УДК 004.415.52, 004.89, 004.492 http://doi.org/10.15407/pp2024.02-03.253

Б.О. Панчук

ФОРМАЛЬНА ВЕРИФІКАЦІЯ НЕЙРОННИХ МЕРЕЖ ГЛИБОКОГО НАВЧАННЯ

У цій роботі представлено спосіб формальної верифікації властивостей нейронних мереж за допомогою “Satisfiability Modulo Theories” (SMT) розв’язувача. Цей підхід дозволяє математично доводити специфічні властивості нейромереж, що робить їх більш передбачуваними. У процесі дослідження було розроблено новий метод спрощення обчислювального графа нейромережі над певними регіонами вхідного простору, шляхом заміни кусково-лінійних функцій активації нейронів на підмножину їхніх лінійних сегментів. Запропонований спосіб оптимізації ґрунтується на гіпотезі існування значно простішої інтерпретації нейромережі на обмежених діапазонах вхідних даних. У роботі пропонується спосіб отримання такої спрощеної інтерпретації через понейронне розв’язування локальних SMT-задач з інкрементальним спрощенням графу нейромережі. Ця оптимізація дозволила значно прискорити алгоритм верифікації порівняно з розв’язанням єдиної SMT-задачі над суцільним неоптимізованим графом мережі. Описаний метод можна застосовувати для верифікації довільних повнозв’язних нейромереж глибокого навчання з кусково-лінійними функціями активації.

Для практичної демонстрації результатів створений метод верифікації було застосовано для перевірки роботи нейромережі-аналізатора мережевого трафіку, метою якого є виявлення активності ботнетів. Модель класифікації перевірялась на стійкість до «змагальних атак» – спроб зловмисника уникнути виявлення, вносячи спеціальні збурення в дані, що класифікуються. В роботі було проведено автоматизовану перевірку коректності роботи класифікатора над сумнівними регіонами даних, які розміщені біля межі ухвалення рішень. Отримані результати роблять важливий внесок у розвиток застосування штучного інтелекту в царині кібербезпеки, де інтерпретація та передбачуваність є критично необхідними. Крім того, запропонований метод понейронного спрощення є перспективним напрямком для подальшого розвитку верифікації нейронних мереж.

Ключові слова: нейронні мережі, формальна верифікація, satisfiability modulo theories, задача виконуваності обмежень, системи виявлення вторгнень, аналіз мережевих даних.

B.O. Panchuk

FORMAL VERIFICATION OF DEEP NEURAL NETWORKS

This paper introduces a method for the formal verification of neural networks using a Satisfiability Modulo Theories (SMT) solver. This approach enables the mathematical validation of specific neural network properties, enhancing their predictability. We propose a method for simplifying a neural network’s computational graph within certain input space regions. This is achieved by replacing neurons’ piecewise-linear activation functions with a subset of their linear segments. This optimization hypothesizes a simpler interpretation of a neural network over limited input data ranges. The simplified interpretation is derived from the incremental simplification of the neural network graph, achieved by solving local SMT tasks on a neuron-by-neuron basis. This optimization significantly speeds up the verification algorithm compared to solving a single SMT task over the entire unoptimized network graph. The method is applicable to any deep neural networks with piecewise-linear activation functions.

The approach’s effectiveness was demonstrated by automatically verifying a network traffic classifier specializing in botnet activity detection. The classification model was tested for robustness against adversarial attacks, where attackers attempt to evade detection by introducing specially crafted disturbances into the network data. The verification procedure was conducted over regions in the feature-space near the classifier’s decision boundary. The results contribute to the prospects for more active application of artificial intelligence models in cybersecurity, where result predictability and interpretability are crucial. Additionally, the neuron-wise simplification technique proposed is a promising direction for further development in neural network verification.

Key words: neural networks, formal verification, satisfiability modulo theories, constraints satisfaction, intrusion detection systems, network traffic analysis.

Вступ

На сьогодні написано чимало робіт із застосування ШІ у сфері кібербезпеки. Зокрема, моделі ШІ неодноразово використовувалися для аналізу інтернет трафіку з метою виявлення мережевих атак (IDS, intrusion detection system). Значно менше уваги приділено стійкості таких моделей до так званих «змагальних атак». Такі атаки передбачають спробу ухилення від виявлення шляхом збурення вхідних даних з метою спричинення помилкової класифікації, водночас залишаючи дані правдоподібними. В роботах, де зазвичай ця проблема розглядається, використовуються класичні емпіричні методи статистичної оцінки стійкості моделей на штучно згенерованих даних. Недоліком такого підходу, є те, що в результаті не надається жодних абсолютних гарантій. Відповідно поведінка IDS залишається непередбачуваною на даних, які не присутні в тестовій вибірці, що є вагомим контраргументом до використання ШІ в цій сфері.

