Міністерство освіти і науки України

Департамент науки і освіти Харківської обласної державної адміністрації

Харківське територіальне відділення МАН України

Відділення: комп'ютерних наук

Секція: безпека інформаційних та телекомунікаційних систем

РОЗРОБКА СИСТЕМИ ІДЕНТИФІКАЦІЇ СТАНУ КОМП’ЮТЕРНОЇ СИСТЕМИ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ

Роботу виконав:

Кречко Богдан Станіславович,

учень 11 класу Харківського

Навчально-виховного комплексу №45 «Академічна гімназія» Харківської міської ради Харківської області

Науковий керівник  
Семенов Сергій Геннадійович

завідувач кафедри обчислювальної техніки та програмування НТУ “ХПІ”, доктор технінчних наук, професор

Харків - 2018

**Тези**

Розробка системи ідентифікації стану комп’ютерної системи на основі нейронної мережі

Автор роботи: Кречко Богдан Станіславович.  
Харківське територіальне відділення МАН України; Харківський навчально-виховний комплекс №45 «Академічна гімназія» Харківської міської ради Харківської області; 11 клас; м. Харків;

Науковий керівник:   
Семенов Сергій Геннадійович, завідувач кафедри обчислювальної техніки та програмування НТУ «ХПІ», доктор технічних наук, професор

У сучасних реаліях, коли збільшується кількість вірусів і зростає показник кіберзлочинності стає актуальним питання комп'ютерного захисту. Комп'ютерні віруси перешкоджають роботі ОС. Багато з них розроблені з метою створювати незручності при роботі з ПК і приводять її в неробочий стан.

Антивірус має захищати від всіх видів шкідливих програм, і чим краще він це робить, тим спокійніше живе його користувач. На жаль, далеко не всі антивірусні продукти, забезпечують задовільний рівень захисту. В цьому і полягає основна проблема антивірусних програм на сьогоднішній день.

Метою роботи є створення евристичного аналізатора, який обробляє вхідні дані, що представляють собою сигнатури шкідливих файлів.

Для досягнення цієї мети потрібно: проаналізувати та дослідити можливості середовища програмування Java, адже він гарно пристосований до роботи з штучними мережами.

Робота містить опис процесу роботи нейронної мережі, розгляд вхідних та вихідних даних, вибір середовища програмування й опис застосованих методів роботи мережі, а також створення алгоритмів роботи основних методів і відповідного програмного забезпечення.

У результаті був написаний евристичний аналізатор на основі нейронної мережі АРТ-1, що дозволяє вдосконалити евристичний аналізатор антивірусної програми.

## ЗМІСТ

**ВСТУП 4**

**РОЗДІЛ 1   
ШТУЧНІ НЕЙРОННІ МЕРЕЖІ 7**

1.1. Загальні положення про нейронні мережі 7

1.2. Навчання нейронних мереж 9

1.3. Види нейронних мереж, що дозволяють вирішити проблему класифікації 13

1.4. Аналізатор стану системи на основі нейронної мережі АРТ-1 14

1.4.1 Нейронна мережа АРТ-1 і її базова архітектура 15

1.5. Вивчення аналогів 18

**РОЗДІЛ 2   
РОЗРОБКА СИСТЕМИ ІДЕНТИФІКАЦІЇ СТАНУ КОМП‘ЮТЕРНОЇ СИСТЕМИ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ 21**

2.1 Алгоритм та структура програми 21

2.2 Структура вхідних даних 23

2.3 Робота евристичного аналізатора 24

2.4 Аналіз вихідних даних системи ідентифікації 27

**ВИСНОВОК 31**

**ПЕРЕЧЕНЬ ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 32**

**ВСТУП**

Існує кілька основних методів пошуку вірусів, які застосовуються антивірусними програмами: сканування; евристичний аналіз; виявлення змін; резидентні монітори. Антивіруси можуть реалізовувати всі перераховані вище методики, або тільки деякі з них.

Антивірусне програмне забезпечення (АПО), влаштовано так, що аналізує програмний код і дані шляхом сканування файлів і потоків даних, що приймаються в комп'ютер по мережі, потім порівнює ці дані з сигнатурами з баз даних сигнатур відомих вірусів.

АПО на додаток до можливості простого сканування і порівнювання з сигнатурами, даних і коду, з вірусами відомих штамів, часто доповнюється евристичними аналізаторами. Це найдієвіша зброя з відомих засобів для протидії ще невідомим вірусам, але те, як функціонують реальні евристичні аналізатори, піддається критиці.

Евристичні аналізатори були задумані так, щоб «парсити» (від англ. Parse - розбирати) код, тобто виробляти його розбирання на елементарні фрагменти, з яких можуть генеруватися послідовності потенційно небезпечних комбінацій.

Причина неефективності евристичних аналізаторів криється в тому, що, як виявляється, випадків визначення коду як потенційно небезпечного, виникає набагато більше, ніж можна було припустити. Відбувається це тому, що файли даних можуть не обов'язково містити виключно тільки виконуваний код. Але і дані, як картинки, відео та іншу мультимедійну інформацію.

