

Міністерство освіти і науки України  
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЇВО-МОГИЛЯНСЬКА  
АКАДЕМІЯ»

Кафедра інформатики

## **Програмне забезпечення для виявлення спаму на основі машинного навчання**

**Текстова частина до курсової роботи  
за спеціальністю «Програмна інженерія»**

Керівник курсової роботи:  
доцент, кандидат технічних наук  
Ковалюк Т.В.

\_\_\_\_\_  
(підпис)

“ \_\_\_\_ ” \_\_\_\_\_ 2020 р.

Виконав студент

Денисенко Ігор Михайлович

“ \_\_\_\_ ” \_\_\_\_\_ 2020 р.

Київ 2020

**КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ КУРСОВОЇ РОБОТИ**

Тема: Програмне забезпечення для виявлення спаму на основі машинного навчання

**Календарний план виконання роботи:**

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	15.11.2019	
2.	Огляд технічних статей за темою роботи.	15.01.2020	
3.	Збір інформації	10.03.2020	
4.	Розгляд і аналіз існуючих застосунків	20.03.2020	
5.	Переваги та недоліки застосунків	20.03.2020	
6.	Розробка застосунку	30.03.2020	
7.	Тестування застосунку	05.04.2020	
8.	Створення слайдів для доповіді та написання доповіді.	17.04.2020	
9.	Аналіз отриманих результатів з керівником, написання доповіді та попередній захист курсової роботи.	20.04.2020	
10.	Корегування роботи	21.04.2020	
11.	Остаточне оформлення пояснювальної роботи та слайдів.	23.04.2020	
12.	Захист курсової роботи (проекту)	25.04.2020	

Студент: Денисенко Ігор Михайлович

Керівник: Ковалюк Тетяна Володимирівна

“ \_\_\_\_ ” \_\_\_\_\_ 2020 р.

## Зміст

<b>АНОТАЦІЯ.....</b>	<b>4</b>
<b>Вступ.....</b>	<b>5</b>
<b>Розділ 1 Аналіз спаму.....</b>	<b>9</b>
<b>Розділ 1.1 Види спаму.....</b>	<b>9</b>
<b>Розділ 1.2 Машинне навчання. Задача класифікації.....</b>	<b>12</b>
<b>Розділ 1.3 Класифікація спаму.....</b>	<b>16</b>
<b>Розділ 2. Переваги та недоліки аналогів.....</b>	<b>20</b>
Розділ 2.1 Переваги та недоліки SpamTitan.....	20
Розділ 2.2 Переваги та недоліки MailWasher.....	23
<b>Розділ 3 Застосунок для виявлення спаму.....</b>	<b>26</b>
<b>Розділ 4 Тестування застосунку.....</b>	<b>31</b>
<b>Висновки.....</b>	<b>33</b>
<b>Список використаної літератури.....</b>	<b>34</b>

## АНОТАЦІЯ

*Інтернет відгуки відіграють все більш важливу роль у прийнятті рішень щодо отримання потенційних клієнтів. Між іншим, керуючись бажанням отримати прибуток чи велику кількість трафіку, можна використати спам-бота, для написання неправдивих відгуків або піднімати чи знижувати репутацію товарів чи послуг. Відповідно, застосунки для виявлення спаму є корисними як і для бізнесу, так і для фізичних осіб. Однак, на відміну від інших завдань, таких як класифікація новин або класифікація веб-ресурсів, існуючі бази даних спаму дуже обмежені через високу вартість роботи людей, що збирають та власноруч класифікують спам, проте ефективність виявлення спаму вручну набагато більша ніж в автоматичному режимі, навіть у порівнянні з висококласними класифікаторами. У цій роботі я займаюсь створенням застосунку для виявлення спаму на основі нейронної мережі, спочатку буде створена модель нейронної мережі, далі набір даних для навчання мережі. Фінальним етапом проекту є створення додатку для можливостю людини перевірки його повідомлення на спам.*

## ВСТУП

В даний час, час високих технологій, автоматизованого розвитку широкого поширення набула електронна пошта, її використовує кожен, від простого громадянина до великих корпорацій.

Зараз багато фірм реєструють на кожного співробітника поштову скриньку, на який працівникові приходить різна інформація від фірми, також ця скринька використовується для реєстрації в різних сервісах компанії.

Крім цього майже кожна людина на Землі має особисту електронну скриньку, і як правило, не одну. Крім листів від близьких і колег скринька дуже часто засмічується спамом.

В силу надзвичайно високої прибутковості спамерського бізнесу технічні засоби боротьби зі спамом не завжди досягають своєї мети - спамери без кінця винаходять все нові і нові способи обходу фільтрів. Тому для ефективної протидії поширенню незапрошених електронних повідомлень необхідне об'єднання різних зусиль - технологічних (виробництво програмного забезпечення), політичних (прийняття законів) і громадських (роз'яснення малому бізнесу, чим шкідливий спам).

Таким чином, спам як комплексна проблема потребує комплексного вирішення, що включає наступні елементи:

1. Просвіта (освітня діяльність);
2. Організаційна діяльність;
3. Технологічні заходи;
4. Законодавство.

Для ефективної боротьби зі спамом потрібно не тільки взаємодія різних суб'єктів, але і активна позиція всіх учасників. Складність проблеми обумовлює відносно тривалі терміни її вирішення; проте в цілях підвищення суспільної значимості боротьби зі спамом необхідно вже в найкоротші терміни забезпечити досягнення «проміжних перемог». В рамках вирішення проблеми слід також широко використовувати міжнародний досвід, накопичений в цій області

Актуальність даної теми обумовлена тим, що сьогодні кожна людина, що має електронну скриньку і спілкується по електронній пошті відчуває, м'яко кажучи певні незручності, коли на його адресу листи сумнівного характеру

Таким чином, необхідно шукати шляхи вирішення цієї проблеми, яка на руку лише тіньовим менеджерам мережевого маркетингу.