Нещодавно почав набувати розвитку підхід «формальної верифікації» нейронних мереж, за якого вимоги задаються формально у формі обмежень, що накладаються на входи та виходи моделі. Після цього відбувається автоматична математична перевірка виконання обмежень моделлю на всіх можливих вхідних даних. Такий підхід принципово відрізняється від перевірки результатів для окремих штучно-згенерованих прикладів, адже отримані результати надійності розповсюджуються на цілий спектр наборів вхідних даних.

Мета статті

Першочерговою метою цієї роботи було створення методу для отримання математичних гарантій стійкості глибоких нейронних мереж-класифікаторів. Отриманий метод верифікації, може бути застосований для довільних нейромереж прямого розповсюдження з більше, ніж одним прихованим шаром та кусково-лінійною функцією активації прихованих нейронів.

Метою експериментальної частини роботи була демонстрація застосування даного методу для верифікації попередньо навченої нейромережі, що спеціалізується на класифікації мережевих даних з метою виявлення шкідливих дій. Сам алгоритм має бути оптимізовано з метою уникнення комбінаторного вибуху слкладності при аналізі розв'язувачем графу нейромережі.

Методи верифікації властивостей моделей

Найбільш поширеним методом є емпіричні оцінки характеристик моделі. Перевірка даних властивостей здійснюється шляхом отримання результатів роботи моделі над фіксованою множиною точок з вхідного простору. Після чого порівнюють отримані результати із заздалегідь відомими та підраховуються статистичні оцінки їх схожості. Класичними показниками при такій оцінці є, точність, влучність, частка позитивно-негативних результатів, відсоток помилок над множиною змагальних прикладів і т.д. Як і випливає з назви, такі властивості не надають жодних гарантій якості моделі, а лише показують статистику роботи моделі на доступній вибірці.

Інший метод верифікації є перевірка виконання глобальних обмежень (constraints satisfaction). Такі властивості визначаються у формі системи жорстких обмежень накладених на простір вхідних та вихідних даних моделі. А верифікація здійснюється не на фіксованому наборі вхідних даних, а на повному теоретично можливому підпросторі з простору входів. Прикладом такої властивості у випадку багатокласової класифікації може бути певна умова. А саме: якщо модель класифікації вже віднесла певний об'єкт до якогось класу, то показник впевненості для всіх інших класів має буде нижчим за певний поріг.

Метод, який розглядається в даній роботі – це верифікація локальної надійності (robustness). Якщо модель є локально надійною (стійкою), очікується, що за умови, коли для певного елементу з області входу модель продукує коректний результат, то під час внесення відносно незначних пер-

турбацій в значення ознак даного вхідного елементу, результат змінюється лише в очікуваних межах. Наприклад, у випадку задачі класифікації, обидві – оригінальна та змінена точки мають бути віднесені моделлю до однакового класу. Ця властивість часто розглядається в контексті стійкості моделі до змагальних атак, однак може бути розширена до перевірки узагальнюючих властивостей моделі щодо довільних збурень вхідних даних в допустимих межах. У роботі [1] наводяться 4 типи класів надійності в порядку посилення їхніх умов та обмежень:

1. **Надійність класифікації**, що передбачає збереження класу для всіх елементів з околу точки з простору вхідних даних.
2. **Стандартна надійність** полягає в тому, що для околу вхідної точки вихідне значення моделі варіюватиметься також у певному околі.
3. **Надійність Ліпшица** вирізняється тим, що під час внесення змін у значення ознак вхідного елементу, вихід змінюється пропорційно.
4. **Жорстка надійність** класифікації передбачає, що в околі вхідної точки значення на виході класифікатора для коректного класу буде вище певного порогу.

За реалізацією методи верифікації можна розділити на п’ять наступних класів.

Емпіричне тестування. Найпростіший клас методів, суть якого - перевірка результатів виходів моделі на множині заздалегідь відомих вхідних елементів. Тестова множина може бути отримана разом із навчальним набором даних, або згенерована через тривіальні методи розширення даних (як-от деформації, накладання випадкових шумів тощо). До цього класу належать й інші класичні методи такі, як фазинг [2] чи тестування на основі покриття [3].