Помилкове спрацьовування евристичного аналізу буває настільки часто, що його використовувати стає важко. І це перша проблема. А друга проблема полягає в тому, що всі варіанти комбінацій потенційно небезпечного коду практично неможливо прорахувати, поєднань такого безліч. В результаті виходить, що першу проблему можна тільки погіршити, нарощуючи варіанти новими комбінаціями потенційно небезпечних поєднань, які можуть зустрічатися в абсолютно будь-яких файлах, крім самого програмного коду.

Практично всі сучасні антивірусні засоби застосовують технологію евристичного аналізу програмного коду(Antivir, Avast, AVG, Avira, Panda, NOD32). У такому випадку, як правило, потрібне додаткове оновлення антивірусних баз для отримання останніх сигнатур і алгоритмів лікування, які, можливо, містять інформацію про раніше невідомий вірус. В іншому випадку, файл передається для дослідження антивірусним аналітикам або авторам антивірусних програм.

Інтелектуальні системи на основі штучних нейронних мереж дозволяють з успіхом вирішувати проблеми розпізнавання образів, виконання прогнозів, оптимізації, асоціативної пам'яті і керування. Традиційні підходи до рішення цих проблем не завжди надають необхідної гнучкість і багато застосувань виграють від використання нейромереж.

Штучні нейромережі є електронними моделями нейронної структури мозку, який, головним чином, навчається з досвіду. Природній аналог доводить, що множина проблем, які поки що не підвладні розв'язуванню наявними комп'ютерами, можуть бути ефективно вирішені блоками нейромереж.

Але навчання нейронної мережі є складним та багато ітераційним процесом. При цьому швидкі зміни у технологіях доступу до комп’ютерних ситсем зазвичай призводить до необхідності повного перенавчання цих мереж. Цю проблему можна вирішити використовуя нейронну мережу на основі АРТ-1. Мережі й алгоритми APT зберігають пластичність, необхідну для вивчення нових образів, у той же час, запобігаючи зміні раніше запам’ятованих образів, і тому вони можуть бути ефективними при розпізнаванні комп’ютерних вирусів.

Для навчання нейронної мережі APT пропонується метод, за яким особа, що керує мережею, бере особисту участь у навчанні мережі, вона сама задає еталонні значення для всіх типів вірусів.

Раніше цей метод застосовувався до імовірнісних нейронних мереж, але використання цього методу для мереж багатошарових нейронів значно підвищить його продуктивність із розпізнавання небезпечного програмного забеспечення.

Об'єктом дослідження є евристичні аналізатори антивірусних програм.

Предметом дослідження є метод евристичного аналізу комп’ютерних вирусів з використанням нейронної мережу АРТ-1.

Було вирішено розробити евристичний аналізатор на основі нейронних мереж, а саме АРТ-1.

**РОЗДІЛ 1** **ШТУЧНІ НЕЙРОННІ МЕРЕЖІ**

**1.1 Загальні положення про нейронні мережі**

Штучні нейронні мережі (ШНМ) − математичні моделі, а також їхня програмна та апаратна реалізація, побудовані за принципом функціонування біологічних нейронних мереж − мереж нервових клітин живого організму. Системи, архітектура і принцип дії базується на аналогії з мозком живих істот. Ключовим елементом цих систем виступає штучний нейрон як імітаційна модель нервової клітини мозку − біологічного нейрона. Як наслідок, після розробки алгоритмів навчання, отримані моделі стали використовуватися в практичних цілях: в задачах прогнозування, для розпізнавання образів, в задачах керування та інших. ШНМ являють собою систему з'єднаних між собою простих процесорів (штучних нейронів), які взаємодіють. Кожен процесор схожої мережі має справу тільки з сигналами, які він періодично отримує, і сигналами, які він періодично посилає іншим процесорам. І тим не менш, будучи з'єднаними в досить велику мережу з керованою взаємодією, такі локально прості процесори разом здатні виконувати достатньо складні завдання.

З точки зору машинного навчання, нейронна мережа є окремим випадком методів розпізнавання образів, дискримінантного аналізу, методів кластеризації тощо. З математичної точки зору, навчання нейронних мереж − це багатопараметрична задача нелінійної оптимізації. А з точки зору штучного інтелекту, ШНМ є основою філософської течії коннективізму і основним напрямком в структурному підході з вивчення можливості побудови (моделювання) природного інтелекту за допомогою комп'ютерних алгоритмів. [1 с. 363—384.]

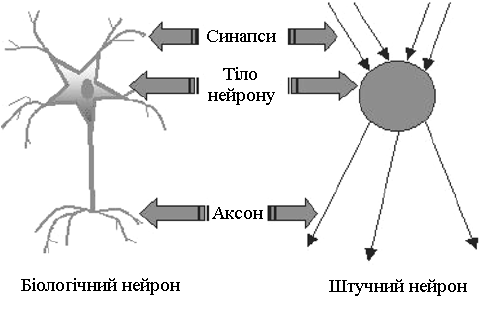


Рис. 1.1 Біологічний та штучний нейрони

Кожен нейрон характеризується своїм поточним станом за аналогією з нервовими клітинами головного мозку, які можуть бути порушені або загальмовані. Він володіє групою синапсів − односпрямованих вхідних зв'язків, з'єднаних з виходами інших нейронів, а також має аксон − вихідну зв'язок даного нейрона, з якої сигнал (збудження або гальмування) надходить на синапси наступних нейронів. Загальний вигляд нейрона наведено на рис. Кожен синапс характеризується величиною синаптичного зв'язку або її вагою .