Мета даної роботи - розглянути методи і можливості боротьби зі спамом, створити нейронну мережу для розпізнавання спау, та застосунок-обгортку навколо мережі.

Завдання - оцінити ефективність методів боротьби зі спамом, виявити тенденції, які намічаються в методах боротьби зі спамом.

Отже, перейдемо до самого поняття спау.

Спам – це, незаконно розповсюджена шляхом масових розсилок, інформація рекламного характеру, отримання якої не узгоджено з користувачем. Іноді можна зустріти написання СПАМ або SPAM, оскільки слово приймають за черговий відомий нам акронім. І це дійсно акронім, за

походженням є складноскороченим словом, але не аббревіатурою, тому писати його треба «в нижньому регістрі», малими літерами. Це «складне» словосполучення утворилося з усіченого spiced ham. У буквальному перекладі spiced ham - «шинка зі спеціями», але насправді в консервних банках з написом «Spam», наскільки відомо з історії, перебував ковбасний фарш. А тепер було б непогано розповісти детальніше про історичний аспект поняття «спам».

У 1937 році досить відома фірма у США Hormel Foods розробила новий різновид консервів. Компанією був оголошений конкурс на кращу назву новому продукту, в якому переміг радіоактор з Нью-Йорка Кеннет Денью, який запропонував назвати тушонку звучною словом «СПАМ». Успіх нової назви був приголомшливим! Воно було на слуху у всіх і кожного, і збіднілі після кризи американці змітали дешеві консерви з полиць магазинів. Фірма пропонувала сотні рецептів приготування страв з СПАМА, від простих сендвічів до вишуканих страв: тонко порізані скибочки СПАМА, политі солодкими соусами і заправлені дрібно порубаними шматочками СПАМА.

СПАМ настільки заволодів умами простих американських трудівників, що його виробництво зростало рік від року, і неможливо було пройти вздовж вітрин магазинів або відкрити будь-яку бульварну газету, щоб не наштовхнутися на рекламу СПАМА - недорогий, економний, є прикладом смачної та поживної їжі для всієї родини. Пізніше СПАМ вийшов за межі Америки і став завойовувати континенти - його продавали в Північній і Південній Америці, Європі, Азії і навіть в Японії. Таким чином слово СПАМ стало як би синонімом нав'язливої реклами.

Достеменно невідомо, чи була фірма Hormel Foods першовідкривачем в масову розсилку листів з рекламою по електронній пошті, однак в 1997 році

у неї була вже маса конкурентів, які робили подібні розсилки без жодного докору сумління. А слово «СПАМ» в Інтернеті прижилося і стало позначенням для наглої, безпардонного і нахабної непрошеної реклами.



## РОЗДІЛ 1. АНАЛІЗ СПАМУ

### Розділ 1.1 Види спаму

1. **Реклама.** Деякі компанії, що займаються легальним бізнесом, рекламують свої товари чи послуги за допомогою спаму. Привабливість такої реклами - низька вартість і (імовірно) велике охоплення потенційних клієнтів. Втім, така реклама може мати і зворотний ефект, викликаючи відторгнення у одержувачів.
2. **Реклама незаконної продукції.** За допомогою спаму рекламують продукцію, про яку не можна повідомити іншими способами - наприклад, порнографію, контрафактні товари (підробки, конфіскації), лікарські засоби з обмеженнями по обороту, незаконно отриману закриту інформацію (бази даних), контрафактне програмне забезпечення. Сюди ж відноситься і реклама самих послуг розсилки спаму.
3. **Антиреклама.** Заборонена законодавством про рекламу інформація - наприклад, ганьбити конкурентів і їхню продукцію, - також може поширюватися за допомогою спаму.
4. **«Нігерійські листи».** Іноді спам використовується шахраями, щоб виманити гроші в одержувача листа. Найбільш поширений спосіб отримав назву «нігерійські листи», тому що велика кількість таких листів приходило з Нігерії. Такий лист містить повідомлення про те, що одержувач листа може отримати будь-яким чином велику суму грошей, а відправник може йому в цьому допомогти. Потім відправник листа просить перевести йому трохи грошей під приводом, наприклад, оформлення документів або відкриття рахунку. Виманювання цієї суми і є метою шахраїв.

5. **Фішинг.** Він являє собою спробу спамерів виманити в одержувача листа номера його кредитних карток або паролі доступу до систем онлайн-платежів. Такий лист зазвичай маскується під офіційне повідомлення від адміністрації банку. У ньому говориться, що одержувач повинен підтвердити відомості про себе, інакше його рахунок буде заблокований, і наводиться адреса сайту (належить спамерам) з формою, яку треба заповнити. Серед даних, які потрібно повідомити, присутні і ті, які потрібні шахраям. Для того, щоб жертва не здогадалася про обман, оформлення цього сайту також імітує оформлення офіційного сайту банку.
6. Також є досить розповсюдженими DoS і DDoS-атаки, масове розсилання від імені іншої особи, для того щоб викликати до нього негативне ставлення чи масове розсилання листів, що містять комп'ютерні віруси (для їх початкового поширення).
7. Розсилка листів, що містять жалісну історію з інформацією про те, що за кожному пересилання листа якийсь інтернет-провайдер нібито виплатить родині потерпілого певну суму грошей на лікування. Метою такої розсилки є збір e-mail адрес: після численних пересилань всім знайомим в тексті такого листа часто містяться e-mail адреси всіх, кому воно було переслано раніше. А в числі чергових адресатів цілком може виявитися і ініціював її спамер.
8. Поштові черв'яки певного типу самі поширюють себе за допомогою електронної пошти. Заразивши черговий комп'ютер, такий хробак сканує комп'ютер в пошуках e-mail адрес і розсилає себе за знайденими адресами і так далі.

