УДК 681.3, 004.8, 004.432

В.В. Троян, Л.О. Березко

Національний університет "Львівська політехніка",

кафедра електронних обчислювальних машин

**ПРОГРАМНИЙ СЕРВІС ФІЛЬТРАЦІЇ СПАМУ**

*© Троян В. В., Березко Л.О., 2019*

**Розглянуто проблему фільтрації спаму, проаналізовано основний алгоритм фільтрації небажаних та шкідливих повідомлень, запропоновано його покращений варіант. Основною метою є вдосконалення розпізнавання спам-повідомлень. Запропоновано підхід до розгортання сервісу на платформі Google Cloud, що надасть змогу в реальному часі аналізувати існуючі повідомлення та покращувати роботу алгоритму фільтрації при знаходженні спам-повідомлень.**

**Ключові слова: машинне навчання, фільтрація спаму, GCP, Google Cloud.**

 **V.Troian, L.Berezko**

 Lviv Polytechnic National University,

 **Computer Engineering Department**

**SPAM FILTERING SERVICE**

*© Troian V., Berezko L., 2019*

**The proposed article demonstrates how to improve a basic algorithm of filtering of unwanted and harmful messages, and why this problem is so dangerous. The main purpose is to improve the recognition of spam messages. Offered an approach of deploying the service on the Google Cloud platform, allowing real-time analysis of existing messages and improving the filtering algorithm for detecting spam messages.**

**Key words: machine learning, spam filtering, GCP, Google Cloud.**

**Вступ**

Спам - масово відправлені повідомлення, призначені особам, які не виражали бажання їх отримувати, незважаючи на вжиті заходи щодо запобігання цим розсилкам. Боротьба зі спамом йде вже не один десяток років, і незважаючи на успіхи в розробці і дослідженні різних підходів до вирішення даної проблеми, згідно з дослідженнями, частка спаму в світовому і вітчизняному поштовому трафіку все ще велика.

Це вказує на необхідність розробки нових і поліпшення існуючих алгоритмів фільтрації спаму. В даній роботі розглядається алгоритм, що дозволяє вирішити задачу ідентифікації спаму за допомогою машинного навчання.

**Стан проблеми**

Зі спамом знайомий кожен користувач інтернету та електронної пошти. В даний час поштовий трафік засмічений більш ніж на 48%.

Спам є джерелом цілої низки серйозних проблем. Це і зайвий трафік, навантаження на сервер, від якого страждають поштові провайдери та локальні мережі організацій, витрати робочого часу співробітників. Через свою анонімність він є чудовим інструментом для шахрайства, реклами нелегальних, підроблених та контрафактних товарів, розповсюдження порнографії та інших злочинів. Крім того, спам є засобом розповсюдження шкідливого коду. При цьому шкідлива програма може знаходитися як у додатку до листа, так і на сайті, який відкриється за посиланням зі спам-листа. Крім цього, спамом, як платформою, часто користуються фішери, щоб заманити користувачів на підроблені сайти, створені для розкрадання конфіденційної інформації.

На будь-якій поштовій системі, зокрема безкоштовній, стоять спам-фільтри, без яких знайти оригінальну пошту серед потоків реклами було б просто неможливо. Однак, коректно відфільтрувати всі спам-повідомлення за один момент, неможливо, і потрібний постійний аналіз вхідних листів.

**Постановка задачі**

Саме тому постає задача розробити програмний сервіс фільтрації спаму, що буде розгорнутий на платформі Google Cloud. Це надасть змогу в реальному часі аналізувати існуючі повідомлення та покращувати роботу алгоритму фільтрації при знаходженні спам-повідомлень.

**Розв’язання задачі**

Розробку алгоритму фільтрації можна розділити на 5 етапів (рис.1).



Рис.1. Алгоритм машинного навчання сервісу фільтрації спаму

Робимо підготовку тестових даних. Розбиваємо усі повідомлення на 2 підмножини - спам та звичайні листи. Після того як ми отримали дані, ми можемо почати створювати словник, де ми будемо вибирати функції (слова в даному випадку), на основі яких алгоритм згодом вирішить, чи буде вказане повідомлення електронної пошти спамом чи не спамом. Перше, що нам потрібно зробити - це створити словник слів, який буде використаний для нашої моделі. Існує маса можливих способів генерування (вибору) слів, у цьому прикладі ми виберемо перші 2500 найпоширеніших слів, підраховуючи всі повідомлення електронної пошти (з усіх папок). Далі ми будемо витягувати функції з поштових і тестових електронних листів, тому структура результатів готується як вхід до алгоритму “Наївний баєсів класифікатор” на наступному кроці створення моделі прогнозування.