Пошук контрприкладів. У даному класі різними евристичними методами здійснюється пошук правдоподібних вхідних елементів, для яких модель порушує властивості, що перевіряються. Одним із класичних способів є перетворення початкових вірно класифікованих елементів вхідної вибірки на “змагальні приклади” за допомогою градієнтних методів (наприклад, “методом швидкого градієнту” (FGSM) [4]). Даний клас методів однак не обґрунтовує стійкості моделі, а лише може виявити її вразливість.

Рішення задачі оптимізації. Якщо модель можна звести до системи лінійних рівнянь та нерівностей, то для верифікації можна застосувати методи лінійного чи змішаного цілочислового програмування. Слід зауважити однак, що через наявність нелінійних функцій активації нейронів, як правило, неможливо застосувати даний клас методів до нейронних мереж “як-є”. Відповідно їх зазвичай використовують у комбінації з іншими підходами, такими як застосування SMT (Satisfiability Modulo Theories) розв’язувача [5][6].

Формування абстрактної інтерпретації моделі. Цей метод зазвичай доповнює інші. Його суть у перебудові структури нейронної мережі, що розглядається як семантично схожа, але простіша [7]. Результуюча модель є під- або над- апроксимацією оригінальної. Мета такої операції – спрощення обчислювальної складності у подальшому доведенні виконання (чи порушення), накладених обмежень іншими методами. Очікується, що з виявлення порушення обмеження для підапроксимованої моделі, випливає порушення даного обмеження і для оригінальної. Водночас доведення стійкості підапроксимованої моделі, не гарантує стійкості оригінальної. У випадку надапроксимації все з точністю до навпаки.

Доведення за допомогою SMT – розв’язувача. В даному методі обчислювальний граф моделі представляється у вигляді SMT формули. Далі на входи та виходи моделі накладаються певні обмеження, після чого SMT-розв’язувач доводить або спростовує їх виконуваність. Зазвичай цей метод вимагає попередньої апроксимації всіх нелінійних функцій активації нейронів через лінійні сегменти.

В даній роботі застосовується комбінація методу абстрактної інтерпретації та

доведення властивостей моделі за допомогою SMT розв'язувача (Microsoft Z3 [8]).

1. Постановка задачі верифікації в околі точки

Ми продемонструємо автоматизацію перевірки локальної стійкості (надійності) класифікатора мережевого трафіку методом верифікації у відносному околі точки з вхідного простору. Класифікатор вважається стійким в околі точки, якщо для будь-якої точка в межах даного околу зберігається вихідний клас.

Нехай x – n-вимірний вектор на вхідному просторі ознак, F – множина індексів ознак, для яких здійснюється верифікація, ε – параметр, що задає величину околу ($0 \le \varepsilon \le 1$). Тоді позначимо **відносний** окіл точки x за ознаками F як $E(x, \varepsilon, F)$.

$$E\left(x,\ \varepsilon,\ F\right) \subset \mathbb{R}^{n}_{\ge 0}, \quad (1)$$

Будемо важати, що в разі, якщо точка із вхідного простору x' лежить в цьому околі:

$x' \in E\left(x,\ \varepsilon,\ F\right)$, то

$$\begin{cases} (1-\varepsilon)\cdot x_i \le x'_i \le (1+\varepsilon)\cdot x_i, & якщо\ i \in F \\ x'_i = x_i, & інакше \end{cases} \quad (2)$$

Нехай N – бінарний класифікатор, t – вихідний поріг класифікації. Тоді модель вважатимемо стійкою у відносному околі точки x якщо:

$$\forall x' \in E(x,\varepsilon,F), \begin{cases} N(x) > t \Rightarrow N(x') > t \\ N(x) \le t \Rightarrow N(x') \le t \end{cases} \quad (3)$$

2. Задача виконуваності обмежень

Для доведення (або спростування) властивості стійкості моделі у відносному околі точки, зведемо задачу верифікації до задачі виконуваності системи обмежень. У систему включатимемо 3 класи обмежень:

Обмеження належності точки до відносного околу над підмножиною ознак визначають базову форму регіону вхідних даних над яким проводиться верифікація.

$$C_{eps} : \left(\bigwedge_{i \in F} \left((1-\varepsilon)\cdot x_i \le x'_i \le (1+\varepsilon)\cdot x \right) \right) \wedge$$

$$\wedge \left(\bigwedge_{i \notin F} \left(x'_i = x_i \right) \right)$$

Семантичні обмеження над вхідними ознаками необхідні для забезпечення правдоподібності вхідних даних. Ці обмеження відсікають теоретично недосяжні регіони в просторі ознак шляхом встановлення міжознакових відношень.

$$C_{semantic} : \left(\bigwedge x'_i > x'_j \right) \wedge \left(\bigwedge x'_k < x'_l \right) \left(\bigwedge x'_h = x'_m \right)$$

де $i, j, k, l, h, m \in F$.

