

МЕТОДИ ТА АЛГОРИТМИ ВИЯВЛЕННЯ ШКІДЛИВИХ ПОСИЛАНЬ

Є. О. Хукаленко¹, І. В. Стьопочкіна¹

¹ Навчально-науковий Фізико-технічний інститут

Анотація

Було вивчено та проаналізовано різні підходи до розпізнавання шкідливих URL-адрес, відібрано характеристики двох груп для подальшого дослідження обраними алгоритмами класифікації, досліджено ефективність роботи відомих класифікаторів на відібраних групах характеристик та запропоновано нову модель класифікації даних, яка відрізняється від існуючих покращеною точністю розпізнавання.

Ключові слова: машинне навчання, стекінг, URL, кібербезпека

Вступ

У сучасному світі інтернет має величезний вплив на життя людей. Інтернет-ресурси надають можливість отримувати різноманітну інформацію та спілкуватися з людьми з усього світу. Проте, з поширенням інтернету з'явилася також проблема шкідливих посилань, які можуть спричинити значні проблеми для користувачів мережі. Виявлення таких посилань є важливим завданням для забезпечення безпеки в інтернеті. У даній статті розглядаються методи та алгоритми виявлення шкідливих посилань, а також їх ефективність та можливості застосування. Розглядаються як методи, що використовують класичні підходи, так і нові моделі на основі машинного навчання та штучного інтелекту.

Результати дослідження можуть бути корисні для розробників програмного забезпечення та власників веб-сайтів для підвищення безпеки користувачів мережі.

1. Формальний опис задачі

Ми формулюємо виявлення шкідливих URL-адрес як завдання двійкової класифікації для двокласного передбачення: «зловмисний» і «безпечний».

Зокрема, враховуючи набір даних із T URL-адресами $\{(u_1, y_1), \dots, (u_T, y_T)\}$, де $t = 1, \dots, T$ представляє URL-адресу з навчальних даних, а $y_t \in \{1, -1\}$ – відповідну мітку, де $y_t = 1$ позначає шкідливу URL-адресу, а $y_t = -1$ позначає безпечну URL-адресу.

Суть автоматичного виявлення шкідливих URL-адрес полягає в двох аспектах: (1) Представлення даних: отримання відповідного представлення ознак: $u_t \rightarrow x_t$, де $x_t \in \mathbb{R}^d$ – d -вимірний вектор ознак, що представляє URL-адресу; і (2) Машинне навчання: вивчення функції передбачення $f: \mathbb{R}^d \rightarrow \mathbb{R}$, яка прогнозує мітку класу для будь-якого екземпляра URL-адреси x за допомогою нале-

жного представлення ознак.

Мета машинного навчання для виявлення шкідливих URL-адрес – максимізувати точність прогнозування. Обидві перераховані вище складові важливі для досягнення цієї мети. У той час як перша частина представлення ознак часто базується на знаннях предметної області та евристичах, друга частина зосереджена на навчанні моделі класифікації за допомогою підходу оптимізації на основі даних.

Першим ключовим кроком є перетворення URL-адреси u у вектор ознак x , де можна розглянути декілька типів інформації та використовувати різні методи. Вони варіюються від лексичної інформації (довжина URL-адреси, слова, що використовуються в URL-адресі тощо) до інформації на основі хоста (інформація WHOIS, IP-адреса, місцезнаходження тощо).

Після того, як інформація зібрана, вона обробляється для збереження у векторі ознак x . Після отримання вектора ознак x для навчальних даних, щоб дізнатися функцію передбачення $f: \mathbb{R}^d \rightarrow \mathbb{R}$, її зазвичай формулюють як задачу оптимізації так, що точність виявлення максимізується. Функція f (зазвичай) параметризована d -розмірним вектором ваги w , таким, що $f(x) = (w \cdot x)$. Нехай $\hat{y}_t = \text{sign}(f(x_t))$ позначає передбачення мітки класу, зроблене функцією f . Кількість помилок, допущених моделлю прогнозування для всіх навчальних даних, визначається так: $\sum_{t=1}^T \mathbb{I}_{\hat{y}_t \neq y_t}$ де $\mathbb{I}$ — показник, який оцінюється як 1, якщо умова істинна, і 0 — в іншому випадку. Оскільки індикаторна функція не є опуклою, оптимізацію може бути важко вирішити. У результаті часто визначається опукла функція втрат, яка позначається $\ell(f(x), y)$, а всю оптимізацію можна сформулювати так: $\min_w \sum_{t=1}^T \ell(f(x_t), y_t)$.