Також аналогічну поведінку демонструють деякі антивірусні програми і спам-фільтри. Але, така поведінка зустрічається тільки у дуже старих програм - випущених ще до того, як проблема спаму в Інтернеті стала досить розповсюдженою.

## **Розділ 1.2. Машинне навчання. Задача класифікації**

Машинне навчання - це розділ штучного інтелекту, який досліджує методи, що дозволяють комп'ютерам покращувати свої характеристики на основі отриманого досвіду.

Завдяки машинному навчанню програміст не зобов'язаний писати інструкції, що враховують всі можливі проблеми і містять всі варіанти рішення проблем. Замість цього в програму закладають алгоритм самостійного знаходження рішень шляхом комплексного використання статистичних даних, з яких виводяться закономірності і на основі яких робляться прогнози.

Технологія машинного навчання на основі аналізу даних бере початок ще в минулому столітті, коли почали розробляти перші програми для гри в шашки. За минулі десятиліття загальний принцип не змінився. Зате завдяки вибухового зростання обчислювальних потужностей комп'ютерів, багаторазово ускладнилися закономірності і прогнози, створювані ними, і розширилося коло проблем і завдань, що вирішуються з використанням машинного навчання.

Щоб запустити процес машинного навчання, для початку необхідно завантажити в комп'ютер датасет, тобто певний набір даних, на яких алгоритм буде вчитися обробляти запити. Наприклад, можуть бути фотографії собак і котів, на яких вже є мітки, що позначають до кого вони відносяться. Після процесу навчання, програма вже сама зможе розпізнавати собак і котів на нових зображеннях без вмісту міток. Процес навчання триває і після виданих прогнозів, чим більше даних ми проаналізували програмою, тим більше точно вона розпізнає потрібні зображення.

Завдяки машинному навчанню комп'ютери вчаться розпізнавати на фотографіях і малюнках не тільки різних людей, але і пейзажі, предмети, текст і цифри. Що стосується тексту, то і тут не обійтися без машинного навчання: функція перевірки граматики зараз присутня в будь-якому текстовому редакторі і навіть тут в Microsoft Word, де зараз пишу цей текст. Причому враховується не тільки написання слів, а й контекст, відтінки лексики, яку використовую і інші тонкі лінгвістичні аспекти. Більш того, вже існує програмне забезпечення, здатне без участі людини писати новинні статті завдяки машинному навчанню.

**Класифікація текстів** – це розподіл текстових документів за заздалегідь визначеними категоріями (на противагу кластеризації, де безліч категорій заздалегідь невідомо). Класифікація текстів є основою для алгоритмів розпізнавання спаму, так як розпізнавання спаму базується на методах класифікації текстів.

Методи класифікації текстів лежать на стику двох областей - машинного навчання і інформаційного пошуку. Відповідно автоматична класифікація може здійснюватися на основі заздалегідь заданої схеми класифікації і вже наявного безлічі класифікованих документів, або повністю автоматизовано.

При застосуванні підходів машинного навчання, класифікаційне правило будується на основі тренувальної колекції текстів (навчання на прикладах), тобто на датасеті інформації.

Задача класифікації текстів полягає у визначенні приналежності тексту, який розглядається, до одного або до декількох класів. Класифікація може визначатися загальною тематикою текстів, наявністю певних дескрипторів

або виконанням певних умов, за якими визначається приналежність тексту до класу чи декількох класів.

Для кожного класу експерти відбирають текстові масиви, тобто датасети підготовлених даних для навчання, які використовуються системою класифікації в режимі навчання. Після того як навчання закінчено, система за допомогою спеціальних алгоритмів зможе розподіляти вхідні потоки підготовленої текстової інформації за заздалегідь визначеними класами.

Класифікацію можна розглядати як задачу розпізнавання образів, при такому підході для кожного об'єкта виділяються набори ознак. У разі текстів ознаками є слова і взаємозалежні набори слів - терми, які містяться в текстах. Для формування набору ознак для кожного документа використовуються лінгвістичні і статистичні методи. Ознаки групуються в спеціальну таблицю - інформаційну матрицю. Кожен рядок матриці відповідає одному з класів, кожен елемент рядка - одному з ознак; чисельне значення цього елемента визначається в процесі навчання системи класифікації. Коли навчання завершується, приналежність нового тексту до одного з класів встановлюється шляхом аналізу ознак цього тексту з урахуванням відповідних вагових значень. Існуючі алгоритми дозволяють проводити класифікацію з досить високою точністю, проте результати досягаються за рахунок великих розмірів інформаційної матриці, яка визначається загальним числом дескрипторів - термів.

Автоматична класифікація може застосовуватися в таких процедурах інформаційного пошуку як:

1. Фільтрація інформації.

2. Формування тематичних каталогів.
3. Пошук по класах.
4. Реалізація зворотного зв'язку за релевантністю шляхом класифікації результатів пошуку і вибору користувачем релевантних класів.
5. Розширення запитів за рахунок термів, які характеризують тематику класу.
6. Автоматичне реферування.