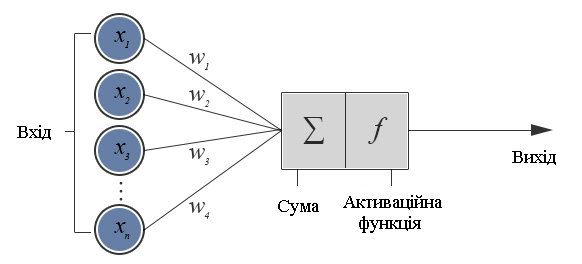


Рис. 1.2 Принцип роботи штучного нейрона

|  |  |  |
| --- | --- | --- |
|  |  |  |

Однією з найбільш поширених функцій є нелінійна функція з насиченням, так звана логістична функція або сигмоїда (тобто функція S-подібного виду (див формулу 1.1)): 

 (Формула 1.1)

При зменшенні *a* сигмоїда стає більш пологою, в межі при вироджується в горизонтальну лінію на рівні 0.5, при збільшенні a сигмоїда наближається за зовнішнім виглядом до функції одиничного стрибка з порогом в точці = 0. З виразу для сигмоїда очевидно, що вихідне значення нейрона лежить в діапазоні [0,1]. Одне з цінних властивостей сигмоїдной функції − простий вираз для її похідної (див формулу 1.2):



|  |  |  |
| --- | --- | --- |
|  | (Формула 1.2) |  |

Повертаючись до загальних рис, притаманних всім нейронним мережам, відзначимо принцип паралельної обробки сигналів, який досягається шляхом об'єднання великого числа нейронів в так звані шари і з'єднання певним чином нейронів різних шарів.

**1.2 Навчання нейронних мереж**

Нейронні мережі не програмуються в звичайному розумінні цього слова, вони навчаються. Здатність до навчання є фундаментальною властивістю мозку. Процес навчання може розглядатися як визначення архітектури мережі і налаштування ваг зв'язків для ефективного виконання спеціальної задачі. Нейромережа налаштовує ваги зв'язків по наявній навчальній множині. Властивість мереж навчатися на прикладах робить їх більш привабливими в порівнянні із системами, які функціонують згідно визначеній системі правил, сформульованої експертами.[2 c.480]

Для процесу навчання необхідно мати модель зовнішнього середовища, у якій функціонує нейронна мережа та інформацію для навчання. По-друге, необхідно визначити, як модифікувати вагові параметри мережі. Алгоритм навчання означає процедуру, в якій використовуються правила навчання для налаштування ваг.

Можливість навчання − одна з головних переваг нейронних мереж перед традиційними алгоритмами. Технічно навчання полягає в знаходженні коефіцієнтів зв'язків між нейронами. У процесі навчання нейронна мережа здатна виявляти складні залежності між вхідними даними і вихідними, а також виконувати узагальнення. Це означає, що у разі успішного навчання мережа зможе повернути правильний результат на підставі даних, які були відсутні в навчальній вибірці, а також неповних та / або «зашумлених», частково перекручених даних. Таким чином, нейронні мережі є надзвичайно складними.

Для навчання нейромереж в багатьох випадках використовують алгоритм зворотного розповсюдження помилки. Розв’язок задачі за допомогою нейронної мережі зводиться до наступних етапів:

* вибрати відповідну модель мережі (наприклад, трьохкрокову);
* визначити топологію мережі (кількість елементів та їх зв’язки);
* вказати спосіб навчання (наприклад, зі зворотним розповсюдженням помилок) і параметри навчання.

Хоч один нейрон і здатний виконувати найпростіші процедури розпізнавання, сила нейронних обчислень виникає від з'єднань нейронів в мережах. Найпростіша мережа складається з групи нейронів, що створюють прошарок (див Рис 1.3).

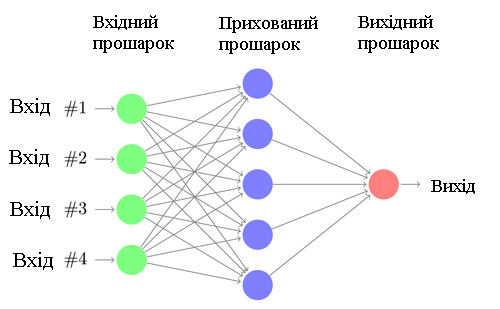


Рис. 1.3 Структура нейронної мережі

Зазначимо, що вершини-кола зліва служать лише для розподілу вхідних сигналів. Вони не виконують ніяких обчислень, і тому не будуть вважатися прошарком. Кожний елемент з множини входів окремою вагою сполучений з кожним штучним нейроном. А кожний нейрон видає зважену суму входів в мережу.

Більш великі і складні нейронні мережі мають, як правило, і великі обчислювальні можливості. Хоч створені мережі всіх конфігурацій, які тільки можна собі представити, пошарова організація нейронів копіює прошаркові структури певних відділів мозку. Виявилося, що такі багатошарові мережі володіють більшими можливостями, ніж одношарові і в останні роки були розроблені алгоритми для їх навчання.

Багатошарові мережі можуть утворюватися каскадами прошарків. Вихід одного прошарку є входом для подальшого прошарку. Подібна мережа показана на рис 1.4.