2. Порівняння існуючих робіт

У літературі існує багате сімейство алгоритмів машинного навчання, які можна застосувати для ви-

явлення шкідливих URL-адрес. Після перетворення URL-адрес у вектори ознак багато з цих алгоритмів навчання зазвичай можна застосувати для навчання прогнозувальної моделі досить просто. Однак, щоб ефективно вирішити цю проблему, були також докладені деякі зусилля щодо розробки конкретних алгоритмів навчання, які або використовують властивості, що демонструються навчальними даними шкідливих URL-адрес, або вирішують деякі конкретні проблеми, з якими стикається програма.

В роботі [1] автори вивчають поведінкові фактори як того, хто публікує URL-адресу, так і того, хто взаємодіє з URL-адресою. Основна припущення полягає в тому, що цими поведінковими сигналами може бути складніше маніпулювати, ніж традиційними сигналами. Авторі пропонують та оцінюють п'ятнадцять функцій на основі кліків і публікацій. Фінальний результат такого підходу: precision (0.86), recall (0.86), і AUC (0.92).

В роботі [2] автори використовували інформацію домену про посилання для розпізнавання фішингових сторінок. У цьому запропонованому методі в першу чергу збирались всі прямі і непрямі посилання на обраний сайт. Далі, використовуючи вихідний код сторінки, всі домени, пов'язані з прямими посиланнями збирались і поміщались в набір S1. Потім непрямі посилання збирались і поміщались в набір S2, після цього обидва набори об'єднувались і збирались тільки загальні домени, з яких згодом, за допомогою DNS пошуку, витягувались IP-адреси цих доменів. Результат дослідження показує, що асигасу, FP, FN, TN і TP запропонованого методу 99.62, 0.32, 0.5, 99.5 і 99.67 відсотків. Незважаючи на таку високу точність прогнозування, залежність цього методу від різних зовнішніх методів типу DNS-пошуку і пошукових систем може вплинути на його ефективність.

В роботі [3] автори використовують конструкцію моделі класифікатора Transformer для виявлення шкідливих URL-адрес. Авторі порівнюють свою модель із шістьма іншими моделями (дерево рішень, випадковий ліс, багатопаровий перцептрон, XGBoost, SVM та Auto Encoder) та показують, що їх модель є кращою у precision, recall та асигасу за всі інші моделі на їх наборі даних. Фінальний результат такого підходу: precision (0.98), recall (0.96), і асигасу (0.97). Особливістю підходу є те що використовується лише URL-адреса без попередньої обробки. Дану роботу можна вважати продовженнями роботи [4], оскільки вона розвиває ідею застосування sequential models для класифікації шкідливих URL-адрес.

3. Алгоритм розпізнавання

У даній роботі пропонується поєднати сильні сторони існуючих алгоритмів та підходів до виділення ознак із URL-адрес шляхом комбінування моделей методом стекінгу.

Стекінг передбачає навчання алгоритму для поєднання прогнозів кількох інших алгоритмів машин-

ного навчання. Спочатку всі інші алгоритми навчаються з використанням доступних даних, потім алгоритм комбінатора навчається робити остаточний прогноз, використовуючи всі передбачення інших алгоритмів як додаткові вхідні дані.

Стекінг зазвичай забезпечує кращу точність, ніж будь-яка одна з навчених моделей.[5] Він успішно використовується як для навчання з учителем, так і для навчання без учителя.

На відміну від більш традиційних методів ансамблювання бустингу і берінгу (boosting, bagging) для стекінгу краще використовувати алгоритми різної «природи», тобто такі що мають під собою різні початкові припущення про модель даних.