### Розділ 1.3. Класифікація спаму

Тепер від більш загальної задачі класифікації перейдемо до конкретно задачі класифікації спаму. Отже, множина  $A$  складається з двох різних повідомлень. Кожне повідомлення позначено міткою з безлічі  $M = \{\text{не спам, спам}\}$ . Для того щоб сформулювати поняття різних ознак, ми будемо використовувати модель уявлення, проілюструємо це на дуже простому прикладі. Припустимо, у нас є всього два не спам повідомлення в нашій базі даних (dataset):

Hi, how your life is going?

Hi, what about your life?

Тоді я пропоную створити таблицю зі словами з речень, та їх частотою:

Слово	Частота
hi	2
how	1
what	1
life	2
about	1
your	2
going	1
is	1

Всього ми маємо 11 слів в серед не-спам повідомлень, тоді після нормування ми отримаємо апостеріорну ймовірність слова, використовуючи



спеціальну функцію likelihood estimation. Для прикладу ймовірність слова «how» за умови, що повідомлення не є спамом, буде така:

$$P(h = \text{«how»} \mid M = \text{не спам}) = 1/11$$

У цей момент саме час звернути увагу на наступну проблему. Згадаймо нашу базу з двох не спам повідомлень, і, припустимо, до нас прийшло на класифікацію повідомлення: "hi there", і, припустимо, завжди апріорна вірогідність не-спаму  $P(\text{не спам}) = 1/2$ . Обчислимо ймовірності слів:

$$P(\text{«hi»} \mid \text{не спам}) = 1/11$$

$$P(\text{«there»} \mid \text{не спам}) = 0/11 = 0$$

Очевидно, що ми отримаємо або помилку або негативну нескінченність, тому що логарифм в точці нуль не існує. Якби ми не використали логарифмування, то ми б отримали просто число 0, тобто ймовірність цього повідомлення була б нуль, що в даній ситуації не грає жодної ролі.

Уникнути цього дозволяє така функція, як розмиття за Лапласом - цей метод дозволяє робити розмиття при обчисленні ймовірностей категорійних даних.

Припустимо, що при зчитуванні не спам і спам повідомлень ми знайшли 22 унікальних слова, тоді  $P(\text{«hi»} \mid \text{не спам}) = (1 + 1) / (8 + 1 * 22) = 2/30 = 1/15$ , при коефіцієнті розмиття  $k = 1$ . А нульова ймовірність перестає бути такою:  $P(\text{«there»} \mid \text{не спам}) = (0 + 1) / (8 + 1 * 22) = 1/30$ .

З Байєсівської точки зору, даний метод відповідає математичному очікуванню апостеріорного розподілу, використовуючи в якості апріорного розподілу - розподіл Діріхле, параметризуемое параметром  $k$ .

Тепер перейдемо до технік фільтрації спаму після прикладу на основі наївного Байєсівського принципу.

Техніка фільтрації на основі контенту. Фільтрація на основі контенту зазвичай використовується для створення правил автоматичної фільтрації та класифікації електронних листів, використовуючи підходи машинного навчання, такі як наївна Байєсівська класифікація, нейронні мережі. Цей метод зазвичай аналізує слова, виникнення та розподіл слів і фраз у вмісті електронних листів, а потім використовує створені правила для фільтрації вхідних повідомлень електронної пошти на предмет спаму.

Техніка фільтрації спаму на основі евристики або правил: Цей підхід використовує вже створені правила або евристику для оцінки величезної кількості шаблонів, які зазвичай є регулярними виразами щодо обраного повідомлення. Кілька схожих шаблонів збільшують оцінку повідомлення. На відміну від попереднього типу, спам вираховується з оцінки, якщо жоден із зразків не відповідав спаму. Будь-яка оцінка, що перевищує визначений поріг, фільтрується як спам, в іншому випадку повідомлення вважається не спамом. Хоча деякі правила ранжування не змінюються з часом, то інші правила вимагають постійного оновлення, щоб можна було ефективно справлятися із новими типами спаму, які постійно з'являються.

Попередня техніка фільтрації на основі спаму на основі подібності: Цей підхід використовує методи машинного навчання на основі пам'яті або на основі екземплярів для класифікації вхідних електронних листів на основі

їх подібності до збережених прикладів з датасету. Атрибути електронної пошти використовуються для створення багатовимірного простору, який використовується для побудови нових екземплярів у вигляді точок.

Техніка адаптивної фільтрації спаму: метод виявляє та фільтрує спам, групуючи їх у різні класи на основі отриманих даних. Він розділяє корпус електронної пошти на різні групи, кожна група має певний тип тексту. Тоді проводиться порівняння між кожним вхідним повідомленням на електронній пошті та кожною групою, і визначається відсоток подібності між ними, щоб визначити групу, до якої належить повідомлення, наприклад спам чи ні.

## **РОЗДІЛ 2. ПЕРЕВАГИ ТА НЕДОЛІКИ АНАЛОГІВ**

### **Розділ 2.1 Переваги та недоліки SpamTitan**

Головним плюсом є легкість, при якій можна перевірити повідомлення у спам-карантині. З будь-яким рішенням щодо фільтрації спаму час від часу програма час від часу помиляється та робить неправильні. Огляд результатів показав, що частота помилкових рішень SpamTitan є дуже низькою – менше ніж 0,03%. Однак, при запуску перевірки на звичайних нормальних повідомлень, результати, зазвичай, однозначні та правильні.

У своєму огляді програми SpamTitan можу відзначити простоту, з якою повідомлення з карантину можна безпечно переглядати з консолі адміністратора. Також користувачі можуть переглядати різні показники, такі як: дата повідомлення, одержувач, адресу та розмір повідомлення, а також більш детальну інформацію, використовуючи одну з чотирьох різних вкладок.