Вихідне обмеження – це обмеження, яке констатує, що передбачуваний моделлю клас для оригінальної точки та точок з її околу мають співпадати.

$$C_{output} : \begin{cases} N(x') > t, & якщо\ N(x) > t \\ N(x') \le t, & якщо\ N(x) \le t \end{cases} \quad (4)$$

Тоді, враховуючи всі зазначені обмеження, сформуємо задачу виконуваності у вигляді:

$$\forall x', C_{eps}(x') \wedge C_{semantic}(x') \rightarrow C_{output}(x') \quad (5)$$

Якщо SMT-розв'язувачу вдається довести загальнозначимість формули вище, то модель є стійкою на заданому околі з урахуванням накладених семантичних обмежень.

Додатково зазначимо, що на практиці може бути корисним позбавлення квантора загальності. В такому разі задачу виконуваності можна переформулювати у тотожну задачу «невиконуваності» (тобто доведення, що не існує жодної інтерпретації, за якого формула є істиною). Для цього виразимо імплікацію через диз'юнкцію, після чого до отриманого виразу застосуємо заперечення і друге правило де Моргана. Тепер задача верифікації звелася до доведення невиконуваності оберненої формули:

$$C_{eps}(x') \wedge C_{semantic}(x') \wedge \neg C_{output}(x') \quad (6)$$

Розглянемо приклад обмежень для класифікатора мережевих потоків. Нехай мережевий потік характеризується вектором із 5 ознак `["Duration", "IAT Std", "IAT Min", "IAT Max", "Bytes"]`. Маємо бінарний класифікатор потоків N, з порогом $t = 0.51$, у разі перевищення якого гіпо-

теза класифікації приймається і потік зараховується до класу «шкідливих». Припустимо, необхідно довести стійкість моделі *N* у відносному околі «шкідливого» потоку *x = [0.3, 0.1, 0.01, 0.05, 0.5]* за варіювання міжпакетних інтервалів (IAT – "inter-arrival time") для *ε = 0.4*. Тоді індекси ознак, для яких проводиться верифікація: *F = {1, 2, 3}* (нумерація починається з 0). Єдине семантичне обмеження над ознаками: *"IAT Min" ≤ "IAT Max"*. Сформулюємо обмеження для цієї задачі:

Обмеження відносного околу:

$$C_{eps}:$$
$$x'_0 = 0.3 \wedge$$
$$(1-0.4)\cdot 0.1 \le x'_1 \wedge x'_1 \le (1+0.4)\cdot 0.1 \wedge$$
$$(1-0.4)\cdot 0.01 \le x'_2 \wedge x'_2 \le (1+0.4)\cdot 0.01 \wedge$$
$$(1-0.4)\cdot 0.05 \le x'_3 \wedge x'_3 \le (1+0.4)\cdot 0.05 \wedge$$
$$x'_4 = 0.5$$

Семантичне обмеження над ознаками $C_{semantic}: x'_2 \le x'_3$.

Вихідне обмеження $C_{output}: N(x') > 0.51$.

Для автоматичної перевірки виконуваності даних обмежень необхідно створити символьне представлення повного обчислювального графу нейронної мережі *N(x)* у вхідній мові SMT-розв'язувача. В наступному розділі описано принцип побудови такого представлення для подальшої обробки розв'язувачем Microsoft Z3.

3. Застосування Z3 для верифікації нейронної мережі

В даній роботі ми обмежились повнозв'язною мережею з 50 входами, 3 прихованими рівнями по 24 нейронами в кожному та одним вихідним нейроном. Функції активації прихованих рівнів - ReLU, а вихідного нейрону - Sigmoid.

Для аналізу нейронної мережі SMT-розв'язувачем Z3 необхідно побудувати її тотожну інтерпретацію через структурні елементи Z3. Для взаємодії із Z3 в даній роботі використовується інтерфейс Z3Py (Python Z3 API).

Побудова абстрактного синтаксичного дерева нейромережі відбувається поріднево, починаючи від вхідного рівня і закінчуючи вихідним, за базовими правилами обрахунку активацій нейронів для повнозв'язних мереж: вектор-рядок активацій попереднього рівня множиться на вагову матрицю наступного з додаванням вектора-рядка зміщень. Після чого до отриманого вектора поелементно застосовується функція активації. Значення кожного параметра моделі «загортаються» в об'єкт `z3.RealVal`, який представляє дійсне число-константу. Значення вільних змінних (нейрони вхідного шару мережі) представляються через `z3.Real`.

