетинаються, можуть бути схожі.

Косинус кута між рядками матриці A_K, або відповідного кута між рядками $U_{|K} \sum_K$ відображає, на скільки два слова мають схожий смисл в тренувальному наборі. Якщо косинус кут між словами рівний 1, то слова мають практично однаковий смисл, якщо рівний 0, то смисл слів різний.

Таким же чином порівнюється "схожість" документів – по косинусу кута між стовпцями A_K або рядками $U_K \sum_K$.

Для порівняння *i*–го слова і *k*–го документа вважається косинус кута між *i*–м рядком $U_{|K} \sum_K{}^{1/2}$ і *k*–м рядком $V_{|K} \sum_K{}^{1/2}$.

Для порівняння нового документа з документом з тренувального набору будується вектор цього документа *d*, по якому знаходиться його представлення $\tilde{d}$:

$$\tilde{d} = d^T U_K \sum_K{}^{-1}. \quad (2.6)$$

Це представлення використовується як рядок V_K, тобто для порівняння документа з документами з тренувального набору шукається косинус кута між $\tilde{d} \sum_K$ і відповідного рядка $V_K \sum_K$.

За допомогою сингулярного розкладання матриці документів, знаходиться задана кількість сингулярних векторів, за допомогою яких для кожного документа будується вектор, що представляє його. Таким чином, розмірність представлення можна вибирати, використовуючи тільки те число сингулярних значень, яка необхідна розмірність. Сингулярне розкладання будується на тренувальному наборі, потім знаходиться матриця, що перетворює вектор довільного документа в псевдо–документ. Потім отримана матриця використовується для тренування класифікатора. За допомогою тієї ж перетворюючої матриці будується представлення тестового вектора в новому просторі.

Складність методу LSI, а фактично складність сингулярного розкладання матриці складає $O(d^2 N) + O(d^3)$ [29], де d – розмірність вектора ознак, N – кількість документів. Таким чином, вона перевищує складність самого вибраного базового алгоритму класифікації. У експериментах з реальними наборами листів розмірність простору ознак навіть після його попереднього скорочення складала декілька десятків тисяч. Час роботи методу LSI з даними такої розмірності дуже великий.

Таким чином, методи вибору оптимального набору ознак на основі їх інформаційної значущості недостатньо точні при значному (на два порядки) скороченні кількості ознак. Це підтверджують експерименти, що проводилися в дослідженні, результати яких представлені нижче. З іншого боку, методи репараметризації володіють дуже високою обчислювальною складністю і практично непридатні для вибору оптимального набору ознак в даній задачі.

Для вирішення цієї задачі був запропонований комбінований підхід, суть якого полягає в суперпозиції двох методів. Спочатку набір ознак скорочується одним з методів вибору простору ознак. Потім до істотно зменшеного набору

застосовується алгоритм репараметризації. Такий підхід дозволив скоротити простір ознак на два порядки при цьому, практично не погіршивши точність класифікації.

2.2.3 Експериментальне обгрунтування методу скорочення простору ознак

Проведена серія експериментів, в яких використовувалися представлені методи з різними параметрами. В якості тренувального і тестового наборів документів використовувався загальнодоступний архів електронних листів, який запропонований розробниками фільтру пошти SpamAssassin [15]. Весь набір листів був розбитий на декілька частин, кожна з яких послідовно використовувалася як тестова, а решта всі, що залишилися – як навчальні. Потім результати експериментів для кожної з частин об'єднувалися. До навчального набору застосовувалися досліджувані методи зменшення простору ознак з різними параметрами, потім для кожного експерименту використовувався один і той же алгоритм класифікації на основі нейронних мереж. Для порівняння результатів різних експериментів застосовувалася стандартна процедура оцінки алгоритмів, що мають помилки першого і другого роду з використанням ROC–кривих.

На рисунку 2.7 верхній графік – результат роботи методу нейронних мереж на всьому наборі ознак без скорочення. Його можна розглядати як еталонний. Два інших графіки представляють два методи скорочення ознак, для яких була вибрана однакова межа кількості відібраних ознак.

Самий нижній графік – вибір ознак за допомогою методу оцінки інформаційної значущості. Як видно, результати класифікації значно погіршали. Другий графік – результат пропонованого рішення. Як видно, результати класифікації відрізняються трохи від результатів з використанням всього простору ознак. Таким чином, вдалося скоротити простір ознак на два порядки, практично не погіршивши якість класифікації.

Основна проблема при розробці статистичних методів класифікації –велика розмірність простору ознак. Існуючі методи скорочення кількості ознак погано підходять для задачі класифікації електронної пошти. Методи вибору

оптимальних ознак недостатньо точні при значному скороченні кількості ознак, а методи репараметризації володіють дуже високою обчислювальною складністю і при розмірності простору ознак в декілька тисяч елементів практично непридатні.

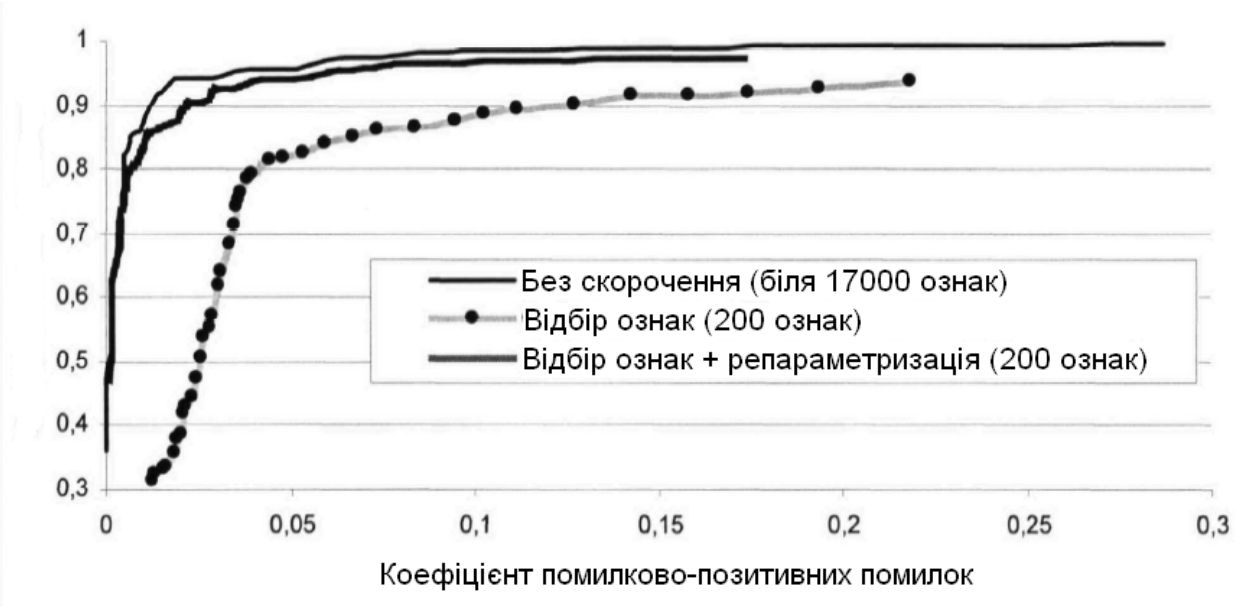

Рисунок 2.7 – Результати порівняльного тестування методів скорочення простору ознак

Для вирішення цієї проблеми був запропонований комбінований підхід, суть якого полягає в суперпозиції двох методів. Спочатку набір ознак скорочується одним з методів вибору простору ознак. Потім до істотно зменшеного набору застосовується алгоритм репараметризації. Такий підхід дозволив скоротити простір ознак на два порядки при цьому практично не погіршивши точність класифікації.

2.3 Нейромережевий класифікатор повідомлень електронної пошти

2.3.1 Обгрунтування вибору архітектури нейронної мережі

Однорівнева НМ – однорівневий персептрон (ОП) – це нейронна мережа, яка складається з єдиного рівня нейронних елементів. Однорівневі НМ формують лінійну розділяючу поверхню, що обмежує коло вирішуваних за допомогою таких мереж задач.

Основним елементом НМ є формальний нейрон, який здійснює операцію

нелінійного перетворення суми добутків вхідних сигналів на вагові коефіцієнти:

$$y = F(\sum_{i=1}^{n} \omega_i x_i), \qquad (2.7)$$

де $X=(x_1, x_2, ..., x_n)$ – вектор вхідного сигналу;

$W = (\omega_1, \omega_2, \ldots, \omega_n)$ – ваговий фактор;

F – оператор нелінійного перетворення, який називається функцією активації нейронного елемента.

Схема нейронного елемента зображена на рисунку 2.8 і складається з суматора та блока нелінійного перетворення F. Кожному і–му входу нейрона відповідає ваговий коефіцієнт ω_i (синапс), який характеризує силу синаптичного зв'язку.

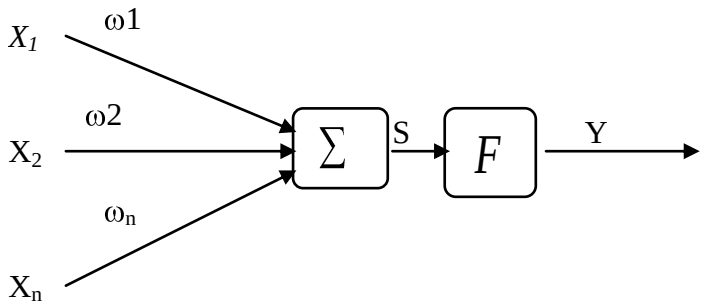

Рисунок 2.8 – Нейронний елемент

Сума добутків вхідних сигналів на вагові коефіцієнти називається зваженою сумою.

В якості оператора нелінійного перетворення можуть використовуватися різні функції, які визначаються у відповідності з вирішуваною задачею і типом НМ. Нехай T – поріг нейронного елемента, який характеризує розміщення функції активації по осі абсцис. Тоді зважена сума буде виглядати наступним чином:

$$S = \sum_{i=1}^{n} \omega_i x_i - T \tag{2.8}$$

Найбільш розповсюдженими функціями активації є наступні:

- лінійна функція;
- порогова функція;
- лінійна обмежена функція;
- модифікована порогова функція;
- сигмоїдна функція;
- біполярна сигмоїдна функція;
- гіперболічний тангенс;
- радіально–базисна функція.