SpamTitan дозволяє користувачам переглядати повне HTML-повідомлення, а також увесь заголовок протоколу SMTP. Якщо електронний лист не є спамом, користувачі можуть швидко та легко відпустити повідомлення для доставки та додати до списку електронну адресу відправника, щоб забезпечити завжди надсилання майбутніх повідомлень.

Нажаль, у багатьох рішеннях щодо фільтрації спаму відсутня можливість перегляду повного тіла захопленого повідомлення. Ця функція може заощадити адміністративному персоналу багато часу.

SpamTitan також має гарну систему звітності. SpamTitan дозволяє користувачам конфігурувати широкий спектр типів звітів, надаючи користувачам всю необхідну інформацію для підрахування показників вхідної пошти. Багато провідних додатків для фільтрації спаму не мають такої широкої функціональності звітування про спам. Також є можливість налаштувати звіти для показу всіх повідомлень, які були отримані з моменту останнього запланованого звіту, тому саме така функція відсутня у багатьох інших рішеннях щодо фільтрації спаму.

Доступ до консолі SpamTitan також є простим, незалежно від пристрою з якого ви використовуєте програму. Оскільки панель управління це HTML сторінка, то доступ до неї можна легко отримати з мобільних пристроїв, крім того консоль адміністратора є інтуїтивно зрозумілою та простою у використанні.

Налаштування програми також є швидким та простим процесом, який може зайняти всього 15 хвилин. За словами користувачів: "15 хвилин це надзвичайно швидко, враховуючи те, що раніше у нас був запис MX TTL лише 5 хвилин, тому існує достатня ймовірність того, що якби домени TTL раніше були встановлені на нижчий TTL, налаштування було б ще швидше."

Єдиним недоліком є відсутність функцій перезапису посилань, проте через високу ступінь точності фільтра спаму ця функція не настільки важливою, аби заради неї нехтувати точністю фільтра та обирати інший продукт.

Загалом, SpamTitan - це дуже стабільна проста платформа з безліччю додаткових функцій, завдяки якій життя адміністратора пошти є набагато простішим у керуванні очікуваннями користувачів. Як на мене ця система є недостатньо популярною, оскільки її прості, зручні у використанні, точні та функціональні можливості, повноцінні функції, безумовно, перевершують та перевершують деякі більш відомі рішення на ринку сьогодні.

<b>Можливості</b>	
Архівування	Відновлення даних
Моніторинг електронної пошти	Білий список/чорний список
Керування чергою повідомлень	Маршрутизація
Спаму фільтр	Управління реагуванням
Спільні папки для аккаунтів	Управління підписками

## Розділ 2.2 Переваги та недоліки MailWasher

MailWasher - це ефективний, надійний фільтр спаму, який працює на комп'ютерах та мобільних пристроях. Це одна з двох антиспамських програм, які сумісні з мобільними телефонами та планшетами Android та iOS. Він дублює ваш клієнт електронної пошти, тому ваші безкоштовні акаунти електронної пошти, такі як Gmail та Yahoo Mail, пересилають до нього - це не тільки блокує спам, але й дозволяє переглядати повідомлення з усіх ваших облікових записів в одному місці. MailWasher також працює в Outlook та інших платних клієнтах електронної пошти, які включають desktop клієнти електронної пошти.

Це програмне забезпечення проти спаму має список друзів та чорний список. Під час імпорту контактів MailWasher визнає їх безпечними та автоматично додає їх до списку друзів. Ви можете налаштувати білий список із адресами електронної пошти, які можуть бути зареєстровані як спам, але ви хочете отримувати повідомлення від таких, які використовуються Facebook, eBay або іншими сервісами, які ви використовуєте.

Коли повідомлення вперше надходять через MailWasher, він виділяє ті, що були додані до списку друзів, зеленим кольором - усі інші позначені червоним кольором. Однак ви можете переглянути список і призначити кожне повідомлення як нормальне повідомлення чи як спам, і наступні електронні листи від цих відправників автоматично перенаправлятимуться у потрібну папку. Спам розміщується в кошику, а не видаляється відразу. Це дає вам можливість переглянути відфільтровані повідомлення на випадок, якщо ви випадково позначили його як спам.

Ця програма фільтрації спаму має байєсівський фільтр, тому він може виявляти повідомлення, які використовують ваше ім'я. Він також розпізнає фішинг-листи, схожі на те, що вони надійшли від законної компанії, з якою ви маєте справу, наприклад, вашої кабельної компанії або IRS. Байєсівський фільтр підбирає мову, спеціальні символи. Спамери часто ховаються користуючись фішинг-листами. Спам-фільтр також порівнює адресу електронної пошти відправника з іменем або компанією в підписі повідомлення - якщо вони не відповідають, MailWasher позначає це повідомлення електронною поштою як спам та спрямовує його у вашу вхідну скриньку.

MailWasher фільтрує повідомлення на основі таких критеріїв, як IP-адреса, тип відправника та вкладення, і він посилається на загальнодоступні списки типів, за якими він блокує повідомлення. Найбільшим недоліком програми є те, що вона не може блокувати повідомлення на основі країни походження. Це є проблемою, оскільки деякі країни мають високу концентрацію спамерів.

Якщо вам потрібна допомога в налаштуванні MailWasher, на її веб-сайті є чимало посібників та відеоуроків. Ви також можете зв'язатися з іншими користувачами на форумах спільноти. MailWasher пропонує підтримку по телефону через свого розробника Firetrust. Тому саме як анти-спам додаток, MailWasher має велику користувацьку базу та гарну службу підтримки, що є звичайно, перевагою серед інших конкурентів на ринку анти-спам застосунків.