Як приклад, на Рис.1 показана тривіальна нейромережа з 2 входами, одним прихованим рівнем та одним вихідним нейроном.

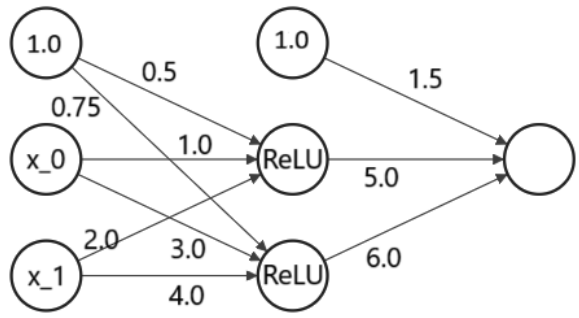

Рис. 1. Приклад графу нейромережі.

На Рис.2 наведено структурну ітерпретацію обчислювального графу вихідного нейрону такої мережі, виражену елементами Z3Py (для простоти сприйняття, обгортки «z3.RealVal» над дійсними константами пропускаються).

```
5*z3.If(
    z3.Real("x_0")*1 + z3.Real("x_1")*2 + 0.5 > 0,
    z3.Real("x_0")*1 + z3.Real("x_1")*2 + 0.5,
    0) +
6*z3.If(
    z3.Real("x_0")*3 + z3.Real("x_1")*4 + 0.75 > 0,
    z3.Real("x_0")*3 + z3.Real("x_1")*4 + 0.75,
    0) + 1.5
```

Рис. 2. Представлення вихідного нейрону в Z3Py.

В побудованому обчислювальному графі відсутня функція активації вихідного нейрону (яка у випадку нашої задачі - сигмоїда). Для представлення сигмоїди необхідна трансцендентна функція, а тому її не можна тотожно представити лише лінійн-

ною арифметикою та булевою алгеброю. Зазвичай, для представлення таких нелінійних функцій використовують апроксимацію лінійними сегментами. Однак у контексті нашої задачі ми пропонуємо скористатися тим, що сигмоїда є монотонно зростаючою на всій множині визначення, а тому порівняння її значення з очікуваним порогом класифікації (`threshold`) можна замінити на порівняння її аргументу (лінійної комбінації активацій нейронів передостаннього рівня) зі значенням оберненої функції від порогу (`inv_sigmoid(threshold)`). Таким чином ми уникаємо апроксимації і таке представлення мережі залишається математично тотожним оригінальному.

Зауважимо, що ідея "наївної" конверсії повнозв'язної нейронної мережі у вираз Z3 вже застосовувалась (як-от в Sapphire[9]). Таке представлення є надзвичайно потужним, адже дозволяє задавати довільні обмеження над вхідними змінними та вихідними значеннями в рамках теорій відомих Z3 (лінійна, нелінійна, булева арифметика і т.д.). Однак цей підхід поки не набув значного розвитку через неприйнятну часовитратність під час обробки SMT розв'язувачем такого обчислювального графу, що особливо помітно у процесі зростання кількості шарів в нейромережі. В наступному розділі запропоновано спосіб вирішення цієї задачі для випадку локальної верифікації мережі.

4. Алгоритм спрощення структури нейромережі

Одним зі шляхів до скорочення часу необхідного для доведення чи спростування властивостей нейронної мережі SMT-розв'язувачем є спрощення представлення її обчислювального графу, а саме виявлення та вилучення нелінійностей.

Розглянемо ситуацію, коли всі функції активації нейронів мережі є глобально нелінійними, але водночас кусково-лінійними (тобто лінійними на інтервалах). Для визначення рельєфу кордону ухвалення рішень класифікатора над усім вхідним простором необхідна повна композиція нелінійних активацій усіх нейронів мережі. Однак на відносно малому локальному проміжку вхідних значень кількість нелінійних функцій, необхідних для тотожного представлення нейронної мережі, значно скорочується. В такому випадку обчислювальний граф нейромережі можна спростити. Для кожного нейрону розглядається можливість спрощення через заміни його нелінійної функції активації на лінійну, тотожну на даному інтервалі оригінальній. Це стає можливим, якщо діапазон можливих значень функції активації лежить повністю в межах інтервалу її лінійності. Чим вужчий діапазон входів мережі, тим більше спрощень нейронних активацій на ньому можна провести. В той же час у випадку звуження вхідного діапазону до єдиної точки, обчислювальний граф вироджується в цільну лінійну функцію.