Рівнем НМ називається множина нейронних елементів, на які в кожен такт часу паралельно надходить інформація від інших елементів мережі. Відповідно до топології однорівневої НМ (див. рисунок 2.8) вихідне значення вихідного нейронного елемента можна представити наступним чином [20–22]:

$$y = F(S) = F(\sum_{i=1}^{n} \omega_i x_i - T). \tag{2.9}$$

Самоадаптація та самоорганізація НМ досягається в процесі їх навчання, в ході якого відбувається визначення синаптичних зв'язків між нейронними елементами. Правила навчання визначають, як змінюються вагові коефіцієнти у відповідь на вхідну дію. Найвідомішими правилами навчання є наступні:

- правило навчання Хебба, яке є основою багатьох методів навчання НМ;
- процедура навчання Розенблатта;
- правило навчання Відроу–Хоффа.

Для прискорення процедури навчання використовується метод градієнтного сходження з обчисленням адаптивного кроку навчання $\alpha(t)$. Адаптивний крок навчання – це такий крок, який цілеспрямовано вибирається на кожному етапі

алгоритму таким чином, щоб мінімізувати середньоквадратичну похибку мережі. Зокрема, для однорівневої НМ такий адаптивний крок навчання буде обчислюватися наступним чином [20]:

$$\alpha(t) = \frac{1}{1 + \sum_i x^2(t)}. \tag{2.10}$$

При використанні методу вікон НМ складається з одного вихідного нейронного елементу та p розподілених нейронів. Така модель НМ відповідає лінійній авторегресії і описується наступним виразом:

$$\overline{x(n)} = \sum_{k=1}^{p} \omega_k x(n - p + k - 1), \tag{2.11}$$

де $\omega_k, k = \overline{1, p}$ – вагові коефіцієнти НМ;

$\overline{x(n)}$ – оцінка значення ряду $x(n)$ в момент часу n.

Похибка класифікації визначається як

$$e(n) = x(n) - \overline{x(n)}. \tag{2.12}$$

Багаторівневий персептрон (БП) являє собою нейронну мережу з одним вхідним, одним вихідним та одним або більше прихованими рівнями. Архітектура БП складається з множини рівнів нейронних елементів (рисунок 2.9).

Вхідний рівень (input layer) нейронних елементів виконує розподільні функції. Вихідний рівень (output layer) нейронів служить для обробки інформації від попередніх рівнів та видачі результатів. Рівні нейронних елементів, що розміщені між вхідним і вихідним рівнем, називаються проміжними або прихованими (hidden layers). Як і вихідний рівень, приховані рівні є рівнями обробки даних. Вихід кожного нейронного елемента попереднього рівня НМ з'єднаний синаптичними зв'язками з усіма входами нейронних елементів наступного рівня. Таким чином, топологія БП є однорідною та регулярною.

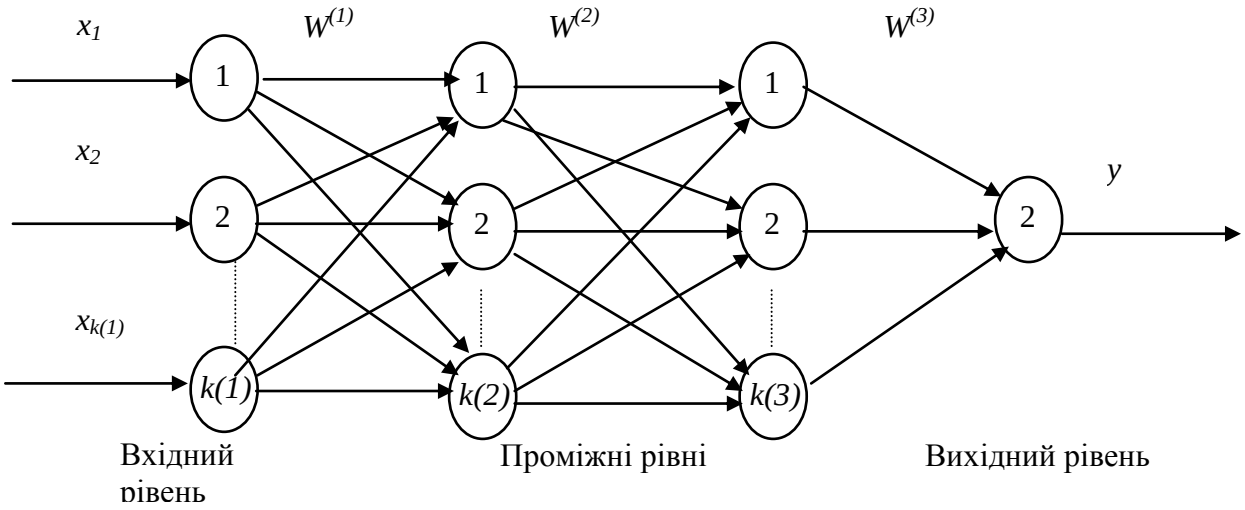

Рисунок 2.9 – Архітектура багаторівневого персептрона

В якості функції активації нейронних елементів тут, як правило, використовуються гіперболічний тангенс або сигмоїдна функція. Нехай $W^{(i)}$ – матриця вагових коефіцієнтів і-го рівня БП. Тоді для НМ з двома прихованими рівнями вихідне значення Y можна визначити наступним чином:

$$Y = F(F(F(XW^{(1)})W^{(2)})W^{(3)}), \tag{2.13}$$

де $X = (x_1, x_2, \ldots, x_n)$ – вектор–стрічка вхідних сигналів;

F – оператор нелінійного перетворення.

Загальна кількість синаптичних зв'язків БП дорівнює:

$$V = \sum_{i=1}^{p-1} k(i) \cdot k(i+1), \tag{2.14}$$

де p – загальна кількість рівнів НМ;

k(i) – кількість нейронних елементів в і–му рівні.

Кількість рівнів в багаторівневій НМ характеризує, яким чином вхідний простір може бути розбитий на підпростори меншої розмірності.

Найефективнішим алгоритмом навчання таких багаторівневих НМ є алгоритм зворотного розповсюдження помилки (backpropagation algorithm) та

його наступні модифікації [20, 21]. Даний алгоритм був запропонований у 1986 році рядом авторів незалежно один від одного. Він є ефективним засобом навчання НМ і являє собою наступну послідовність кроків:

1) Задається крок (швидкість) навчання α $(0<\alpha<1)$ і бажана середньоквадратична похибка НМ E_m.
2) Випадковим чином ініціалізуються вагові коефіцієнти та порогові значення НМ.
3) Послідовно подаються значення (образи) з навчальної вибірки на вхід НМ. При цьому для кожного вхідного значення виконуються наступні дії:
   a) Здійснюється фаза прямого розповсюдження вхідного значення по НМ. При цьому обчислюється вихідна активність усіх нейронних елементів мережі:

$$y_i = F(\sum_i \omega_{ij} \cdot y_i - T_j), \tag{2.15}$$

   де індекс j характеризує нейрони наступного рівня відносно рівня i.
   b) Здійснюється фаза зворотного розповсюдження сигналу, в результаті якої визначається похибка γ_j, j=1,2,... нейронних елементів для усіх рівнів мережі. При цьому відповідно для вихідного і прихованого рівнів:

$$\gamma_j = y_j - t_j, \tag{2.16}$$

$$\gamma_j = \sum_i \gamma_i F'(S_i)\omega_{ji}\ . \tag{2.17}$$

В останньому виразі індекс і характеризує нейронні елементи наступного рівня відносно рівня j.

с) Для кожного рівня НМ відбувається зміна вагових коефіцієнтів та порогів нейронних елементів [20, 21]:

$$\omega_{ij}(t+1) = \omega_{ij}(t) - \alpha\gamma_j F'(S_j) y_j, \tag{2.18}$$

$$T_j(t+1) = T_j(t) + \alpha\gamma_j F'(S_j) \tag{2.19}$$

4) Обчислюється сумарна середньоквадратична похибка НМ:

$$E = \frac{1}{2}\sum_{k=1}^{L}\sum_{j}(y_j^k - t_j^k)^2, \tag{2.20}$$

де L – розмірність навчальної вибірки.

5) Якщо $E > E_m$, то відбувається перехід на крок 3 даного алгоритму. В іншому випадку алгоритм зворотного розповсюдження похибки закінчується.

Таким чином, даний алгоритм функціонує до тих пір, поки сумарна середньоквадратична похибка НМ не стане меншою від заданої, тобто $E \le E_m$.

Алгоритм зворотного розповсюдження помилки, в основі якого лежить градієнтний метод, створює ряд проблем при навчанні багаторівневих НМ. До таких проблем можна віднести наступні [20]:

- Невідомий вибір кількості рівнів і кількості нейронних елементів в рівні для багаторівневих мереж.
- Повільне сходження градієнтного методу з постійним кроком навчання.
- Проблема вибору потрібної швидкості навчання α. Так, занадто мала швидкість навчання збільшує час навчання і приводить до скочування НМ в локальний мінімум. Більша швидкість навчання може привести до пропуску глобального мінімуму і зробити процес навчання таким, що не сходиться.
- Градієнтний метод не розрізняє точок локального і глобального

мінімумів.

– Вплив випадкової ініціалізації вагових коефіцієнтів НМ на пошук мінімуму функції середньоквадратичної похибки.

Дані недоліки легко усуваються, використовуючи подальші модифікації даного алгоритму.

Рекурентною НМ (РНМ) називається така мережа, в якій виходи нейронних елементів наступних рівнів мають синаптичні зв'язки з нейронами попередніх рівнів (рисунок 2.10). Це дозволяє враховувати результати перетворень НМ інформації на попередньому етапі для обробки вхідного вектора на наступному етапі функціонування мережі [20].