Як видно із існуючих робіт, є досить багато ознак, які можна отримати із однієї тільки URL-адреси із використанням сторонніх сервісів, таких як, наприклад, whois. У роботі [6] автори виділяють аж цілих 5 категорій таких ознак:

- Лексичні особливості;
- Ознаки популярності посилання;
- Ознаки вмісту веб-сторінки;
- Ознаки DNS;
- Ознаки мережі.

Також із прикладів існуючих підходів можна побачити, що за допомогою нейронних мереж ми можемо передати сам URL, як самостійний набір ознак.

4. Опис характеристик

Далі у таблицях 1 – 2 подано опис характеристик, що будуть використані у розробці алгоритму.

5. Аналіз результатів дослідження

5.1. Інструментарій та набір даних

Для роботи був використаний набір даних, що містить 96,018 екземплярів, з яких 48,009 надійних і 48,009 фішингових. Дані отримані з сервісу Aalto University Datasets. Обробка датасету і вся подальша робота виконувалась на мові програмування Python. Додаткова інформація про домен була отримана із сервісу whois.

5.2. Результати прогнозування проміжних моделей

Для початку розділемо ознаки на дві категорії і перевіримо якість різних моделей машинного навчання на цих ознаках. Для перевірки візьмемо декілька широкозастосовуваних моделей машинного навчання, а саме: kNN, Naïve Bayes, Decision Tree, Random Forest, SVM, XGBoost, Logistic Regression.

Моделі оцінюються по трьом вимірам: Precision, Recall і AUC. Процедура оцінки – кросс-валідація із параметром k=5.

Кросс-валідація — це метод оцінки аналітичної моделі та її поведінки на незалежних даних з найбільш рівномірним використанням наявних даних. В основі методу лежить розділення вихідної множини даних на k приблизно рівних блоків. Після цього

Таблиця 1. Характеристики адресного рядка

Назва характеристики	Опис
Using the IP Address	IP адреса використовується замість доменного імені, іноді трансформується у шістнадцятковий вигляд
Dot Count	Кількість точок в URL – кількість субдоменів
Digit Count	Кількість цифр у URL-адресі
Special Character Count	Використання символів зі списку (;, +=, _, ?, =, &, [,])
Hyphen Count	Дефіс рідко використовується у перевірених адресах
Double Slash Count	Наявність у URL «//» означає що користувач буде перенаправлений на іїший сайт
Single Slash Count	Кількість одинарних слешів у адресі
URL Length	Довжина URL-адреси, довгий URL може бути потрібен для приховання підозрілої частини

Таблиця 2. Мережеві характеристики (DNS, whois)

Назва характеристики	Опис
Resolved IP count	Кількість IP-адрес, які можна розпізнати для домену URL-адреси
Name server count	Кількість серверів імен, які обслуговують домен
Name server IP count	Кількість IP-адрес, з якими пов'язані ці сервери імен.
Registered Date	Дата реєстрації (whois)
Expiration Date	Термін дії (whois)
Update Date	Дата останнього оновлення (whois)

Таблиця 3. Результати тесту для характеристик адресного рядка

Алгоритм	AUC	Precision	Recall
kNN	0.813	0.820	0.811
Naïve Bayes	0.801	0.800	0.801
Decision Tree	0.761	0.753	0.766
Random Forest	0.810	0.805	0.815
SVM	0.705	0.696	0.715
XGBoost	0.805	0.789	0.813
Logistic Regression	0.764	0.723	0.786

на k-1 блоках проводиться навчання моделі, а останній блок використовується для тестування. Процедура повторюється k разів, при цьому на кожному проході для перевірки вибирається новий блок, а навчання проводиться на інших.

Кросс-валідація має важливу перевагу перед застосуванням однієї множини для навчання та однієї для тестування моделі: якщо при кожному проході оцінити вихідну помилку моделі й усереднити її за всіма проходками, то отримана її оцінка буде більш достовірною.

Далі розглядаються тільки характеристики адресного рядка, результати представлені в таблиці 3. Результати є усередненням за кросс-валідацією

Далі розглядаються тільки мережеві характеристики, результати представлені у таблиці 4.

Таким чином ми обираємо для стекингу два алгоритми першого рівня – kNN і XGBoost для використання на характеристиках адресного рядка і мережевих характеристиках відповідно.