На рис. 3 продемонстрована ідея локальної апроксимації кусково-лінійної функції до нової, тотожної оригінальній лише на певному інтервалі. Як видно, під час звуження інтервалу, глобальна складність апроксимованої функції зменшується, адже зменшується кількість точок зламу («кутових точок»).

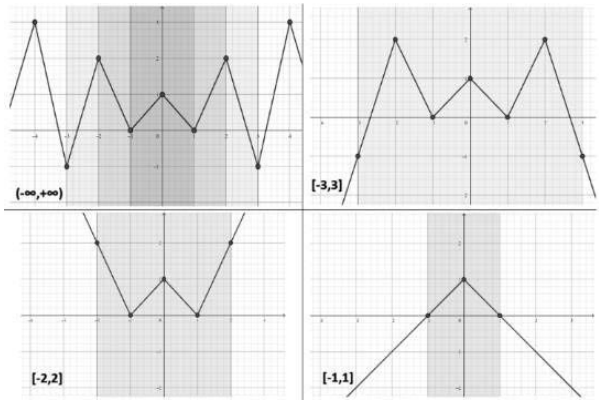

Рис. 3. Приклади локальних апроксимацій кусково-лінійної функції.

Для нейронних мереж властивість нелінійності забезпечується функціями активації нейронів - в нашому випадку це ReLU. Водночас, саме функції активації є найбільшим фактором складності у процесі аналізу графу нейромережі SMT-розв'язувачем, адже кількість можливих шляхів породжених умовними розгалуженнями («if-else», в які транслюється ReLU), у графі комбінаторно зростає разом зі зростанням глибини мережі.

ReLU складається лише з двох лінійних сегментів, які визначаються на проміжках (-inf, 0], та (0, +inf). Якщо на певній підмножині вхідних значень моделі діапазон значень, що подаються на вхід нейрона, лежить повністю в одному з цих двох проміжків, то функція активації ReLU для даного нейрону рівноцінна константі 0 (x = 0), або ж тотожній функції (x = x). Таким чином у разі заміни ReLU на одну з тривіальних альтернатив в побудованому обчислювальному графі нейронної мережі, задача доведення властивостей SMT-розв'язувачем значно спрощується, адже зменшується кількість розгалужень. Можливість такого спрощення згадується в [5], щоправда в контексті звуження можливих меж входів, використовуючи симплекс-метод.

У нашій роботі, пропонується альтернативний метод для знаходження можливостей спрощення функцій активації ReLU шляхом розв'язання локальної SMT-задачі для кожного окремого нейрону. Нехай необхідно провести верифікацію повнозв'язної нейромережі в околі точки *x* з вхідного набору даних. Як і раніше, окіл позначимо як *E(x,ε,F)*. Спрощення здійснюється порівнево в один прохід по графу мережі, починаючи з першого прихованого рівня. Для кожного рівня алгоритм виконує 2 фази.

Перша фаза – це спроба провести попередню «наївну» оптимізацію функцій активації нейронів без розв'язування SMT-задач. Для нейрона проводиться оцінка найширших теоретично можливих меж його вхідних значень, виходячи з обрахованих меж активацій нейронів попереднього рівня. Очевидно, що отримані межі не обов'язково є практично досяжними і зазвичай будуть значно ширші, ніж реально можливі. Якщо нижня межа вхідних значень нейрону > 0, то ReLU заміняється на тотожну функцію, а якщо верхня межа ≤ 0, то ReLU заміняється на константу 0. Водночас межі для вхідного рівня визначаються безпосередньо обраним околом *E(x,ε,F)*. Якщо для нейрону на першій фазі спрощення не відбулося, то для нього буде застосована друга фаза.

В **другій фазі**, SMT-роз'язувач проводить повний аналіз обчислювального графу кожного з неспрощених нейронів даного рівня над околом *E(x,ε,F)*. Нам уже відоме значення, що подається на вхід функції активації нейрона для вхідної точки *x*. Позначимо це значення як $n(x)$. Задача перевірки можливості спрощення ReLU зводиться до доведення відсутності такої точки $x' \in E(x,\varepsilon,F)$ для якої значення, що подається на вхід даного нейрону $n(x')$ лежатиме по протилежний бік від 0 відносно $n(x)$. Це можна виразити у формі обмеження:

$$C_{neuron}: \begin{cases} n(x') > 0, & якщо\ n(x) < 0 \\ n(x') \le 0, & якщо\ n(x) \ge 0 \end{cases} \quad (7)$$

Якщо SMT-роз'язувач доводить невиконуваність даного обмеження, то діапазон значень $n(x')$ повністю лежить в одному з інтервалів лінійності ReLU для $\forall x' \in E(x,\ \varepsilon,\ F)$, що дозволяє провести спрощення. ReLU замінюється на тотожну функцію, якщо $n(x) > 0$, або ж на константу 0, якщо $n(x) \le 0$. В іншому випадку спрощення для даного нейрона не відбувається.