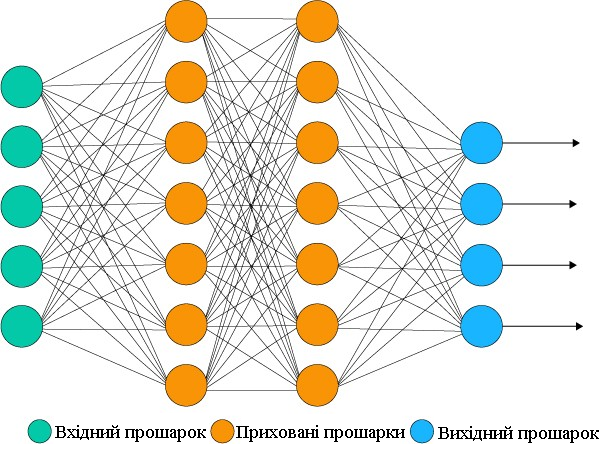


Рис. 1.4 Структура багатошарової нейронної мережі

Обчислення виходу прошарку полягає в множенні вхідного вектора на першу вагову матрицю з подальшим множенням (якщо відсутня нелінійна активаційна функція) результуючого вектора на другу вагову матрицю.

|  |  |  |
| --- | --- | --- |
|  | , | (1.6) |

де *х* – вагова матриця, *w1*– ваги зв’язків.

Оскільки множення матриць асоціативне, то

|  |  |  |
| --- | --- | --- |
|  | . | (1.7) |

Це показує, що двошарова лінійна мережа еквівалентна одному прошарку з ваговою матрицею, рівною виробленню двох вагових матриць. Таким чином, для розширення можливостей мереж в порівнянні з одношаровою мережею необхідна нелінійна активаційна функція.

У мереж, розглянутих досі, не було зворотних зв'язків, тобто з'єднань, що йдуть від виходів деякого прошарку до входів цього ж прошарку або попередніх шарів. Цей спеціальний клас мереж, який називається мережами без зворотних зв'язків або мережами прямого поширення. Мережі більш загального вигляду, що мають з'єднання від виходів до входів, називаються мережами із зворотними зв'язками. [3 c.20]

В мережах без зворотних зв'язків немає пам'яті, їх вихід повністю визначається поточними входами і значеннями ваг. В деяких конфігураціях мереж із зворотними зв'язками попередні значення виходів повертаються на входи; вихід, отже, визначається як поточним входом, так і попередніми виходами. З цієї причини мережам із зворотними зв'язками притаманні властивості, схожі з короткочасною людською пам'яттю.

**1.3 Види нейронних мереж, що дозволяють вирішити проблему класифікації.**

Існує декілька типів нейронних мереж, що дозволяють швидко вирішити проблему класифікації:

Мережі прямого поширення - всі зв'язки цієї мережі мають суворе напрямок від вхідних нейронів до їх виходів. Серед таких мереж хочеться відзначити: найпростіший персептрон автором якого є Френк Розенблат і багатошаровий персептрон. Нейронні мережі реккурентное типу - дані з вихідних нейронів або з прихованого шару передається частково назад на вхідні нейрони.

Радіально базисні функції - це нейронна мережа, в основі якої є наявність прихованого шару з радіальних елементів і вихідного шару з лінійних елементів. Такі мережі досить компактні і навчаються досить швидко. Радіально базисна мережа користується наступними унікальними властивостями: один прихований шар, нейрони тільки прихованого шару мають нелінійну функцію активації і синаптичні ваги прихованого і вхідного шарів є одиницею.

Мережі Кохонена - це клас мереж зазвичай навчається без допомоги вчителя і часто застосовується в задачах пов'язаних з розпізнаванням зображень. Такі мережі здатні визначати нові елементи у вхідних даних: якщо пройшовши навчання мережу побачить набір даних, несхожий ні на один зі знайомих зразків, то вона класифікує такий набір і не виявить його новизну. Мережа Кохонена має всього два шари: вихідний і вхідний, складений з радіальних елементів.

Але навчання нейронної мережі є складним та багато ітераційним процесом. При цьому швидкі зміни у технологіях доступу до комп’ютерних ситсем зазвичай призводить до необхідності повного перенавчання цих мереж. Цю проблему можна вирішити використовуя нейронну мережу на основі АРТ-1. Мережі й алгоритми APT зберігають пластичність, необхідну для вивчення нових образів, у той же час, запобігаючи зміні раніше запам’ятованих образів, і тому вони можуть бути ефективними при розпізнаванні комп’ютерних вирусів.

## 1.4 Аналізатор стану системи на основі нейронної мережі АРТ-1

За основу евристичного аналізатора було взято класичну архітектуру нейронної мережі АРТ-1. Шар сприймає пред’явлені зображення і передає отриману інформацію нейронам інтерфейсного шару та управляючому нейрону.

Шар *Y* являється шаром конкуруючих або змагаючихся нейронів. В будь-який час кожен елемент *Y* розпізнавального шару знаходиться в одному з трьох станів:

– активний (*UвихYj* = *d*; *d* = 1);

– неактивний (*UвихYj* = 0, але здатний приймати участь у змаганні);

* загальмований (*UвихYj* = -1, й не допущений до змагань при пред'явленні поточного вхідного вектору).