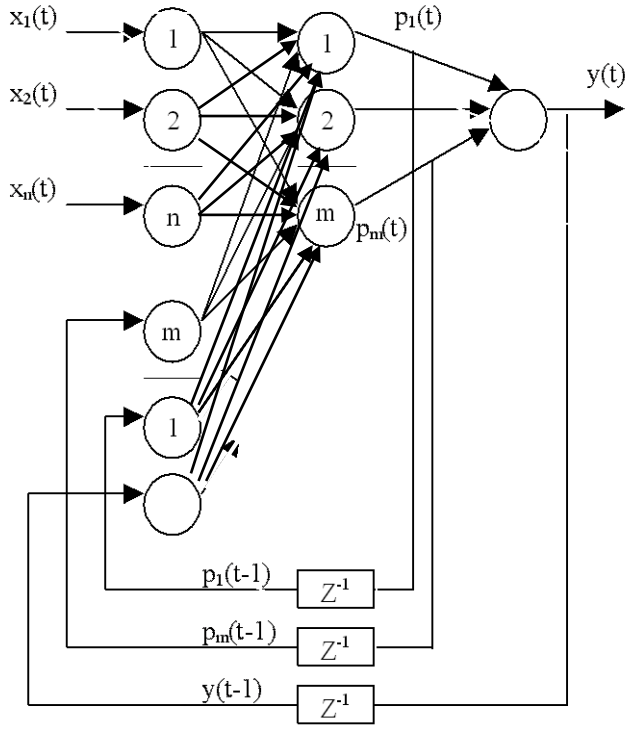

Рисунок 2.10 – Архітектура рекурентної НМ

У даній нейромережі виходи нейронів прихованого та вихідного рівнів з'єднані з входами нейронів проміжного рівня за допомогою контекстних нейронів. Вони розподіляють вихідні дані на нейрони проміжного рівня. Кількість контекстних нейронів дорівнює сумі нейронів вихідного та проміжного рівнів. В якості прихованого рівня використовуються нейронні елементи з сигмоїдною функцією активації, а для вихідного нейрона використана лінійна.

Зважена сума і–го нейрона проміжного рівня визначається наступним чином:

$$S_i(t) = \sum_{j=1}^{n} w_{ji} x_j(t) + w_i y(t-1) + \sum_{l=1}^{m} w_{li} p_l(t-1) - T_i, \quad (2.21)$$

де w_{ji} – ваговий коефіцієнт між j–им нейроном вхідного рівня та і–им нейроном прихованого рівня;

x_j – вхідний образ, що подається на j–ий нейрон розподільчого рівня;

w_i – ваговий коефіцієнт між вихідним нейроном та і–тим нейроном прихованого рівня;

y(t–1) – вихідне значення нейромережі, отримане на попередньому етапі;

w_{li} – ваговий коефіцієнт між виходом l –го нейрона прихованого рівня та і–тим нейроном прихованого рівня;

$p_l(t–1)$ – вихідна активність l –го нейрона прихованого рівня, отримана на попередньому етапі;

T_i – порогове значення і –го нейрона прихованого рівня.

Вихідна активність j –го нейрона прихованого рівня нейромережі дорівнює:

$$p_j(t) = \frac{1}{1 + e^{-S_j(t)}}. \quad (2.22)$$

Вихідне значення НМ розраховується за допомогою формули:

$$y(t) = \sum_{i=1}^{m} p_i(t) v_i - T, \quad (2.23)$$

де v_i – ваговий коефіцієнт між і–тим нейроном проміжного рівня та вихідним нейроном;

T – межа вихідного нейрона.

Для навчання РНМ використовується алгоритм зворотного розповсюдження

помилки [20–22].

Середньоквадратична похибка для одного вхідного вектора визначається так:

$$E(t) = \frac{1}{2}(y(t) - x(t+1))^2, \tag{2.24}$$

де x(t+1) – значення часового ряду в момент часу t+1.

Для налаштування параметрів навчання нейронної мережі використовуються наступні формули [20]:

$$\Delta v_i(t+1) = -A(y(t) - x(t+1))p_i(t), \tag{2.25}$$

$$\Delta T(t+1) = A(y(t) - x(t+1)), \tag{2.26}$$

$$\Delta w_{ji}(t+1) = -A(y(t) - x(t+1))v_i p_i(t)(1 - p_i(t))x(t - j + 1), \tag{2.27}$$

$$\Delta w_i(t+1) = -A(y(t) - x(t+1))v_i p_i(t)(1 - p_i(t))y(t-1), \tag{2.28}$$

$$\Delta w_{li}(t+1) = -A(y(t) - x(t+1))v_i p_i(t)(1 - p_i(t))p_l(t-1), \tag{2.29}$$

$$\Delta T_i(t+1) = A(y(t) - x(t+1))v_i p_i(t)(1 - p_i(t)) \tag{2.30}$$

Для прискорення навчання використовується адаптивний крок навчання [20]. Для вихідного рівня він буде дорівнювати:

$$A(t) = \frac{1}{\left(1 + \sum_{i=1}^{m} p_i^2(t)\right)}. \tag{2.31}$$

А для нейронів прихованого рівня:

$$A(t) = \frac{4}{R^2 \sum_{i=1}^{m} v_i^2 p_i(t)\left(1 - p\ \ (t)\right)}, \tag{2.32}$$

де R знаходиться за допомогою формули:

$$R = \left(1 + \sum_{j=1}^{n} x^2(t - j + 1) + y^2(t-1) + \sum_{l=1}^{m} p_l^2(t-1)\right). \tag{2.33}$$

Розглянувши основні співвідношення для рекурентної нейронної мережі, можна сформулювати алгоритм навчання, який складається з наступних кроків:

1. В початковий момент часу t=0 всі контекстні нейрони встановлюються в нульовий стан, тобто їх вихідні значення рівні нулю;

2. На вхід подається тренувальний образ і відбувається пряме розповсюдження його в НМ згідно формул (2.21) – (2.23);

3. У відповідності з алгоритмом зворотного розповсюдження помилки проводиться модифікація вагових коефіцієнтів і порогових значень нейронних елементів за виразами (2.24) – (2.30);

4. Встановлюється t = t+1 і здійснюється перехід до пункту 2.

Навчання РНМ проводиться до тих пір, поки сумарна середньоквадратична помилка мережі не стане меншою заданої.

2.3.2 Алгоритм навчання нейронної мережі

На основі математичного апарату функціонування НМ побудований алгоритм навчання нейронної мережі (рисунок 2.11).

Алгоритм навчання нейронної мережі складається з наступних етапів:

– Ініціалізація вагових коефіцієнтів та порогових значень НМ.

– Обчислення виходу нейронів мережі.

– Обчислення помилки нейронів всіх рівнів мережі.

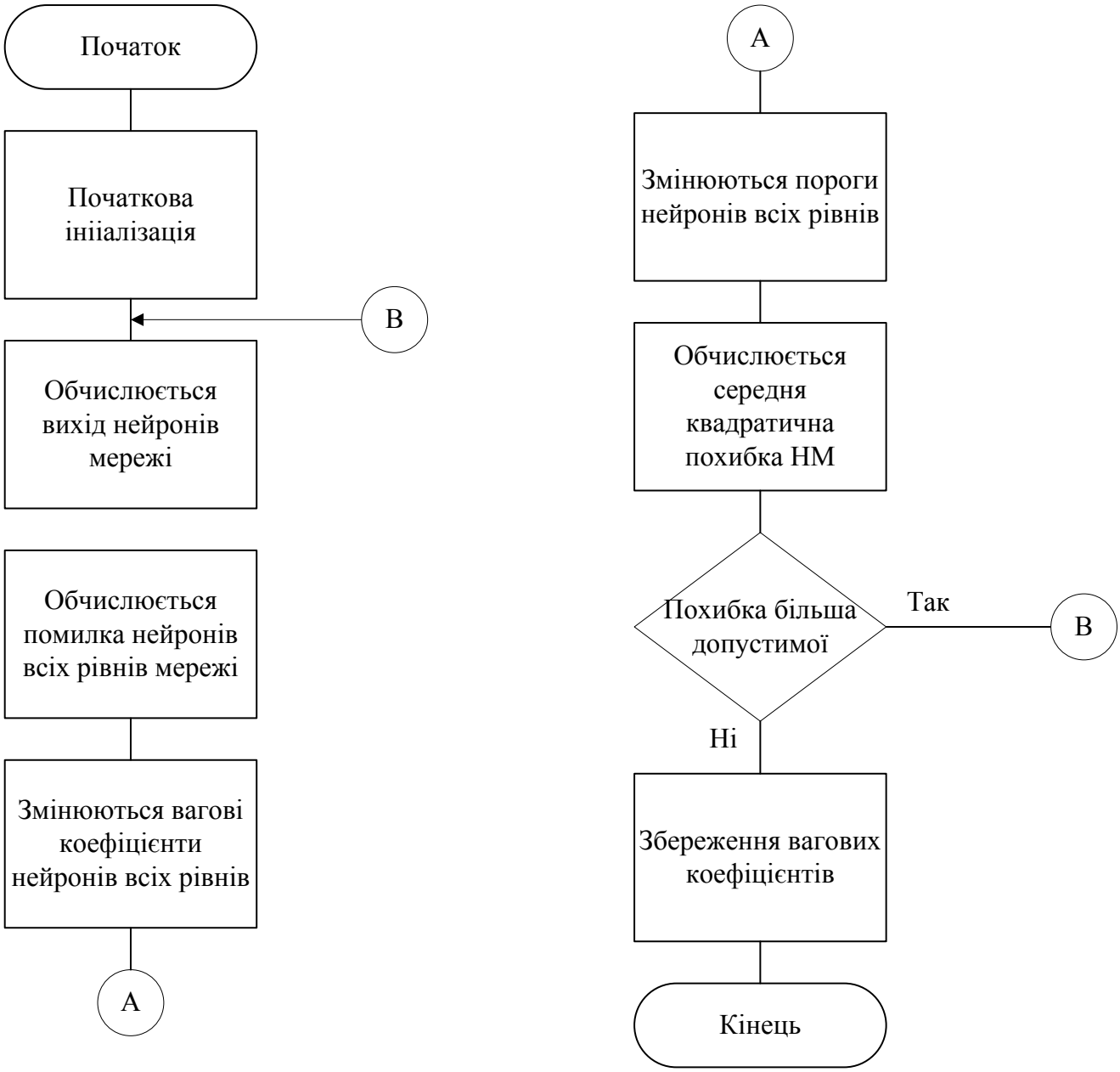

Рисунок 2.11 – Алгоритм навчання нейтронної мережі

– Зміна вагових коефіцієнтів нейронів всіх рівнів.

– Зміна порогів нейронів всіх рівнів.

– Обчислення середньої квадратичної похибки НМ.

– Перевірка чи похибка більша допустимої.

– Збереження вагових коефіцієнтів.

Отже, даний алгоритм функціонує до тих пір, поки сумарна середньоквадратична похибка НМ не стане меншою від заданої.

В даному розділі проведений вибір архітектури системи, яка забезпечує застосування навчального алгоритму класифікації за допомогою ефективного розподілу навантаження і забезпечує масштабованість системи і здатність працювати в гетерогенному середовищі. Розглядається вибір моделі представлення даних та результати експериментів, що обгрунтовують ефективність даного підходу. Проведений вибір нейтронної мережі та розроблений алгоритм її навчання для класифікації повідомлень електронної пошти.