Зауважимо, що ідея такого спрощення є евристичною. Очікується, що витрати на інкрементальне спрощення шляхом понейронного ров'язування SMT-задач на вже спрощених підграфах є менш часовитратними, ніж одноразовий аналіз оригінального графу нейромережі. Крім того, очевидно, що зі збільшенням вхідного околу зменшується кількість можливостей спрощення функцій активації, і тому ефективність такого підходу буде падати. В найгіршому випадку час виконання буде більшим, ніж під час аналізу неоптимізованого графу мережі. Незважаючи на це, в контексті задачі верифікації моделі класифікації мережевих потоків саме така оптимізація дозволила проаналізувати повну множину точок інтересу за прийнятний час, про що йдеться в наступному розділі.

5. Експериментальні результати

Ми розглядали модель класифікації мережевих потоків на предмет виявлення активності ботнетів, які є джерелом атак на інтернет ресурси та WEB-застосунки.

Модель представлена нейронною мережею, навченою на мережевих потоках

виділених з набору даних ISCX Botnet 2014 [10]. Кожен мережевий потік представлений набором з 50 ознак, отриманих за допомогою інструменту CIC Flow Meter V4.0 [11]. Поріг класифікації встановлено в 0.51 – якщо значення на вихідному нейроні мережі перевищує цей поріг, потік вважається «шкідливим», тобто таким, що містить активність ботнетів. Після цього з навчальної вибірки було виділено всі потоки, які модель класифікувала вірно, але вони лежать близько до межі ухвалення рішень (тобто для них прогнозована впевненість > 0.51, але < 0.55) – всього 836 потоків (тобто 0.76% від повної вибірки).

Верифікація у відносному околі точок здійснювалась над семантично зв'язаною групою із 9 ознак, а саме, «загальна тривалість» та статистики «міжпакетних інтервалів» потоків (інші ознаки залишались фіксованими):

```
"Flow Duration", "Fwd IAT Total", "Fwd IAT Std", "Fwd IAT Max", "Fwd IAT Min", "Bwd IAT Total", "Bwd IAT Std", "Bwd IAT Max", "Bwd IAT Min"
```

Водночас накладались додаткові обмеження правдоподібності:

```
"Fwd IAT Max" >= "Fwd IAT Min",
"Bwd IAT Max" >= "Bwd IAT Min"
```

Верифікація проводилась для відносних околів з параметром ε: 0.05, 0.1 та 0.15.

Таблиця 1

Результати верифікації класифікатора мережевих потоків

ε	# verified	# failed	Time	Avg. final ReLU count
0.05	763	73	6m 32s	0.23
0.1	722	114	15m 8s	0.48
0.15	711	125	92m 22s	0.83

У Табл. 1, колонка "# verified" показує кількість мережевих потоків, для яких доведено, що завжди зберігається клас під час довільного варіювання міжпакетних інтервалів у зазначених межах. Колонка "# failed" відображає кількість випадків, для яких існує така варіація, за якої передбачуваний моделлю клас не співпадає з оригінальним. Як і очікувалось, у процесі зростання ε, також зростає і кількість можливостей помилки класифікації. В останній колонці наведено середня кількість функцій ReLU, які лишились після спрощення обчислювального графу мережі. Як видно, для більшості випадків достатньо щонайбільше однієї ReLU для математично-тотожного представлення моделі на даних вхідних діапазонах.

Висновки

В даній роботі було розроблено метод для доказової перевірки надійності нейромереж-класифікаторів. Було показано практичне застосування даного методу для верифікації системи аналізу мережевого трафіку на предмет активності ботненів. Це є важливим результатом, адже застосування ШІ в галузі кібербезпеки суттєво обмежене обсягом наявних наборів даних для їх емпіричного тестування. Крім того, даний метод дає можливість гарантувати локальну стійкість класифікаторів до змагальних атак, тобто навмисних спроб ухиляння хакерів від виявлення їх дій. Було показано, що алгоритм здатен як довести коректність роботи моделі класифікації на певних регіонах вхідного простору даних, так і виявити проблемні зони, де модель може припуститися помилки.

Крім того, отримані експериментальні дані підтверджують ефективність запропонованої оптимізації обчислювального графу мережі через порівневе спрощення функцій активації шляхом рішення локальних SMT-задач.