Після пред’явлення вхідного зображення активним залишається лише один розпізнавальний нейрон, усі останні *Y*-елементи мають нульові або негативні вхідні сигнали. Виділений розпізнавальний нейрон допускається до навчання вхідним зображенням тільки в тому випадку, якщо його ваговий вектор зв’язків з шару *Y* до шару *Z* подібний до вхідного вектору. Це рішення приймається за допомогою *R*-нейрона на основі спеціального параметра, який отримав назву параметр подібності, й сигналів, які поступають з вхідного і інтерфейсного шару елементів. Через допоміжні елементи виконується або навчання виділеного розпізнавального *Y*-елементу, або його загальмування з подальшим виключенням з числа змагаючихся при повторних пред’явленнях того ж самого вхідного зображення, коли виділяються нові кандидати для навчання вхідним зображенням.[4 c.316]

В АРТ розрізняють два типи навчання: швидке і повільне. Вони відрізняються як теоретичними передумовами, так і своїми робочими характеристиками. У методі швидкого навчання ваги протягом часу тривалого резонансу досягають рівноважного стану при кожному пред'явленні зображень. Ця форма навчання типова для довічних нейронних мереж АРТ-1. У методі повільного навчання має місце істотно більш повільна зміна ваг під час резонансу, і вони не досягають рівноважних значень при кожному пред'явленні вхідних зображень.

**1.4.1 Нейронна мережа АРТ-1 і її базова архітектура**

Нейронні мережі АРТ-1 розраховані на роботу з бінарними вхідними зображеннями або векторами. Їх загальна архітектура наведена на рис і відрізняється від базової архітектури мереж АРТ наявністю додаткових елементів, які забезпечують управління процесом навчання. Всі зв'язки мережі АРТ-1 передають тільки бінарні сигнали 0 або 1.

Кожен елемент в інтерфейсному або *Y*-шарі мережі АРТ-1 має три джерела вхідних сигналів. Довільний інтерфейсний елемент *Z* може отримувати сигнали від елемента *S* вхідного шару, з вершин *Y*-шару і від нейрона *G1*. Аналогічно, елемент *Yj* може отримувати сигнали від інтерфейсних елементів, нейронів *R* і *G2*. Для переводу нейронів інтерфейсного або розпізнавального шарів в активний одиничний стан необхідна наявність вхідних збуджуючих сигналів з двох джерел. Оскільки кожний з розглянутих нейронів має три можливих джерела сигналів, то умова збудження цих нейронів отримало назву «Правила два з трьох». (Рис 1.4)

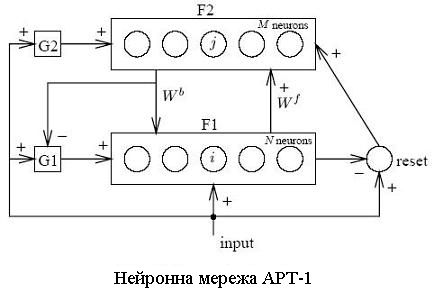


Рис. 1.4 Структура нейронної мережі АРТ-1

У початковому стані нейрони *R*, *G1*, *G2* і вхідного шару *Fa* мають нульові вихідні сигнали (перебувають у стані «0»). При подачі на входи *S*-елементів бінарних компонент пред’являємого зображення частина з них, які отримали одиничні вхідні сигнали, переходить у стан «1». Збудливі сигнали з виходів цих нейронів переводять у стан «1» нейрони *G1*, *G2* і *R*, а також надходять на входи відповідних нейронів інтерфейсного шару. Нейрони інтерфейсного шару, що отримали одиничні сигнали від нейронів вхідного шару і елемента, за правилом два з трьох переходять в активний стан і посилають свої збуджуючі сигнали зі зв’язків з вагами на входи нейронів розпізнавального шару. Нейрони розпізнавального шару переходять в активний стан також за правилом два з трьох, отримуючи збуджуючі сигнали не тільки від елементів інтерфейсного шару, але і від елемента. Після цього в шарі *Y*-нейронів відбувається латеральний процес і виділяється єдиний *Y*-нейрон. Всі нейрони *Y*-шару, крім переможного, переходять у стан «0», а нейрон переможець – у стан «1». Одиничний сигнал нейрона-переможця загальмовує керуючий нейрон, а також надходить по зв’язках з вагами на входи елементів інтерфейсного шару. Оскільки елементи інтерфейсного шару підкоряються правилу два з трьох, то у відсутності збуджуючих сигналів від нейрона, в активному стані залишаться тільки ті інтерфейсні елементи, які отримують поодинокі сигнали і від елемента вхідного шару, і від нейрона-переможця розпізнавального шару.