3 РЕАЛІЗАЦІЯ СЕРВЕРНОЇ СИСТЕМИ КЛАСИФІКАЦІЇ ЕЛЕКТРОННОЇ ПОШТИ

3.1 Розробка архітектури системи

У другому розділі були розглянуті загальні концепції побудови навчальної системи фільтрації пошти серверного рівня. Були визначені основні якості системи, такі як загальна продуктивність і масштабованість, а також базова архітектура, яка представляє собою багатоагентну, багатопотокову, розподілену систему.

Експериментальна серверна система фільтрації, побудована на основі загальної структури системи, розглянутої в другому розділі, складається з наступних основних функціональних блоків (рисунок 3.1):

– класифікуючий агент - надає інтерфейси для фільтрації пошти для зовнішнього по відношенню до системи поштового сервера, виконує попередній аналіз листа і передає його на класифікацію агентові-процесору;

– навчальний агент - одержує листи для навчання, виконує їх попередній аналіз, будує тренувальний набір і передає його для побудови моделі класифікації агентові-процесору;

– агент-процесор - безпосередньо реалізує алгоритми навчання і класифікації, додатково виконує оптимізацію тренувального набору;

– комунікаційний агент - виконує задачі взаємодії і інтеграції інших блоків системи.

Загальні структури даних, словник лексем, персональні настройки користувачів зберігаються в базі даних.

Далі розглянемо детально кожен з блоків системи, особливості їх реалізації, функціонування і взаємодії.

Комунікаційний агент системи реалізований як один або декілька веб-серверів, за допомогою яких здійснюється взаємодія між різними агентами системи і взаємодія з користувачем.

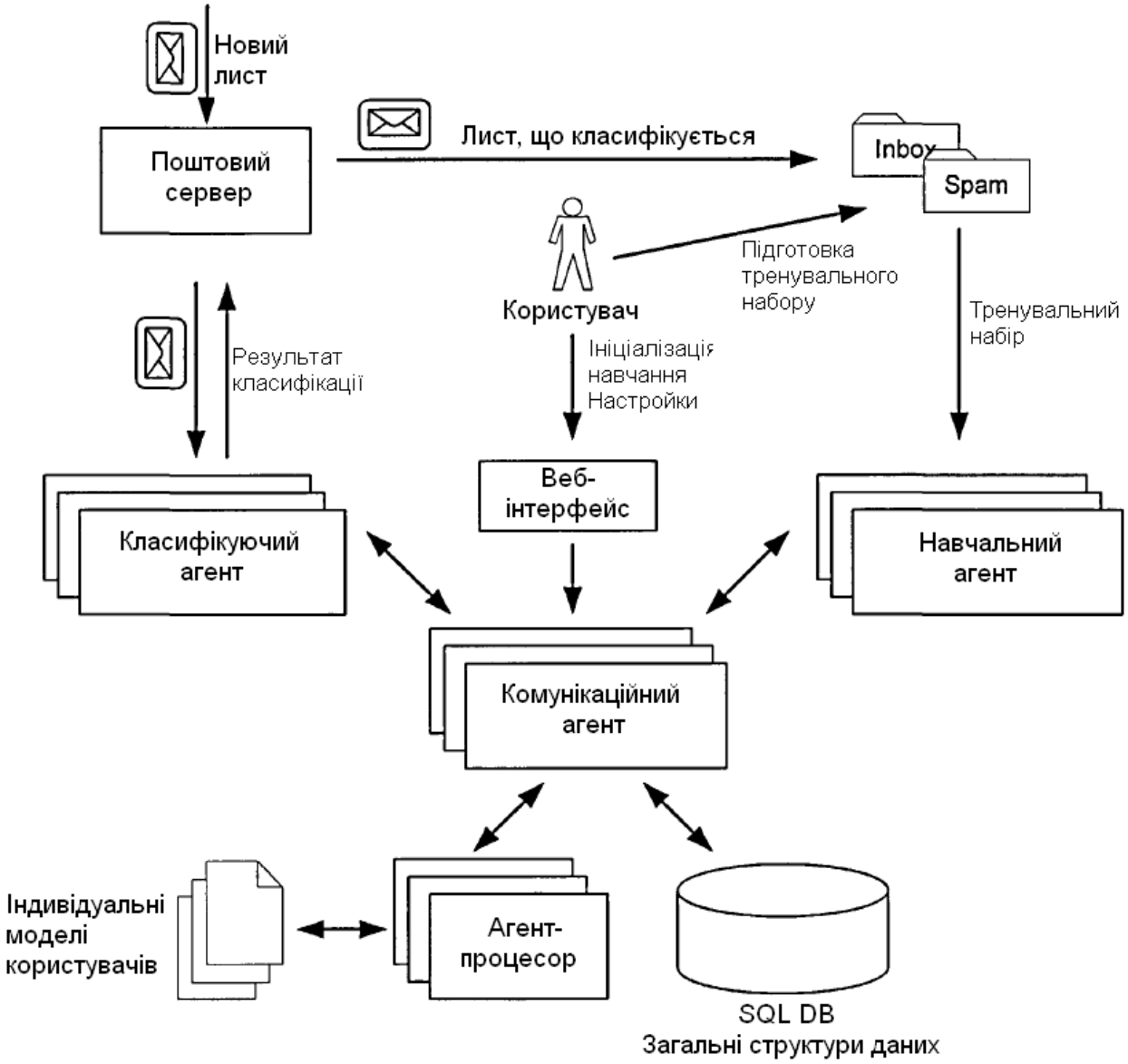

Рисунок 3.1 – Архітектура експериментальної серверної системи фільтрації електронної пошти, яка заснована на навчальних методах класифікації

У даній архітектурі комунікаційний агент не є «вузьким місцем» продуктивності, не дивлячись на те, що він є центральним вузлом системи. Його основні функції – приймати зовнішні запити на обробку даних і передавати їх на виконання відповідному агентові.

Крім незначних операцій, таких як зміна налаштувань користувачів і глобальних налаштувань сервера, основні ресурсоємні операції, що обробляються комунікаційним агентом – це заявки користувачів на побудову моделі класифікації пошти, яка передається навчальному агентові і запити класифікуючого агента на класифікацію нової пошти. Таким чином

комунікаційний агент здійснює баланс і перерозподіл навантаження для комунікаційного і повчального агентів.

Як було визначено раніше, основним показником продуктивності для системи фільтрації пошти є кількість нових повідомлень, що класифікуються, в одиницю часу. Швидкість навчання не так важлива, оскільки це достатньо рідкісна для кожного користувача подія, його можна зробити відкладеним і потім здійснювати в періоди найменшого навантаження системи. Таким чином, на рівні комунікаційного агента основним показником продуктивності є кількість оброблених запитів від класифікуючого агента. Оскільки роль комунікаційного агента в запропонованій архітектурі виконує web-сервер, для забезпечення необхідних характеристик продуктивності системи використовуються такі властивості веб-сервера як високий ступінь паралелізму, масштабованість, підтримка протоколів безпеки. Взаємодія між агентами здійснюється за допомогою захищеного протоколу обміну даних.

Агент-процесор є окремим додатком, в якому реалізовані алгоритми, пов'язані безпосередньо з побудовою моделі класифікації користувача і класифікації нових повідомлень.

Електронні листи проходять попередню обробку на рівні навчального і класифікуючого агентів, де будується їх векторне представлення. Таким чином, агент-процесор вже працює з листами у вигляді наборів векторів. Всі алгоритми попередньої обробки векторів, побудови моделі і класифікації реалізуються безпосередньо в агентові-процесорі.

Зокрема це:

- алгоритми скорочення розмірності простору ознак;
- алгоритми скорочення розміру тренувального набору;
- алгоритм скорочення шуму в тренувальному наборі;
- алгоритм побудови моделі класифікації на основі нейронних мереж;
- алгоритм класифікації нових векторів на основі побудованої моделі класифікації.

Агент-процесор – це окремий виконуваний модуль, який реалізований на мові C++. Має два режими роботи: побудова моделі по набору векторів і

класифікація нового вектора на підставі існуючої моделі. Агент має інтерфейс командного рядку. У разі побудови моделі йому передається тільки файл, в якому міститься навчальний набір, представлений у вигляді множини векторів. Результатом роботи є файл, в якому зберігається побудована модель. У разі класифікації агентові передається файл, в якому міститься новий вектор, результат класифікації якого зберігається в окремому файлі.

Теоретично здається логічним розбити агент-процесор на дві частини (навчання і класифікація) і помістити реалізацію кожної частини у відповідному агентові системи (навчальному і класифікуючому). Проте, було декілька причин, що послужили виділенню основної алгоритмічної частини системи в окремий агент.

Перш за все, таким чином реалізується концепція максимальної гнучкості конфігурації системи. Навчальний і класифікуючий агенти є компонентами системи, що настроюються. Класифікуючий агент залежить від поштового сервера, з яким він працює. Навчальний агент залежить від способу збору тренувального набору листів. У такій конфігурації є можливість достатньо невеликими зусиллями переписати навчальний і класифікуючий агенти, настроївши їх для роботи, наприклад з новим поштовим сервером. При цьому ядро системи, основні алгоритми, залишаться без зміни.

Агент-процесор виконує найбільш ресурсоємну частину операцій, як під час класифікації нових повідомлень, так і тим більше на етапі навчання. Таким чином, необхідно забезпечити його максимально можливу продуктивність. З цієї причини він написаний на C++, без використання додаткових бібліотек. Додатково, агент-процесор легко переноситься, він не залежить від операційної системи.

Навчальний агент – компонент системи, який завантажує листи користувача в систему, виконує їх попередній аналіз і розбір, і передає їх спеціальне представлення агентові-процесору.

Ініціалізація навчального агента і початок побудови моделі може здійснюватися двома способами. З боку користувача обидва варіанти абсолютно прозорі і однакові – він ініціює процес навчання за допомогою веб-інтерфейсу

системи. Далі, залежно від налаштувань системи, або безпосередньо запускається навчальний агент, або заявка на навчання потрапляє в чергу заявок. У другому випадку навчальний агент запускається планувальником заявок, який враховує завантаженість системи і розмір черги.

Мета навчального агента – отримати одним із способів, заздалегідь класифікований користувачем набір його листів, і представивши їх в певному форматі, передати агентові-процесору для побудови моделі. Формат даних для агента-процесора визначається вибраним алгоритмом. Основне завдання для навчального агента – отримати набір класифікованих листів користувача.