Нехай словник буде містити усі слова (функції), на основі яких ми створимо модель передбачення. Що нам потрібно зробити далі - це підрахувати кількість входжень кожного слова зі словника в електронних листах. Отже, у нас є структуровані та підготовлені дані до їх запуску через алгоритм “Наївний баєсів класифікатор”, для того щоб ми могли отримати модель прогнозування. Наївний баєсів класифікатор також називають "імовірнісним класифікатором", оскільки він будується на обчисленні ймовірності того, що один елемент (електронний лист у нашому випадку) належить до певного класу (класифікації):


де w – слова зі словника, для яких обраховується ймовірність (у нашому випадку - 1);

c – всі входження слів із словника в спамі чи електронних листах без спаму (залежно від того, яка саме ймовірність оцінюється);

V – згладження Лапласа;

P – ймовірність того, що повідомлення є спамом або не-спамом;

Сount(w,c) – кількість входжень w в усіх с.

Сервіс фільтрації спаму складається з 5 основних елементів – служби хостингу веб-аплікацій (Google App Engine), вузла зберігання файлів (Google Storage), вузла побудови моделі ML (TensоrFlow), вузла тренування моделі (Colab) та вузла розгортки моделі (AI Platform). Він отримує дані із поштових клієнтів користувачів за допомогою Gmail API (рис.2).

Для зберігання даних будемо використовувати Cloud Storage - веб-службу хостингу файлів для їх зберігання і доступу через REST в інфраструктурі Google Cloud Platform. Служба поєднує в собі продуктивність і масштабованість хмари Google з поліпшеними можливостями безпеки і спільного використання. Це інфраструктура як послуга (IaaS), подібна сервісу онлайн-зберігання Amazon S3.

Модель будемо будувати за допомогою TensorFlow – відкритої програмної бібліотеки для машинного навчання. Вона була розроблена компанією Google для задоволення її потреб у системах, здатних будувати та тренувати нейронні мережі для виявлення та розшифровування образів та кореляцій, аналогічно до навчання й розуміння, які застосовують люди. Її наразі застосовують як для досліджень, так і для розробки продуктів.

Тренувати модель будемо у Colaboratoty. Це безкоштовне середовище для Jupyter, яке не потребує налаштувань і працює повністю у хмарі. За допомогою Colaboratory можна писати та виконувати код, зберігати та ділитися своїми аналізами та отримувати доступ до потужних обчислювальних ресурсів.

Розгортання системи відбувається на AI Platform. Платформа AI дозволяє розробникам, науковцям даних та інженерам даних легко переносити свої проекти ML від стадії ідеї до стадій використання та розгорки швидко та економічно.



Рис.2. Структурна схема сервісу фільтрації спаму

Було розроблено функцію для тренування алгоритма на мові Python (рис.3).

|  |
| --- |
| def trainingNaiveBayes(trainMarkedWords, trainCategory): """ spamicity：P（Wi|S） :param trainMarkedWords:  :param trainCategory: :return: """ numTrainDoc = len(trainMarkedWords) numWords = len(trainMarkedWords[0]) # P(S) = Probability in spam emails  pSpam = sum(trainCategory) / float(numTrainDoc)  wordsInSpamNum = np.ones(numWords) wordsInHealthNum = np.ones(numWords) spamWordsNum = 0.0 healthWordsNum = 0.0 for i in range(0, numTrainDoc): if trainCategory[i] == 1: # if is spam wordsInSpamNum += trainMarkedWords[i] spamWordsNum += sum(trainMarkedWords[i])  else: wordsInHealthNum += trainMarkedWords[i] healthWordsNum += sum(trainMarkedWords[i]) #gives the conditional probability p(W\_i | S\_x) pWordsSpamicity = np.log(wordsInSpamNum / spamWordsNum) pWordsHealthy = np.log(wordsInHealthNum / healthWordsNum) return pWordsSpamicity, pWordsHealthy, pSpam |

 Рис.3. Лістинг функції тренування моделі

**Висновки**

У даній роботі були розглянуто алгоритм, що дозволяє здійснити фільтрацію спаму методами машинного навчання. Як розширення до нього було запропоновано власний покращений варіант. Виділено найбільш перспективні напрямки для подальшого дослідження і використання в практиці нових методів ідентифікації спаму.

*1.Katiyar D., Khan M. Electronic Mail Spam Filtering//LAP LAMBERT Academic Publishing. – 2019. 2. Huda A. Spam Filtering System using Ant Colony Optimization and Naive Bayesian//LAP LAMBERT Academic Publishing. – 2016. 3. Conway D., White J. Machine Learning for Email: Spam Filtering and Priority Inbox//O'Reilly Media. – 2011. 4. Cormack G. V. Email spam filtering: A systematic review //Foundations and Trends in Information Retrieval. – 2008. 5. Wu C. Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks//Expert Systems with Applications. – 2009. 6. Bratko A., Filipic B., Zupan B. Towards Practical PPM Spam Filtering: Experiments for the TREC 2006 Spam Track // Proceedings of the 15th Text REtrieval Conference (TREC 2006). – 2006. 7. Bratko A., Cormack, G. V., Filipič, B., Lynam, T. R., Zupan, B. Spam filtering using statistical data compression models //Journal of machine learning research. – 2006. 8.Shakhnarovich, G. Darrell, T., Indyk, P. Nearest-neighbor methods in learning and vision. Theory and Practice. – MIT Press. – 2006. 9. Sculley D., Wachman G. M. Relaxed online SVMs for spam filtering //Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. – 2007.*