Гальмуючі сигнали активних елементів інтерфейсного шару надходять на входи *R*-елемента, який також отримує і збуджуючі сигнали від нейронів вхідного шару. Залежно від співвідношення величин збуджуючих і гальмуючих сигналів *R*-елемент переходить або в стан «0», або в стан «1». При нульовому вихідному сигналі *R*-елемента в мережі наступає резонанс і відбувається навчання ваг зв’язків *Y*-нейрона-переможця, а при одиничному вихідному сигналі – *Y*-нейрон-переможець загальмовується і позбавляється можливості брати участь у змаганнях при повторних пред’явлення поточного зображення. Потім в *Y*-шарі проводиться вибір нового нейрона-переможця.   
 Алгоритм роботи нейронної мережі наведено на рис 1.5

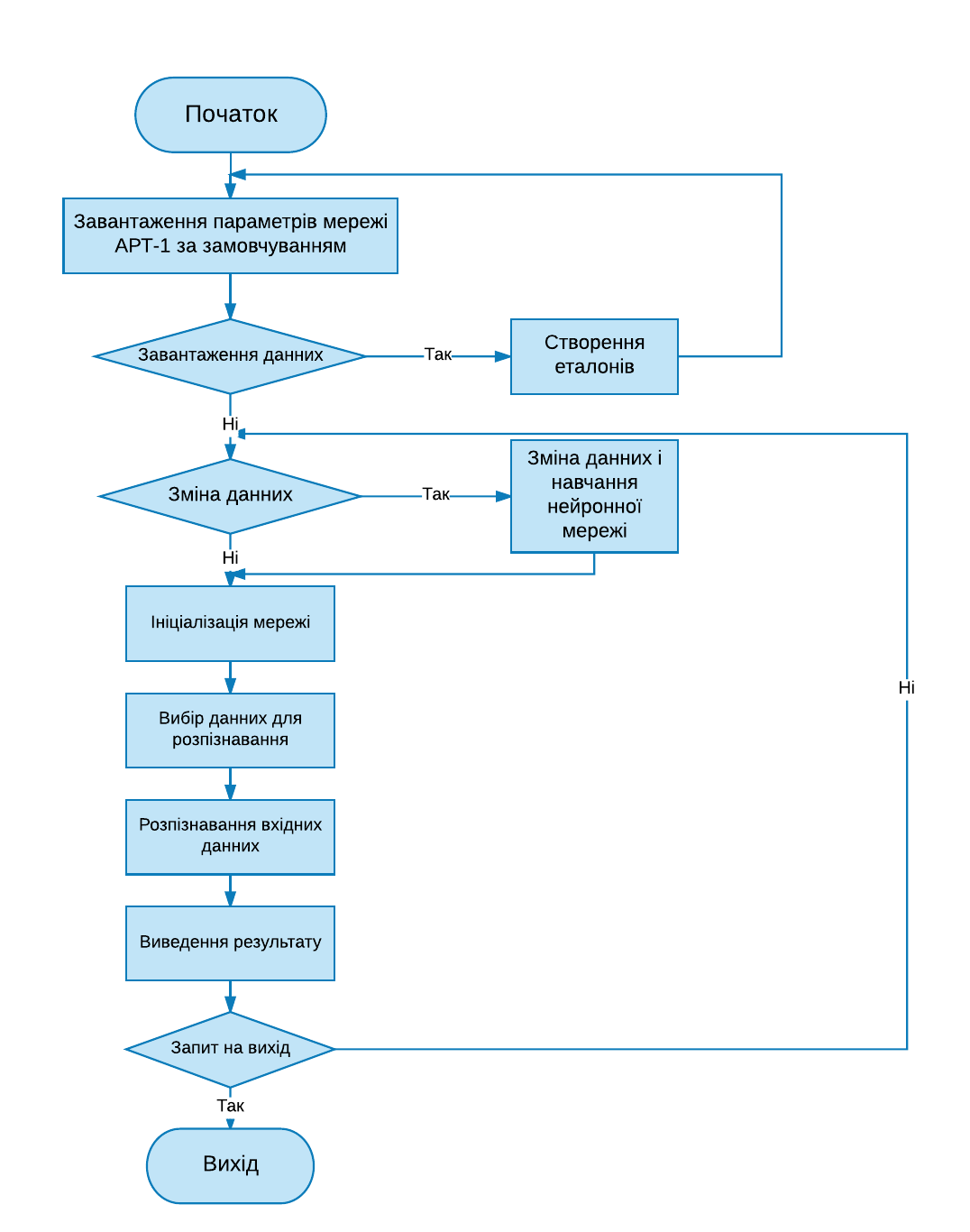


Рис 1.5 Схема алгоритму роботи нейронної мережі АРТ-1 для розпізнавання комп’ютерних вірусів

**1.5 Вивчення аналогів**

Для різних шкідливих програм характерна множина подій, і їх кількість стрімко підростає. Загальну картину можна побачити, наприклад, на рис 1.6 на якому показано сукупності подій, на які реагує «Антивірус Касперьского».

Деякі шкідливі програми також намагаються змінити або додати якісь інші ключі реєстру (наприклад, прописати свою DLL, змінити параметр Internet Explorer ’a або встановленої у системі антивірусної програми і т. д.).

Найпоширеніші способи контролю змін в реєстрі — це перехват API-функцій роботи з реєстром або використання спеціалізованих функцій контролю над реєстром. Що стосується першого способу, то для контролю над реєстром зазвичай достатньо перехопити функції RegOpenKey, RegCreateKey, RegDeleteKey з advapi32.dll або більш низько рівневі NtOpenKey, NtCreateKey, NtDeleteKey і т. п.