У експериментальній системі для вирішення цього завдання був запропонований і реалізований підхід, при якому навчальний агент завантажує листи користувача автоматично з відповідної теки на поштовому сервері про протоколу IMAP. Перш за все, такий підхід застосовний тільки для користувачів, які для читання пошти користуються протоколом IMAP або web-інтерфейсом. Це є деяким обмеженням, але тільки для навчального агента такого типу.

На практиці робота навчального агента виглядає таким чином (рисунок 3.2). Під час запуску йому передається дані про налаштування користувача, які включають інформацію, необхідну для того, щоб дістати доступ до поштового сервера користувача, інформацію про те, яку кількість листів використовувати для навчання, інформацію про попередні навчання користувача і т. д. Далі навчальний агент з'єднується з поштовим сервером користувача по протоколу IMAP і завантажує спочатку необхідну (вказану користувачем в його настройках) кількість останніх по даті отримання, помічених як прочитані листів з теки Inbox (тека по-замовчуванню для вхідних листів), вважаючи їх легальними, тобто нормальними листами. І потім необхідна кількість останніх по даті отримання листів з теки Spam (тека, яка створюється системою фільтрації пошти спеціально для спам-листів).

В процесі завантаження над кожним листом проводяться наступні операції:

- розбір тіла листа на текстові лексеми;
- попередній аналіз і відсікання неінформаційних лексем;
- виділення нетекстових ознак листа;

- передача набору лексем і нетекстових ознак комунікаційному агентові, який завантажує їх в базу даних і будує словник лексем.

Словник лексем є набором лексем і додатковою інформацією по кожній з них, що включає статистичні дані лексеми, яка зустрічається в листах користувачів.

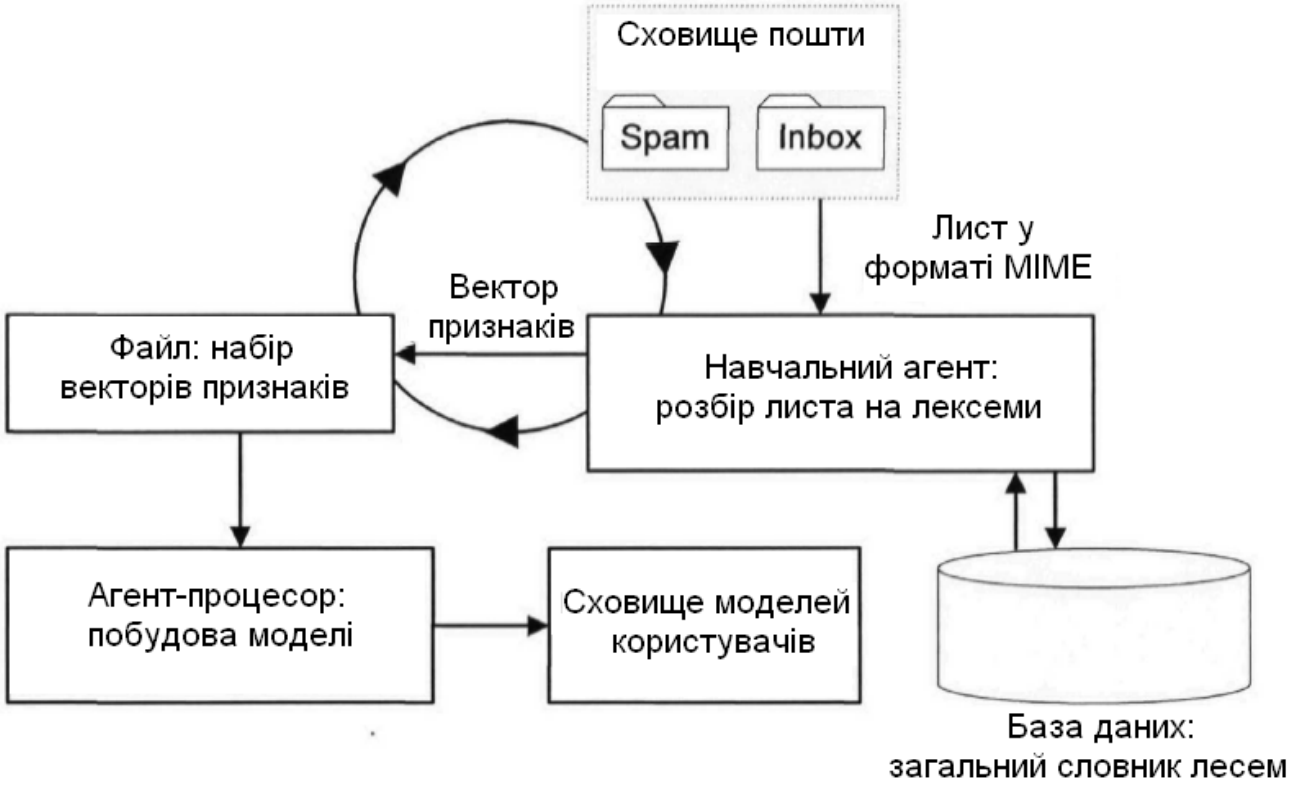

Рисунок 3.2 – Стадії навчання системи фільтрації пошти

Після завантаження всіх листів в базу даних (як легальних, так і спаму) функції навчального агента закінчуються. Далі комунікаційний агент по запиту навчального агента з наборів лексем, які відповідають листам користувача, на підставі додаткової інформації із словника, будує векторне представлення тренувального набору і зберігає його у файловій системі. Після цього тренувальний набір передається агентові-процесору, який, аналізуючи його, будує модель класифікації користувача.

Класифікуючий агент одержує нові листи від поштового сервера, використовуючи персональну модель, побудовану для конкретного користувача, класифікує їх, і повертає результат серверу.

Розглянемо детально послідовність дій на прикладі класифікуючого агента

для роботи з агентом локальної доставки Procmail (рисунок 3.3).

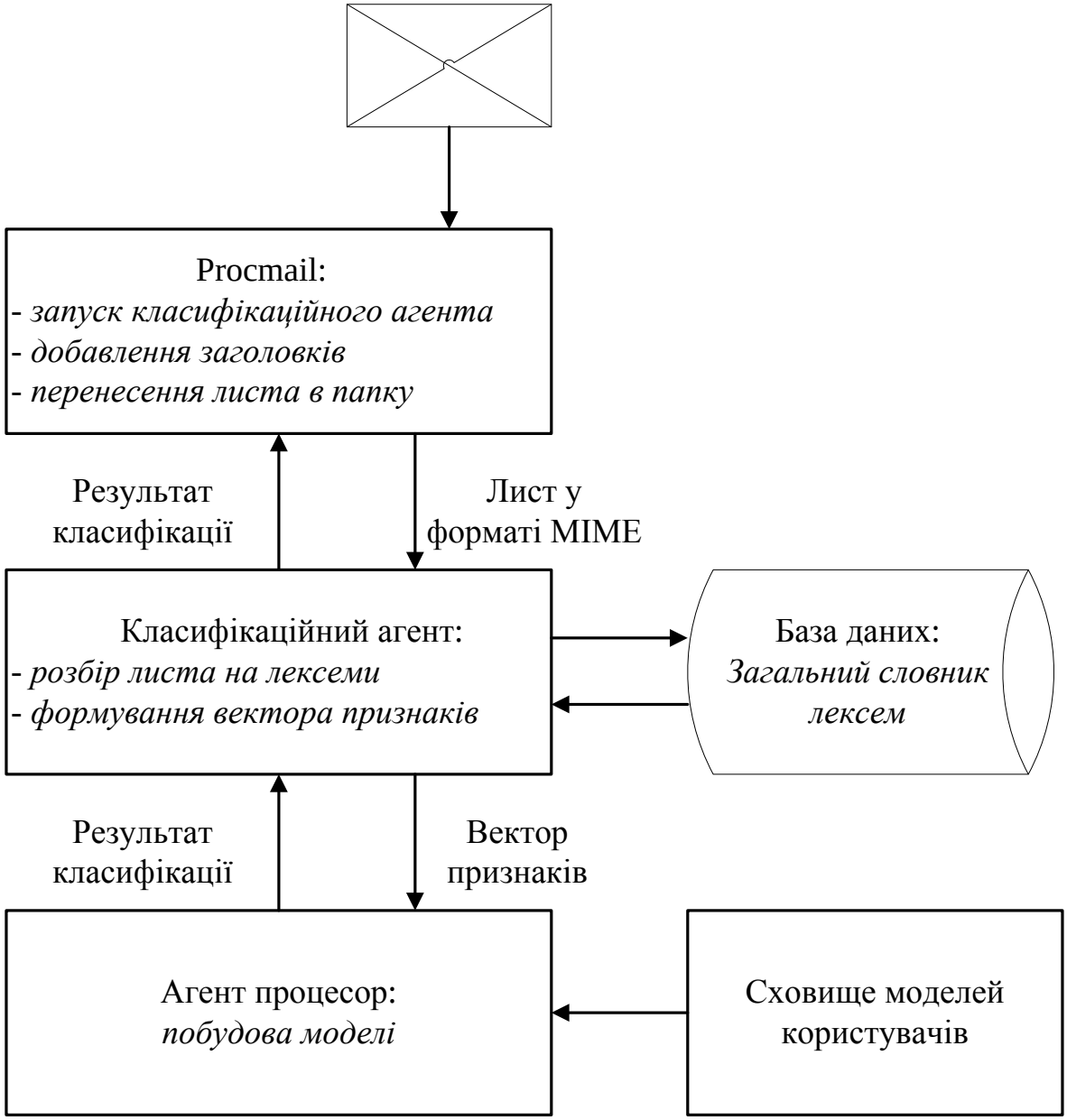

Рисунок 3.3 – Стадії класифікації системи фільтрації пошти

В якості механізму для передачі листів від поштового сервера до класифікуючого агента і виконання реакції на результат класифікації використовується агент локальної доставки Procmail. Для цього був написаний відповідний скрипт, який виконує обробку кожного листа, що приходить, таким чином:

- запускає виконуваний модуль класифікуючого агента, передаючи йому лист для обробки;
- аналізує результат і виконує необхідну дію над листом.