MailWasher - це окрема програма-фільтр спаму, яка не інтегрується з вашими обліковими записами електронної пошти, натомість виступає як



клієнт електронної пошти, на який переспрямовуються інші ваші облікові записи. Повідомлення розподіляються за кольорами, коли вони надходять, тому ви знаєте, які є більш пріоритетними та на які варто уважніше подивитися. Ця антиспам-програма сумісна з мобільними телефонами та планшетами Android та iOS та має байєсівський фільтр, тому вона розпізнає повідомлення різного рівня складності.

### РОЗДІЛ 3. ЗАСТОСУНОК ДЛЯ ВИЯВЛЕННЯ СПАМУ

Тепер перейдемо до практичної частини моєї курсової роботи, а саме створення додатку для виявлення спаму. Почнемо з використаних технологій. Основними інструментами були мова Python та Tensorflow Keras бібліотека для побудови моделі нейронної мережі для виявлення спаму. Потім я вирішив використати Django Framework для того, аби покласти нейронну мережу на сервер, та зробити мінімальний фронтенд для вводу/виводу інформації.

Тому результатом моєї роботи є клієнт-серверний застосунок з можливістю перевіряти своє повідомлення на те, як воно подібне до спаму, та отримати результат у вигляді «Спам», «Не Спам» та відсоток, наскільки повідомлення схоже на спам.

Для навчання та тестування нейронної мережі я знайшов та використав датасет з кількох тисяч рядків тестових даних, це число невелике, тому точність визначення правильності типу повідомлення іноді не є правдивою, проте, серед введених даних, вона є на високому рівні, та складає майже 100%.

Отже, перейдемо до реалізації застосунку для виявлення спаму на основі машинного навчання.

Першим пунктом у реалізації клієнт-серверного застосунку є саме модель нейромережі:

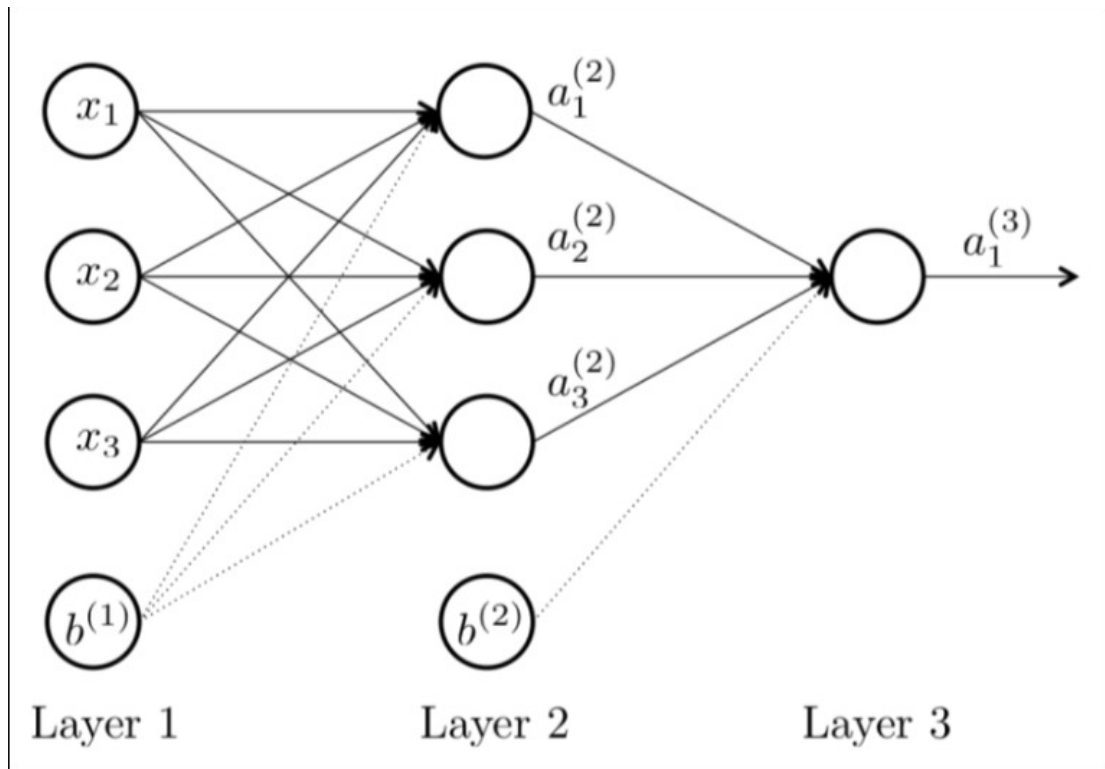
```

83 | train, test = read_all_data()
84 |
85 | train_data_raw, train_labels_raw = split_all_line_data(train)
86 | test_data_raw, test_labels_raw = split_all_line_data(test)
87 |
88 | tokenizer = Tokenizer()
89 | tokenizer.fit_on_texts(train_data_raw)
90 | train_data_seq = tokenizer.texts_to_sequences(train_data_raw)
91 | test_data_seq = tokenizer.texts_to_sequences(test_data_raw)
92 |
93 | x_train = vector_queue(train_data_seq, 4000)
94 | x_test = vector_queue(test_data_seq, 4000)
95 |
96 | y_train = vector_text(train_labels_raw)
97 | y_test = vector_text(test_labels_raw)
98 |
99 | model = models.Sequential()
100 | model.add(layers.Dense(8, activation='relu', input_shape=(4000,)))
101 | model.add(layers.Dense(8, activation='relu'))
102 | model.add(layers.Dense(1, activation='sigmoid'))
103 | model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['accuracy'])
104 |
105 | his = model.fit(x_train, y_train, epochs=8, batch_size=100, validation_split=0.5)
106 |
107 | epochs=range(1, 9)
108 | h = his.history
109 | You, a few seconds ago • Uncommitted changes
110 | results = model.evaluate(x_test, y_test)
111 |

```

Як можна побачити з реалізації, модель є дуже простою, а саме вона має три шари: шар вхідних даних, де я готую дані для нейромережі, та розбиваю дані на токени, невидимий шар, де мережа обраховує введені дані та робить обчислення, розставляючи певні ваги на кожен із нейронів в залежності від параметрів.