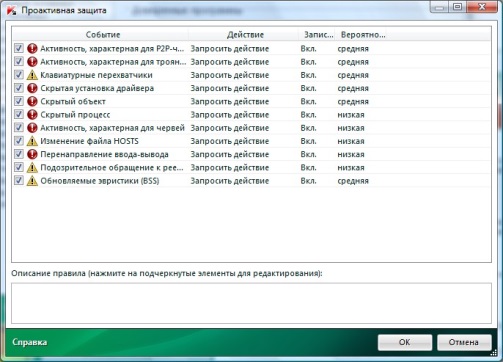


Рис 1.6 - Події, які контролюються про активним захистом «Антивіруса Касперського»

На рис 1.7 показано, як антивірус від McAfee перехоплює функції ядра для роботи з реєстром.

Наступний спосіб полягає у використанні спеціальної API-функції RegNotifyChangeKeyValue, вона оповіщає програму, яка її викликала, про зміни певного ключа реєстру. Цю API-функцію використовують багато антивірусних програм (рис 1.7), вище красної пунктирної лінії показана реалізація в McAfee, нижче лінії — в «Касперському».

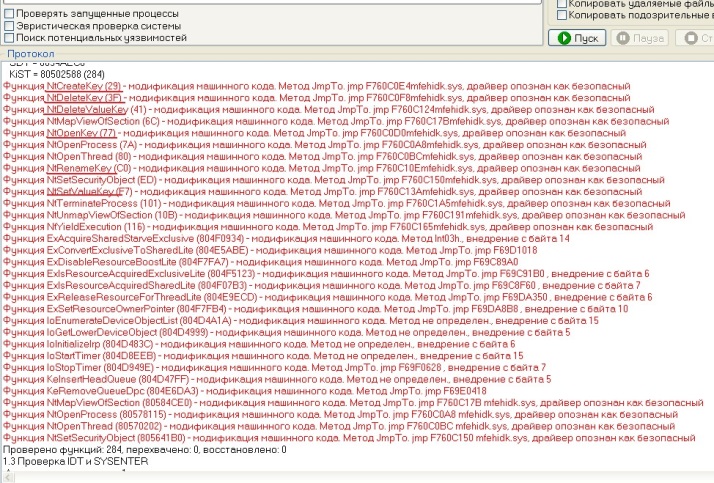


Рис 1.6 – Контроль змін в реєстрі за допомогою McAfee Antivirus Plus

У випадку зміни реєстру функція встановлює в сигнальний стан попередньо створений об’єкт типу Event (подія), і антивірусу залишається тільки відстежити цю подію й прийняти відповідні міри.

Операційна система викликає зареєстрований обробник у двох випадках: коли процес створюється і коли процес завершується. В першому з них функція повідомлення викликається, коли початковий потік уже створений, але його виконання ще не почалось. У другому випадку операційна система викликає функцію повідомлення перед завершенням останнього потоку в процесі.

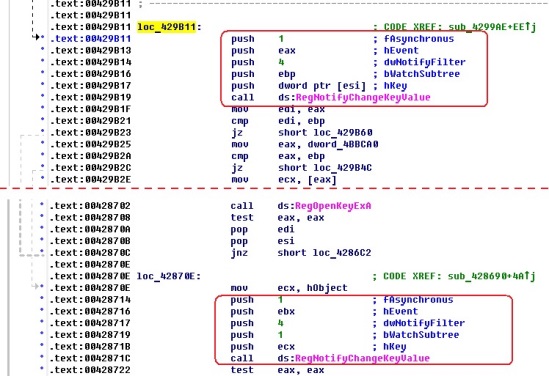


Рис 1.7 - Контроль над змінами в реєстрі

**РОЗДІЛ 2**

**РОЗРОБКА СИСТЕМИ ІДЕНТИФІКАЦІЇ СТАНУ КОМП‘ЮТЕРНОЇ СИСТЕМИ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ**

**2.1 Алгоритм та структура програми**

Найбільш значні дії евристичного аналізатора це обробка та калькуляція вхідних даних, які потрапляють в систему за допомогою спеціально відведеного файлу.

Алгоритм глобального пошуку буде виконаний методом FindFiles. У цьому алгоритмі спочатку виконується перебір всіх значень переліку обраних шляхів пошуку, після чого перевіряється наявність цього шляху та викликається метод локального пошуку. Ці дії повторюються для кожного обраного елементу переліку пошуку. Алгоритм локального пошуку, який виконується рекурсивним методом Find, більш складний. При його розробці використані можливості середовища програмування Java.

Встановлення переліку ліній вхідних даних є порядковий перебір файлу з першої строки до останньої на наявність потенційного вірусу або від втручання інфікованого вірусом персонального комп’ютеру кінцевого користувача, який намагається отримати доступ до операційної системи.

Спочатку виконується пошук всіх строчок ліній файлу за вказаним шляхом. Знайдена сигнатура розраховується на відношення до елементів, що фільтруються та при підтвердженні того, що сигнатура схожа на шкідливу - вона буде занесена у відповідний перелік результатів пошуку.

У алгоритмі виконується перебір всіх обраних елементів списку непотрібних файлів, перевіряється кожний елемент на його присутність на диску та виконується видалення.