5.3. Результати побудови стекингу моделей

Остаточною метою даної роботи є побудова комбінованого класифікатора методом стекингу для покращення метрик наявних простих класифікаторів. За результатами перевірки на різних наборах характеристик було визначено кращі проміжні класифікатори – kNN та XGBoost. Третім класифікатором в ансамблі використаємо абсолютно інший за природою класифікатор на основі нейромережевої архіте-

ктури transformer, що описаний у роботі [3].

Задамо умовну початкову точність від якої будемо відштовхуватись: модель transformer на нашому наборі даних показала наступні результати precision (0.91), recall (0.89), і AUC (0.9).

Наступним кроком є вибір моделі, що буде моделлю стекингу. Результати стекингу трьох моделей різними моделями верхнього рівня наведено в таблиці 5.

В ході експерименту було встановлено, що завдяки новій схемі класифікації спостерігається покращення таких параметрів як AUC, precision та recall. Встановлене підвищення метрик є для більшості алгоритмів другого рівня. Найкращий результат показав алгоритм Logistic Regression із прирістом AUC у 3

Висновки

У роботі було вивчено та проаналізовано різні підходи до розпізнавання шкідливих URL-адрес, вивчено актуальні методи класифікації за допомогою алгоритмів машинного навчання. Відібрано характеристики двох груп для подальшого дослідження обраними алгоритмами класифікації. Досліджено ефективність роботи відомих класифікаторів на відібраних групах характеристик. Запропоновано нову модель класифікації даних, яка є стекингом декількох класифікаторів, яка відрізняється від існуючих покращеною точністю розпізнавання.

Таблиця 4. Результати тесту для мережевих характеристик

Алгоритм	AUC	Precision	Recall
kNN	0.851	0.871	0.831
Naïve Bayes	0.826	0.813	0.847
Decision Tree	0.803	0.798	0.808
Random Forest	0.860	0.844	0.876
SVM	0.789	0.795	0.781
XGBoost	0.872	0.874	0.872
Logistic Regression	0.801	0.790	0.810

Таблиця 5. Результати стекінгу

Алгоритм	AUC	Precision	Recall
kNN	0.901	0.905	0.891
Naïve Bayes	0.891	0.890	0.893
Decision Tree	0.910	0.905	0.915
SVM	0.897	0.889	0.906
Logistic Regression	0.927	0.930	0.926

Перелік використаних джерел

1. *Cao C., Caverlee J.* Detecting Spam URLs in Social Media via Behavioral Analysis // *Advances in Information Retrieval* / за ред. А. Hanbury, G. Kazai, A. Rauber, N. Fuhr. — Cham : Springer International Publishing, 2015. — С. 703—714. — ISBN 978-3-319-16354-3.
2. *Ramesh G., Krishnamurthi I., Kumar K. S. S.* An efficacious method for detecting phishing webpages through target domain identification // *Decision Support Systems*. — 2014. — Т. 61. — С. 12—22. — ISSN 0167-9236. — DOI: <https://doi.org/10.1016/j.dss.2014.01.002>. — URL: <https://www.sciencedirect.com/science/article/pii/S0167923614000037>.
3. *Xu P.* A Transformer-based Model to Detect Phishing URLs. — 2021. — arXiv: [2109.02138](https://arxiv.org/abs/2109.02138) [cs.CR].
4. URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection / H. Le, Q. Pham, D. Sahoo, S. C. H. Hoi. — 2018. — arXiv: [1802.03162](https://arxiv.org/abs/1802.03162) [cs.CR].
5. *Wolpert D. H.* Stacked generalization // *Neural Networks*. — 1992. — Т. 5, № 2. — С. 241—259. — ISSN 0893-6080. — DOI: [https://doi.org/10.1016/S0893-6080\(05\)80023-1](https://doi.org/10.1016/S0893-6080(05)80023-1). — URL: <https://www.sciencedirect.com/science/article/pii/S0893608005800231>.
6. *Choi H., Zhu B. B., Lee H.* Detecting Malicious Web Links and Identifying Their Attack Types // *Proceedings of the 2nd USENIX Conference on Web Application Development*. — Portland, OR : USENIX Association, 2011. — С. 11. — (WebApps'11).