Нижче можна побачити візуальне представлення такого роду нейронної мережі:



Далі варто поглянути на реалізацію методу передбачення, а саме простими словами, перевірки на спам:

```

121
122 def predict(test_text):
123     test_text_list = []
124     test_text_list.append(test_text)
125     test_text_sequence = tokenizer.texts_to_sequences(test_text_list)
126     x_test_text = vector_queue(test_text_sequence)
127     pred = model.predict(x_test_text)[0][0]
128
129     print(pred)
130
131     if pred > 0.5:
132         return round(pred, 3), True
133     else:
134         return round(pred, 3), False

```

Як видно за коду, метод приймає текст, в даному випадку ця змінна є типу String, проте Python є мовою з динамічною типізацією у чистому її вигляді. Далі вхідний текст токенізується, та розбивається на дані, які наша

модель розуміє. Після обчислень всередині мережі ми отримуємо результат у вигляді числа від 0 до 1, де згідно нашої дата моделі, 1 – це спам, а 0 – не спам. Тому я обрав за точку перелому число 0.5, де усі результати, менші за це чисто я вважаю за не спам, а більші – є спамом. У результаті обчислень, метод повертає число, яке вірогідністю вхідного тексту бути спамом та вердикт нейронної мережі у вигляді змінної типу boolean. Не зважаючи на невелику кількість тренувальних даних у порівнянні з іншими нейронними мережами, що використовуються у продуктах на ринку, модель є досить точною.

Наступним моментом, на який варто було б звернути увагу, це перевірка форми на сервері, тобто фактично обробник запиту POST з даними, який перевіряє форму, дістає повідомлення з поля виконує перевірку на спам, та повертає результат, виконуючи про цьому render шаблону сторінки.

```

16 def detect(message_data):
17     try:
18         chance, pred = predict(message_data)
19         return chance, pred
20     except ValueError as e:
21         return (e.args[0])
22
23 def datapage(request):
24     detection_result = ''
25     spam_chance = 0.0
26     if request.method=='POST':
27         form=ApprovalForm(request.POST)
28         if form.is_valid():
29             message=form.cleaned_data['message']
30             spam_chance, detection_result = detect(message)
31             messages.success(request, detection_result)
32             print(detect(message))
33
34         form=ApprovalForm()
35
36     return render(request, 'dataform/dataform.html', {'form': form, 'result': detection_result, 'spam_chance': spam_chance})

```

Тепер перейдемо до самої веб-форми з шаблоном:

```

You, a few seconds ago | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vko"
6     <title>Spam Detector</title>
7   </head>
8   <body>
9     {% load crispy_forms_tags %}
10    <div class="container-fluid my-5">
11      <div class="row">
12        <div class="col"></div>
13        <div class="col-sm-4">
14          <h1>Test your message on spam</h1>
15          <br/>
16          <form method="POST">
17            {% csrf_token %}
18            {{form|crispy}}
19            <button type="submit" class="btn btn-primary">Submit</button>
20          </form>
21        </div>
22        <div class="col"></div>
23        <div class="col-sm-4">
24          <div class="messages">
25            {% if result %}
26              <h2>
27                Your message looks like spam. Try to rewrite it.
28              </h2>
29            {% else %}
30              <h2>
31                Cool, it's completely not a spam!
32              </h2>
33            {% endif %}
34            {% if spam_chance %}
35              <h4>
36                Spam chance: {{ spam_chance }}
37              </h4>
38            {% endif %}
39          </div>
40        </div>
41      </div>
42    </div>
43  </body>
44  <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js" integrity="sha384-J6qa4849b1E2+poT4WnyKhv5vZF5SrPo0iEjwBvKU71"
45  <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHa
46  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js" integrity="sha384-wfSDF2E50Y2D1uUdj003u
47 </body>
48 </html>

```

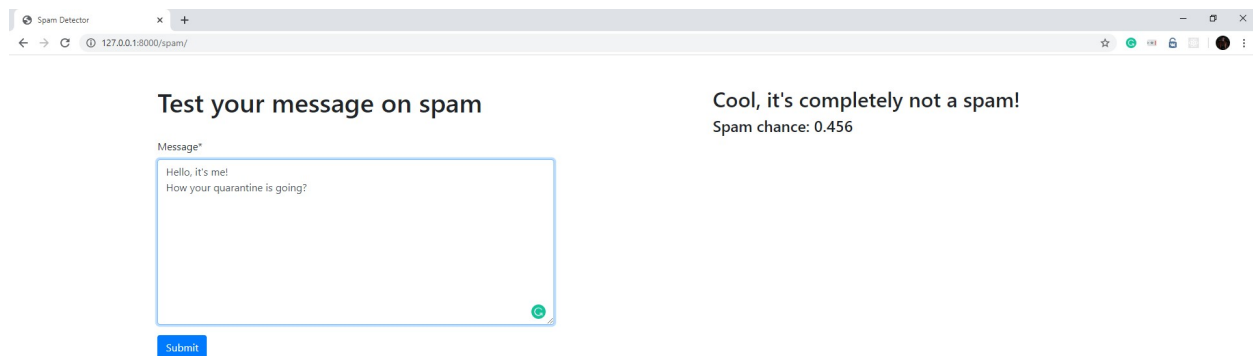
Сама форма не є чимось незвичним, чи дуже складним. Вона лише є клієнтською частиною нашого сервісу для перевірки повідомлень на спам. Як видно шаблонізація проявляється у conditional rendering для декількох полів, які, власне, змінюються у результаті даних, отриманих з сервера після обробки запиту.