Класифікуючий агент при одержанні нового листа виконує наступний набір дій:

- розбір тіла листа на текстові лексеми;
- попередній аналіз і відсікання неінформаційних лексем;
- виділення нетекстових ознак листа;
- передача набору лексем і нетекстових ознак комунікаційному агентові, який у свою чергу:
  - формує векторне представлення листа на основі цього набору і словника лексем з бази даних;
  - класифікує побудований вектор за допомогою агента-процесора;
  - повертає результат класифікуючому агентові;
- повернення результату класифікації викликаючій стороні (в даному випадку в скрипт агента локальної доставки procmail).

Що стосується реакції на результат класифікації, тут можна розглядати декілька варіантів. Якщо лист класифікований як легальний, він повинен бути доставлений користувачеві. Якщо лист класифікований як спам, теоретично можливі наступні дії:

- додавання відповідної текстової мітки в лист і/або його заголовок;
- видалення листа;
- пересилка листа за заздалегідь певною адресою;
- переміщення листа в заздалегідь певну теку на поштовому сервері.

Обробка листа, який класифікований як спам може визначатися специфікою навчального агента.

Відмітимо ще раз, що в цілому на основі розробленої архітектури можлива побудова різних систем фільтрації спаму, які можуть відрізнятися агентами. Відповідно, навчальний агент залежить від джерела одержання заздалегідь класифікованих листів, необхідних для навчання, а класифікуючий агент залежить від механізму, за допомогою якого даний поштовий сервер підтримує зовнішню фільтрацію пошти.

У найбільш поширеній конфігурації системи набори заздалегідь класифікованих листів знаходяться у відповідних теках на поштовому сервері, а

навчальний агент дістає доступ до них через IMAP протокол. Такий механізм навчання визначає і дії з новою поштою. Новий лист переміщається у відповідну теку на поштовому сервері залежно від результату класифікації.

Таким чином, для розширення функціональності такої системи фільтрації спаму за рахунок підключення нових поштових серверів або додавання підтримки специфічних поштових клієнтів досить реалізувати новий класифікуючий або навчальний агент. Це дозволяє створювати єдину добре масштабовану систему фільтрації спаму, яка об'єднує різнорідні клієнти і поштові сервера, такі, що працюють в рамках однієї організації.

У базі даних зберігаються персональні настройки користувачів, словник лексем і їх характеристики, тимчасові дані, необхідні під час побудови моделі.

3.2 Користувацький інтерфейс

Користувачеві системи надається web-інтерфейс, за допомогою якого він може реєструватися в системі, змінювати налаштування і параметри фільтрації пошти, ініціалізувати процес побудови моделі, за допомогою якої надалі будуть класифікуватися листи, проводити донавчання моделі на нових прикладах листів, стежити за статистикою навчання.

3.2.1 Настройки навчання

Користувач має можливість міняти параметри, що впливають на побудову моделі.

1. Використання наперед визначеної моделі. У випадку, якщо у користувача немає накопичених листів із спамом, які він міг би використовувати в якості тренувального набору, він може скористатися даною опцією. При цьому під час навчання у користувача з теки Inbox завантажуються листи, які вважаються легальними, а листи із спамом беруться із спеціальної загальної колекції. У загальному випадку, при використанні цієї опції, якість класифікації повинна бути нижчою, ніж якби користувач підготував тренувальний набір з власних листів. Проте, використання даної опції дозволяє провести навчання і почати фільтрацію

пошти для користувача навіть у разі, коли у нього немає власних прикладів листів із спамом. Після того, як фільтр почне функціонувати і якась кількість пошти буде класифікована, користувач може донавчати фільтр, тобто побудувати нову персональну модель класифікації, яка вже буде заснована на особистих прикладах листів із спамом. Це підвищить точність моделі і якість класифікації пошти.

2. Обмеження на кількість повідомлень, що використовуються для навчання. Ця опція в настройках дозволяє користувачеві встановлювати обмеження на кількість листів кожного класу, які будуть використані для навчання. У системних настройках встановлено обмеження на завантаження і навчання не більше ніж по 2000 повідомлень кожного типу. З одного боку, це обмежує навантаження на сервер, з іншого – на практичних експериментах було визначено, що ця кількості листів достатня для побудови якісної моделі класифікації і подальше збільшення тренувального набору мало впливає на якість класифікації. У реальності, фільтр починає адекватно класифікувати пошту при одиничних прикладах в тренувальному наборі, прийнятні результати класифікації досягаються при 100-200 прикладах, а 500-1000 прикладів можна вважати достатнім розміром для первинного навчання. Тому по замовчуванню в настройках навчання встановлені обмеження по використанню не більше ніж по 500 листів кожного типу.

3. Виключення випадкового спаму. До початку навчання користувач готує тренувальний набір, вручну розкладаючи на поштовому сервері легальну пошту і спам по двох теках відповідно Inbox і Spam. При цьому помилково деяка кількість листів із спамом може залишитися в теці Inbox. Такі помилки погано впливають на якість побудованої моделі класифікації і зрештою на якість фільтрації пошти.

3.2.2 Настройки класифікації

При класифікації пошти алгоритм класифікації для кожного листа у відповідності з персональною моделлю користувача виставляє оцінку, що характеризує ступінь його приналежності до категорії спаму або легальної пошти. Кожному листу приписується числова характеристика - MessageRate. Теоретично її значення лежить від мінус нескінченності до плюс нескінченності. У реальності

воно рідко виходить за відрізок [-1, 1]. По замовчуванню межею між спамом і легальною поштою є нуль. Негативне значення означає приналежність листа до спаму, позитивне – до легальної пошти. Чим більше по модулю значення цього параметра, тим з більшою ймовірністю лист відноситься до відповідної категорії. Таким чином, зона біля нулю – це зона невизначеності.

Користувач має можливість змінити межу за умовчанням, пересунувши її в обмежених межах. Пересуваючи межу в область негативних значень, режим фільтрації лагіднішою – зменшується ймовірність помилково-позитивних помилок (класифікації легальної пошти як спаму), але при цьому погіршується якість виявлення спаму. Пересуваючи межу в область позитивних значень, фільтрація стає агресивнішою – підвищується рівень виявлення, але при цьому і збільшується помилково-позитивних помилок. Таким чином, в настройках класифікації користувач може міняти режим фільтрації залежно від особистих переваг.

3.2.3 Навчання і донавчання

З боку користувача процес роботи з системою фільтрації виглядає таким чином. Для того, щоб почати працювати з системою, користувачеві необхідно активувати фільтрацію для своєї поштової скриньки. Для цього досить авторизуватися на web-сайті системи, використовуючи той же обліковий запис і пароль, які використовуються на поштовому сервері.

Після цього необхідно підготувати тренувальний набір для навчання системи. Для цього потрібно переконатися, що на поштовому сервері в теці Inbox відсутня нелегальна пошта, а в створену теку Spam перекласти максимальну кількість наявних прикладів листів із спамом.

Потім за допомогою web-інтерфейсу ініціалізується процес навчання (побудови моделі). Статус навчання (кількість оброблених листів за час, що минув) відображаються у відповідному вікні web-інтерфейсу системи. Час навчання залежить від розміру тренувального набору, продуктивності сервера і, як правило, складає декілька хвилин. Після завершення навчання система повністю готова до фільтрації пошти і тренувальний набір (зокрема нелегальну

пошту з теки Spam) можна видалити. Нова пошта, що приходить до користувача, класифікуватиметься і відповідно до персональних настройок розкладатиметься в теки Spam і Inbox.

Періодично, у разі погіршення фільтрації пошти, користувач може виконати донавчання системи. Для цього необхідно перемістити невірно класифіковані листи у відповідну для них теку, зайти на web-сайт системи і виконати донавчання. Під час цього процесу використовуються тільки нові (що раніше не брали участь в навчанні) приклади тренувального набору, тому донавчання проходить набагато швидше за первинне навчання системи. У реальній експлуатації, як правило, донавчання не має сенсу проводити частіше, ніж один-два разу на місяць.

3.2.4 «Чорні»/«білі» списки адрес відправників

Користувач має можливість сформувати свій персональний «чорний»/«білий» список e-mail адрес відправників пошти. У реальності, практичну користь має тільки «білий» список адрес. Листи від відправників, які присутні в цьому списку, вважаються легальними незалежно від результату класифікації, і відповідно завжди потрапляють в теку Inbox. Ця функція дозволяє, наприклад, виключити можливість помилково-позитивних помилок для листів найбільш важливих адресатів.

Управління списками адрес здійснюється за допомогою web-інтерфейсу. Є можливість додавати і видаляти одиночні адреси, текстові файли із списками адрес, а також адресні книги у форматі Windows Address Book (wab).

Існує така проблема, що класифікація з використанням «чорних»/«білих» списків може впливати на коректність тренувального набору. Наприклад, користувач може включити в «білий» список адресу якої-небудь комерційної розсилки, листи якої дуже схожі за змістом на рекламні оголошення і в нормальній ситуації були б віднесені до спаму. Таким чином, такі листи знаходитимуться в теці Inbox разом з легальною поштою і потім використовуватимуться як приклади легальної пошти при навчанні. Оскільки такі листи за змістом відносяться швидше до спаму, це негативно впливатиме на

якість побудованої моделі і, отже, на якість класифікації нової пошти.

Для коректної побудови моделі класифікації в системі фільтрації створена опція, за допомогою якої користувач може впливати на стратегію формування тренувального набору. При класифікації кожного листа в його заголовок крім оцінки, проставленої фільтром, додається ознака, чи знаходиться автор листа в одному із списків користувача. Під час навчання ті листи, які були класифіковані за допомогою «чорних»/«білих» списків, виключаються з тренувального набору.

3.2.5 Статистика навчання

У системі зберігається і доступна для перегляду історія всіх навчань і донавчань користувача і їх характеристики, такі як кількість листів в тренувальному наборі, тимчасові характеристики навчання, значення призначених для користувача параметрів, які використовувалися при навчанні. Також під час навчання відображається поточний статус операції і динаміка процесу.

3.3 Програмна реалізація модулів

В даний час серверна система фільтрації електронної пошти інтегрована і випробувана з наступними поштовими серверами: Sendmail 8.12, Exim 4.34, CommuniGate Pro 4.2, Microsoft Exchange 2000.

Для функціонування системи додатково використовується наступне програмне забезпечення: СУБД mySQL 3.23, Web-сервер Apache 2.0.40.