Описаний підхід верифікації нейромереж може бути застосований не лише в задачах кібербезпеки, а й у будь-яких системах класифікації, де точність є критичною, зокрема, в розпізнаванні образів або мови.

Одним із напрямків подальших досліджень, є використання запропонованого методу верифікації для мереж із довільними функціями активації. Наприклад, через їх апроксимацію лінійними сегментами.

Крім того запропонований спосіб інкрементальної оптимізації графу мережі також має значний потенціал для подальшого розвитку. Зокрема, саму процедуру спрощення обчислювального графу можливо значно прискорити, якщо послідовне рішення понейронних SMT-задач в межах одного рівня замінити на паралельне. Такі задачі не потребують між-потокової синхронізації. Також при вилученні нелінійностей можливо брати до уваги межі можливих вхідних значень не якогось окремого нейрону, а комбінації меж кількох нейронів даного рівня.

Нарешті, самим важливим напрямком подальших досліджень у галузі кібербезпеки є створення методів по півищенню надійності та стійкості моделей виявлення мережевих загроз, як до змагальних атак, так і до випадкових флуктуацій в мережевих даних.

Література

1. Casadio, M. et al. (2022). Neural Network Robustness as a Verification Property: A Principled Case Study. In: Shoham, S., Vizel, Y. (eds) Computer Aided Verification. CAV 2022. Lecture Notes in Computer Science, vol 13371. Springer, Cham. https://doi.org/10.1007/978-3-031-13185-1_11
2. Odena, A., Olsson, C., Andersen, D., Goodfellow, I.. (2019). TensorFuzz: Debugging Neural Networks with Coverage-Guided Fuzzing. Proceedings of the 36th International Conference on Machine Learning, in Proceedings of Machine Learning Research 97:4901-4911. https://doi.org/10.48550/arXiv.1807.10875
3. Z. Yang, J. Shi, M. Asyrofi and D. Lo, "Revisiting Neuron Coverage Metrics and Quality of Deep Neural Networks," in 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 2022 pp. 408-419. 10.1109/SANER53432.2022.00056
4. Goodfellow, Ian & Shlens, Jonathon & Szegedy, Christian. (2014). Explaining and Harnessing Adversarial Examples. https://doi.org/10.48550/arXiv.1412.6572
5. Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J. (2017). Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In: Majumdar, R., Kunčak, V. (eds) Computer Aided Verification. CAV 2017. Lecture Notes in Computer Science(), vol 10426. Springer, Cham. https://doi.org/10.1007/978-3-319-63387-9_5
6. Katz, G. et al. (2019). The Marabou Framework for Verification and Analysis of Deep Neural Networks. In: Dillig, I., Tasiran, S. (eds) Computer Aided Verification. CAV 2019. Lecture Notes in Computer Science(), vol 11561. Springer, Cham. https://doi.org/10.1007/978-3-030-25540-4_26
7. Elboher, Y.Y., Gottschlich, J., Katz, G. (2020). An Abstraction-Based Framework for Neural Network Verification. In: Lahiri, S., Wang, C. (eds) Computer Aided Verification. CAV 2020. Lecture Notes in Computer Science(), vol 12224. Springer, Cham. https://doi.org/10.1007/978-3-030-53288-8_3
8. de Moura, L., Bjørner, N. (2008). Z3: An Efficient SMT Solver. In: Ramakrishnan, C.R., Rehof, J. (eds) Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2008. Lecture Notes in Computer Science, vol 4963. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-78800-3_24
9. Sapphire. Accessed: 01.04.2024. https://github.com/wenkokke/sapphire
10. Beigi, E.B., Jazi, H.H., Stakhanova, N., Ghorbani, A.A.: Towards effective feature selection in machine learning-based botnet detection approaches. In: 2014 IEEE Conference on Communications and Network Security. pp. 247–255 (2014)

https://dx.doi.org/10.1109/CNS.2014.6997492
11. CICFlowMeter (formerly ISCXFlowMeter). Accessed: 01.04.2024. https://www.unb.ca/cic/research/applications.html

Одержано: 10.04.2024
Внутрішня рецензія отримана: 18.04.2024
Зовнішня рецензія отримана: 25.04.2024

Про авторів:

[1]*Панчук Богдан Олександрович*,
аспірант.
https://orcid.org/0000-0002-5389-359X.

Місце роботи авторів:

[1] Інститут кібернетики ім. В.М. Глушкова НАН України,
Тел. (+38) (044) 526-20-08
E-mail: incyb@incyb.kiev.ua,
Сайт: www.incyb.kiev.ua