## РОЗДІЛ 4. ТЕСТУВАННЯ ЗАСТОСУНКУ

Тестування полягає у введенні різного роду повідомлень, та отриманні відповідних результатів, щодо тесту. Для початку я ввів повідомлення:

«Hello, it's me!

How your quarantine is going?»

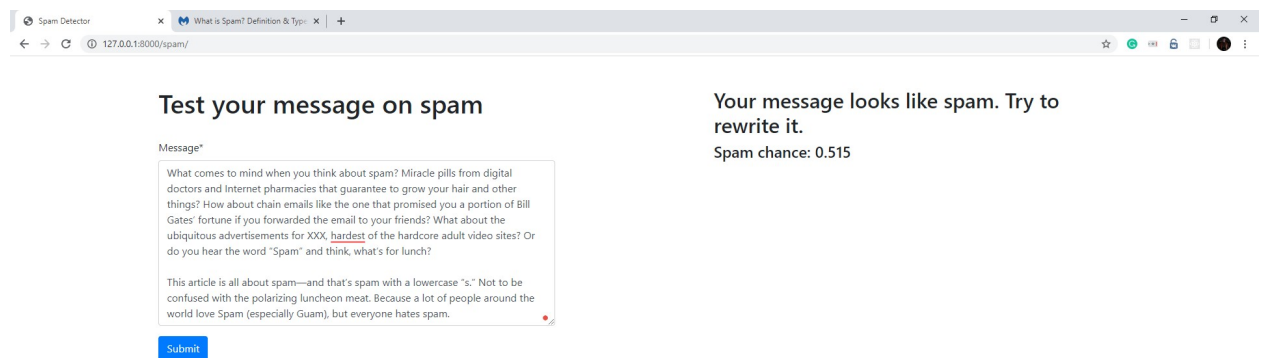


Та як видно зі знімку екрану, отримав результат, що моє повідомлення не є спамом, при цьому шанс, що повідомлення є спамом є досить великим, а саме 45%. Таке число обумовлене невеликою кількістю тестових даних, на яких нейронна мережа була навчена.

Наступним тестом було повідомлення де я намагався зімітувати спамера, тому для цього просто скопіював певний текст з випадкового сайту в інтернеті.

«What comes to mind when you think about spam? Miracle pills from digital doctors and Internet pharmacies that guarantee to grow your hair and other things? How about chain emails like the one that promised you a portion of Bill Gates' fortune if you forwarded the email to your friends? What about the ubiquitous advertisements for XXX, hardest of the hardcore adult video sites? Or do you hear the word "Spam" and think, what's for lunch?

This article is all about spam—and that's spam with a lowercase "s." Not to be confused with the polarizing luncheon meat. Because a lot of people around the world love Spam (especially Guam), but everyone hates spam.»



За результатами тесту можна побачити, що програма розпізнала вхідний текст як спам. Тому ми можемо зробити висновок, що програма працює досить коректно, саме так, як ми очікували.



## ВИСНОВОК

Протягом виконання роботи, я дізнався багато нового для себе, а саме навчився створювати базові моделі для нейронних мереж, готувати для них датасети та взагалі працювати з ними.

Також було недаремним, використання нової для мене мови Python, яку не використовував раніше, проте тепер для подібних задач я розумію, що це є гарним рішенням завдяки великій кількості документації, корисних бібліотек для роботи з нейронними мережами та різних фреймворків.

Далі я розібрався з фреймворком Django, який я використав, як серверне рішення для нейронної мережі та рендер шаблону сторінки з формою відбувається в даному випадку також на сервері.

Загалом, поставлені у роботі цілі, були досягнуті у повному обсязі. Тому результат роботи вважаю успішним.

## Список використаної літератури

1. Opinion Mining on the Web by Extracting Subject-Aspect-Evaluation Relations [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Nozomi Kobayashi, Ryu Iida, Kentaro Inui, Yuji Matsumoto – 2006 – Режим доступу: <https://www.aaai.org/Papers/Symposia/Spring/2006/SS-06-03/SS06-03-018.pdf>
2. The Importance of Neutral Examples for Learning Sentiment [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Koppel, Moshe; Schler, Jonathan – 2005 – Режим доступу: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.9735&rep=rep1&type=pdf>
3. Dphansen. “What Is SQL Server Machine Learning Services (Python and R)?” *Microsoft Docs*, [docs.microsoft.com/en-us/sql/machine-learning/sql-server-machine-learning-services](https://docs.microsoft.com/en-us/sql/machine-learning/sql-server-machine-learning-services).
4. Team, Keras. “Simple. Flexible. Powerful.” *Keras*, [keras.io/](https://keras.io/).
5. TIOBE Index for March 2018 [Електронний ресурс]: [Веб-сайт]. – 2018 – Електронні дані. Режим доступу: <https://www.tiobe.com/tiobe-index/>
6. Text Classification using Naive Bayes [Електронний ресурс] – 2015. – Режим доступу: <http://www.inf.ed.ac.uk/teaching/courses/inf2b/learnnotes/inf2b-learnnote07-2up.pdf>
7. The importance of neutral examples for learning sentiment [Електронний ресурс] : [Веб-сайт]. – Електронні дані. – Rita McCue, Jonathan Schler – 21.10.2005 – <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.84.9735.77>
8. Too much information [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – Режим доступу: <https://www.economist.com/node/18895468>