3.3.1 Концепція інтеграції системи класифікації з поштовими системами

Запропонована концепція архітектури системи класифікації пошти (рисунок 3.4), в основі якої лежить багатоагентний підхід, забезпечує вирішення двох задач. По-перше, це досягнення високого ступеня паралелізму і масштабування, і, по-друге, гнучкість при інтеграції з різними поштовими серверами при класифікації і джерелами даних при навчанні. Для того, щоб мінімізувати зміни у всій системі фільтрації, специфічні для конкретного

поштового сервера інтерфейси, які використовуються при класифікації і специфічні інтерфейси джерела даних винесені в два відповідних агенти: класифікуючий і навчальний.

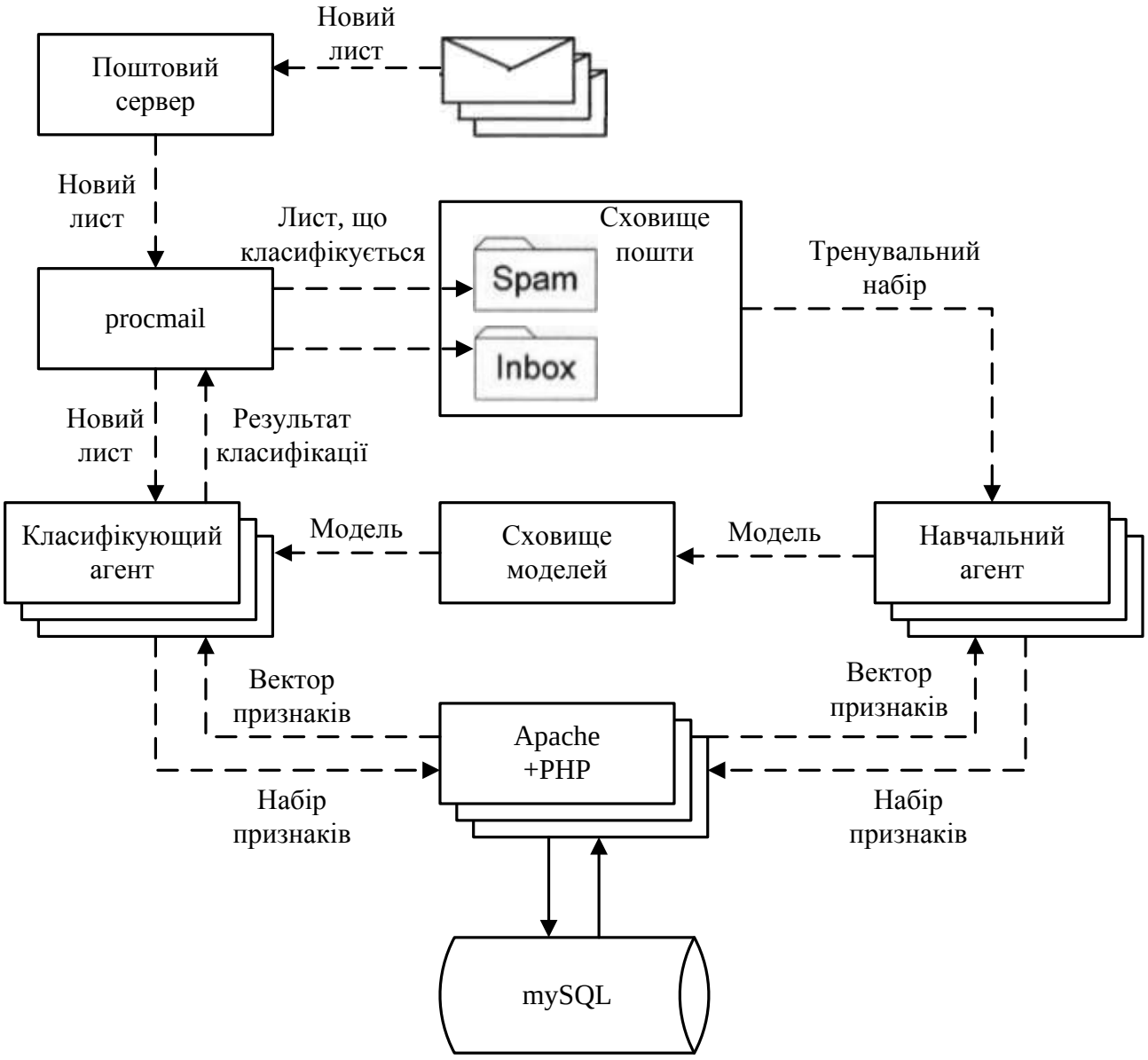

Рисунок 3.4 – Архітектура експериментальної системи

Навчальний агент може бути реалізований для різних сховищ листів, наприклад, на основі протоколу IMAP на централізованому сервері або як персональний агент, що використовує для навчання листи з персональних тек користувача, які передані по POP3 і зберігаються на робочій станції користувача. У експериментальній системі для вирішення цієї задачі був використаний і реалізований підхід, при якому навчальний агент завантажує листи користувача автоматично з відповідної теки на поштовому сервері про протоколу IMAP. Такий

підхід застосовний тільки для користувачів, які для читання пошти користуються протоколом IMAP або web-інтерфейсом. Це є деяким обмеженням, але тільки для навчального агента такого типу.

При інтеграції з новим поштовим сервером всі зміни будуть інкапсульовані всередині класифікуючого агента. Він служить інтерфейсом між системою фільтрації пошти і поштовим сервером, оскільки специфіка механізму передачі листів для класифікації і подальші дії залежно від результатів класифікації враховуються тільки на рівні класифікуючого агента. Практично будь-який поштовий сервер має певний інтерфейс для забезпечення перевірки листів стороннім фільтром. Це інтерфейс зокрема можна використовувати і для фільтрації спаму.

Для експериментальної системи класифікації пошти було розроблено три типи різних класифікуючих агентів для наступних поштових серверів:

- Microsoft Exchange 2000/2003;
- Communigate Pro;
- Sendmail, Exim (а також будь-який поштовий сервер, що підтримує роботу з агентом локальної доставки Procmail).

3.3.2 Приклади інтеграції з поштовими системами

Інтеграція з Sendmail і Exim.

Основна версія системи була випробувана на обчислювальній системі наступній конфігурації: операційна система RedHat Linux 9.0, поштовий сервер Sendmail 8.2 або Exim 4.34, агент локальної доставки Procmail 3.22.

Класифікуючі і навчальні агенти – окремі незалежні компоненти, які взаємодіють з ядром системи за допомогою протоколу HTTP з використанням SSL.

Навчальний агент, використовуючи протокол IMAP (IMAPs), завантажує необхідні для навчання листи з тек користувача на поштовому сервері Inbox і Spam, використовуючи загальний словник термів з бази даних, створює їх векторне представлення, створює на основі листів тренувальний набір і потім передає його в агент-процесор, який будує персональну модель користувача.

Класифікуючий агент отримує поштові повідомлення користувачів системи через агента локальної доставки пошти procmail. Далі, залежно від настройок користувача і результатів класифікації, procmail доставляє лист у відповідну теку користувача (Spam або Inbox). Додатково в заголовок кожного листа додається інформація про його класифікацію і числова характеристика класифікації.

Інтеграція з CommuniGate Pro.

CommuniGate Pro [23] – поширений поштовий сервер. Має базові вбудовані засоби для фільтрації пошти, такі як правила, що задаються вручну. Підтримує зовнішні контекстні фільтри, які використовуються для перевірки на віруси і зокрема, – фільтрації спаму. Поштовий сервер CommuniGate Pro підтримує паралельну обробку повідомлень при фільтрації. Зовнішній фільтр повинен підтримувати певні інтерфейси для взаємодії з поштовим сервером. Таким чином, схема інтеграції включає тільки зміну класифікуючого агента. Навчальний агент залишається незмінним і отримує дані для навчання через протокол IMAP. У класифікуючому агентові було необхідно змінити вхідний інтерфейс, за допомогою якого відбувається передача листа для класифікації і вихідний інтерфейс, для повідомлення сервера про результат класифікації. У web-інтерфейсі настройок поштового сервера вказується в якості зовнішнього фільтра класифікуючий агент і вказуються дії, які необхідно здійснити залежно від результату класифікації. В даному випадку – це перенесення листа у відповідну теку користувача – Inbox або Spam.

Інтеграція з Microsoft Exchange 200012003.

В даному випадку задача інтеграції здійснювалося по тій же схемі що і інтеграція з CommuniGate Pro. Навчальний агент залишений без змін – для доступу до пошти користувача і формування тренувального набору використовується інтерфейс протоколу IMAP. Класифікуючий агент був змінений відповідно до інтерфейсів Microsoft Exchange Server 2000/2003.

MS Exchange підтримує використання зовнішніх фільтрів контекстної фільтрації пошти [24]. Схема інфраструктури фільтрації пошти представлена на рисунку 3.5.

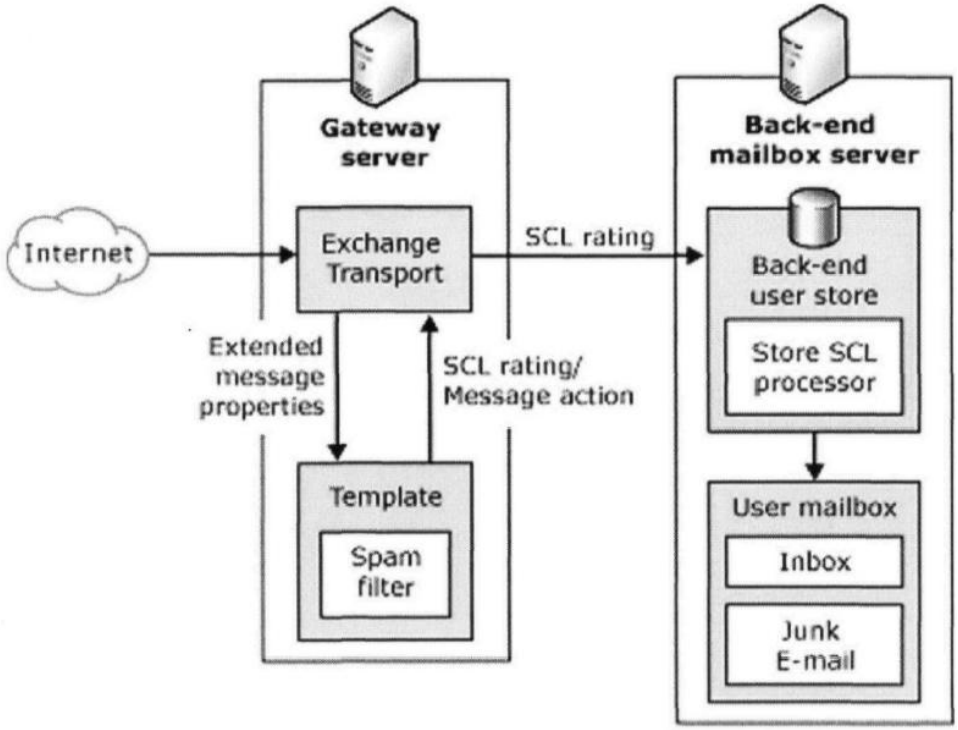

Рисунок 3.5 – Інфраструктура спам-фільтрації в сервері Microsoft Exchange

SMTP-шлюз приймає повідомлення і передає його за допомогою Exchange Transport до зареєстрованого зовнішнього фільтру. Сервер MS Exchange надає фільтру для аналізу крім стандартних властивостей повідомлення розширені властивості (Extended Message Properties). В результаті оцінки повідомлення фільтр виставляє для нього так званий рівень достовірності спаму (Spam Confidence Level, SCL). У реальності фільтр у цей момент може або, проставивши значення SCL, вирішити подальшу доставку повідомлення, або не приймати повідомлення зовсім. У даній конфігурації повідомлення приймається у будь-якому випадку і передається для доставки користувачеві. Залежно від встановленої межі значення SCL, повідомлення доставляється в теку користувача Inbox або Junk E-mail.

3.3.3 Програмні модулі

Експериментальна система фільтрації електронної пошти складається з наступних програмних модулів:

- Комунікаційний агент – web-додаток на мові програмування PHP.
- Навчальний агент – додаток на мові програмування C++. Для взаємодії з

поштовим сервером з використанням протоколу IMAP навчальний агент використовує вільно розповсюджену бібліотеку з відкритим початковим текстом c-client library [25].

- Класифікуючий агент для поштового сервера sendmail – додаток на мові програмування C++.
- Класифікуючий агент для поштового сервера CommuniGate Pro –додаток на мові програмування C++.
- Класифікуючий агент для поштового сервера MS Exchange 2000/2003 – додаток на мові програмування C++.
- Агент-процесор – додаток на мові програмування C++.
- Навчальний і класифікуючий агенти використовують загальну бібліотеку для аналізу листів і виділення текстових і статистичних ознак.
- Навчальний і класифікуючий агенти взаємодіють з комунікаційним агентом, який є веб-сервером, використовуючи протокол HTTPs. Для цього застосовується вільно розповсюджена бібліотека з відкритим початковим текстом libcURL [26].

Розроблена експериментальна система класифікації електронної пошти володіє наступними характеристиками:

1. Універсальність – незалежність від поштової системи. Система фільтрації пошти може бути інтегрована з будь-яким поштовим сервером, що підтримує використання зовнішніх фільтрів для перевірки пошти. Реалізовані агенти, що забезпечують інтеграцію з наступними поштовими серверами: CommuniGate Pro, MS Exchange, Sendmail, Exim.
2. Розподіленість. Система є набором агентів, кожен з яких виконує окрему логічну задачу. Практично всі компоненти системи можуть знаходитися на різних фізичних серверах. Це забезпечує необхідний рівень паралелізму і продуктивності.
3. Безпека. Не знижує існуючий рівень безпеки користувачів. Використання персональної інформації користувачів локалізоване і захищене. Окремі компоненти системи для безпечного обміну даними використовують сучасні протоколи шифрування даних.

4. Персоналізація. Кожен користувач має свою персональну модель, відповідно до якої здійснюється фільтрація пошти. Модель будується на власних листах користувача, і, таким чином відображає його особисті переваги. Додатково є можливість використовувати для навчання загальний, відомий набір листів.

Розроблена експериментальна система класифікації електронної пошти впроваджена в експлуатацію на підприємстві «Д.М.В.», що підтверджує довідка про впровадження (додаток А).

В третьому розділі дипломної роботи проведена реалізація серверної системи класифікації електронної пошти, зокрема розроблена архітектури системи, користувацький інтерфейс, який дозволяє редагувати настройки навчання, настройки класифікації, проводити навчання і до навчання системи, формувати «чорні»/«білі» списки адрес відправників, зберігати статистику навчання. Також здійснена програмна реалізація модулів системи та інтеграції системи класифікації з поштовими системами.

ВИСНОВКИ

Основні результати дипломної роботи:

1. Обгрунтувано основні концепції серверної системи класифікації пошти, яка заснована на навчальних методах класифікації і використовує персональну модель.

2. Представлений аналіз навчальних методів класифікації і їх характеристик. Обгрунтовується вибір базового алгоритму класифікації, формулюються вимоги по його оптимізації і представленню структур даних.

3. Проведений вибір архітектури системи, яка забезпечує застосування навчального алгоритму класифікації за допомогою ефективного розподілу навантаження і забезпечує масштабованість системи і здатність працювати в гетерогенному середовищі.

4. Запропонований оригінальний підхід до вирішення задачі фільтрації електронної пошти на рівні поштового сервера, який заснований на використанні персоніфікованої моделі класифікації.

5. Розроблені алгоритми, які засновані на розвитку методів штучного інтелекту і призначені для вирішення завдання класифікації електронної пошти на рівні поштового сервера.

6. На основі запропонованих алгоритмів розроблена і апробована експериментальна багатоагентна серверна система фільтрації електронних повідомлень, що використовує персоніфіковану модель класифікації.

7. За результатами дослідної експлуатації на підприємстві і проведених порівняльних експериментів показано перевагу системи класифікації електронної пошти в порівнянні з найбільш поширеними в даний час алгоритмами і системами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Email Systems, Email Systems Ltd [Електронний ресурс]. – Режим доступу: http://www.emailsystems.com.
2. Aas K., Eikvil L. Text categorisation: A survey. // Technical report, Norwegian Computing Cente. – 1999. – P. 124.
3. Sebastiani F. Machine Learning in Automated Text Categorization, ACM Computing Surveys. – 2002. – Vol. 34, №. 1. – P. 1-47.
4. Petrovskiy M. An Approach to Membership Model Identification for Fuzzy Support Vector Machines // Proceedings of International Conference on Recent Advances in Soft Computing. – Nottingham (United Kingdom). – 2004. – P. 45-50.
5. O. de Vel, Anderson A., Corney M., Mining email content for author identification forensics. // SIGMOD Record. – 2001. – № 30(4). – P. 55-64.
6. Dumais S., Piatt J., Heckerman D., Sahami M. Inductive learning algorithms and representations for text categorization // Proceedings of 7th ACM International Conference on Information and Knowledge Management. – Bethesda (MD). – 1998. – P. 148-155.
7. Drucker H., Wu D., Vapnik V. Support Vector Machines for Spam Categorization // IEEE Trans on Neural Networb. – 1999. – Vol 10, № 5. – P. 1048-1054.
8. Vapnik V. The nature of statistical learning theory // Springer Verlag. – New York. – 1995. – P. 67.
9. Vapnik, V. Statistical learning theory // Wiley. – New York. – 1998. – P. 91.
10. Бари Н. К. Тригонометрические ряды. М.: Гос. издательство физико-математической литературы, 1961. – 933 с.
11. Joachims T. Text Categorization with Support Vector Machines: Learning with Many Relevant Features // Proceedings of 10th European Conference on Machine Learning // Springer Verlag. – Heidelberg (DE) – 1998. – P.137-142.
12. Bishop Christopher M. Neural Networks for Pattern Recognition // Oxford University Press. – 1995. – P. 135.

13. Gurney K. An Introduction to Neural Networks // UCL Press. – 1997. – P. 342.
14. Salton G., McGill J. An introduction to modern information retrieval. – New York: McGraw-Hill. – 1983. – P.138.
15. Apache Software Foundation. The Apache SpamAssassin Public Corpus [Електронний ресурс]. – Режим доступу: http://spamassassin.apache.org/publiccorpus.
16. Yang Y., Pedersen J. O. Feature selection in statistical learning of text categorization // Proc. of the 14th International Conference of Machine Learning. – 1997. – P.412-420.
17. Yang Y., Pedersen J. O. A comparative study of feature selection in text categorization // Proceedings of the Fourteenth International Conference on Machine Learning – Morgan Kaufman Publishers. – 1997. P.412-420.
18. Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Fumas, Richard A. Harshman. Indexing by Latent Semantic Analysis // Journal of the American Society of Information Science. – 1990.– Vol. 41, №. 6, P. 391-407.
19. Wall Michael E., Andreas Rechtsteiner, Luis M. Rocha. Singular value decomposition and principal component analysis. A Practical Approach to Microarray Data Analysis. D.P. Berrar, W. Dubitzky, M. Granzow, eds. pp. 91-109, Kluwer: Norwell, MA (2003). LANLLA-UR-02-4001.
20. Головко В.А. Нейроинтеллект: теория и применение. Книга 1: Организация и обучение нейронных сетей с прямыми и обратными связями. – Брест: Изд. БПИ, 1999. – 262 с.
21. Головко В.А. Нейроинтеллект: теория и применение. Книга 2: Самоорганизация, отказоустойчивость и применение нейронных сетей. – Брест: Изд. БПИ, 1999. – 228 с.
22. Горбань А.Н., Россиев Д.А.. Нейронные сети на персональном компьютере. – Новосибирск: Наука, сибирская узд. фирма РАН, 1996. – 276 с.
23. Stalker Software Inc. CommuniGate Pro Core Server [Електронний ресурс]. – Режим доступу: http://www.stalker.com/content/groupware.htm.
24. Microsoft Exchange Server 2003, SDK Documentation, Anti-Spam Infrastructure. MSDN Library [Електронний ресурс]. – Режим доступу:

http://msdn.microsoft.com/library/default.asp?url=/librarv/enus/e2k3/e2k3/ast anti spam infrastructure.asp.

25. University of Washington, IMAP Information Center, C-client library [Електронний ресурс]. – Режим доступу: http://www.washington.edu/imap.
26. Stenberg, D. UbcURL - the multiprotocol file transfer library [Електронний ресурс]. – Режим доступу: http://curl.haxx.se/libcurl.
27. Методичні рекомендації до виконання дипломної роботи з освітньо-кваліфікаційного рівня “Магістр”. Спеціальність „Комп’ютерні системи та мережі” / О.М. Березький, Р.Б. Трембач, Г.М. Мельник / Під ред. О.М. Березького – Тернопіль: ТНЕУ, 2012.– 42 с.

Додаток А

Довідка про використання