– **void updateKnowledges(int neuronWinner, double[] UoutZ, double neuron Norma***)* – оновити знання для заданого нейрону, заданим вектором і встановити норму нейрона;

– **double[] countUOutZ(double[] inputLine, double[] t***)* – вирахувати *Uвих* для *Z* шару;

– **boolean isImageIdentified(double inputNorma, double neuronNorma***)* – чи знайомий вже образ, чи ні;

– **double[] countUinputY(double[] inputLine***)* – вирахувати Uвх для *Y* шару;

– **int findNeuronWinner(double[] j***)* – знайти номер нейрона-переможця;

– **double countNorma(double[] inputVector***)* – вирахувати норму для заданого вектора.

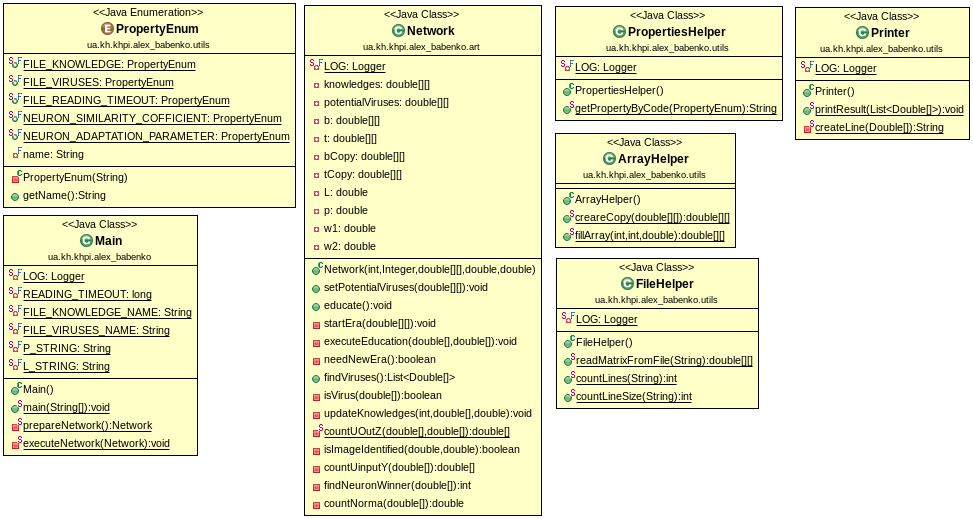


Рис. 2.1 Принцип взаємодії усіх класів проекту між собою.

**2.2 Структура вхідних даних**

Було проаналізована РЕ-структура шкідливого та безпечного програмного забезпечення а саме: 290 файлів типу Worm, 1050 файлів типу Trojan, 1153 файлів типу Backdoorта 1000 безпечних файлів та виділено їх ознаки у вигляді АРІ-функції та строк.Надалі виконано аналіз отриманих ознак та сформована бітова маски ознак шкідливого програмного забезпечення. Проведено сканування 300 шкідливих та 100 безпечних програм за отриманою бітовою маскою та отримано набір ознак кожного файлу у вигляді бітового рядка.

Рядок містить наявність та відсутність певних функцій, строк, звертань до реєстру та наявності підпроцесів (1 – наявний, 0 – відсутній).

Таким чином сформовано набір векторів який має 49 унікальних ознак у порядку зменшення їх частоти згадування в файлах, частина їх наведено у табл 3.1.

Повний перелік ознак: callnexthookex, copyfile, createfile, enumcalendarinfo, findfirstfile, getcurrentprocessid, getdesktopwindow, getdevicecaps, getdiskfreespace, getkeyboardstate, getmonitorinfo, getstartupinfoa, getsysteminfo, getversion, getwindowsdirectory, inet\_addr, regcreatekey, registerserviceprocess, regopenkey, regqueryvalue, regsetvalue, setwindowshook, shellexecute, socket, unhookwindowshook, writeprivateprofilestring, runservices, explore.exe, explorer.exe, regedit.exe, shellapi, win.exe, winexec, winny.exe, gethostbyn, gethostname, host, admin, hostname, localhost, sendmail, userhost, wininet, wnet, getmodulefilename, getmodulehandle, loadlibrary, setfilepointer, virtualalloc.+

**Ознаки шкідливих сигнатур** Таблиця 3.1

|  |  |  |
| --- | --- | --- |
|  | Функції/строки | Кількість у вірусах |
| 1 | callnexthookex | 42 |
| 2 | copyfile | 132 |
| 3 | createfile | 178 |
| 4 | enumcalendarinfo | 43 |
| 5 | findfirstfile | 127 |
| 6 | getcurrentprocessid | 70 |
|  |  |  |
| 37 | host | 10 |
| 38 | admin | - |
| 39 | hostname | 9 |
| 40 | localhost | 19 |
| 41 | sendmail | 28 |
| 42 | userhost | 9 |
| 43 | wininet | 35 |
| 44 | wnet | 55 |
| 45 | getmodulefilename | 206 |
| 46 | getmodulehandle | 247 |
| 47 | loadlibrary | 135 |
| 48 | setfilepointer | 158 |
| 49 | virtualalloc | 158 |

**2.3. Робота евристичного аналізатора**

Після запуску програми на екран виводяться шкідливі сигнатури, що були розпізнані з коефіцієнтом подібності (база – Backdoor, вибірка – всі сигнатури 400шт.) (Рис 2.2).

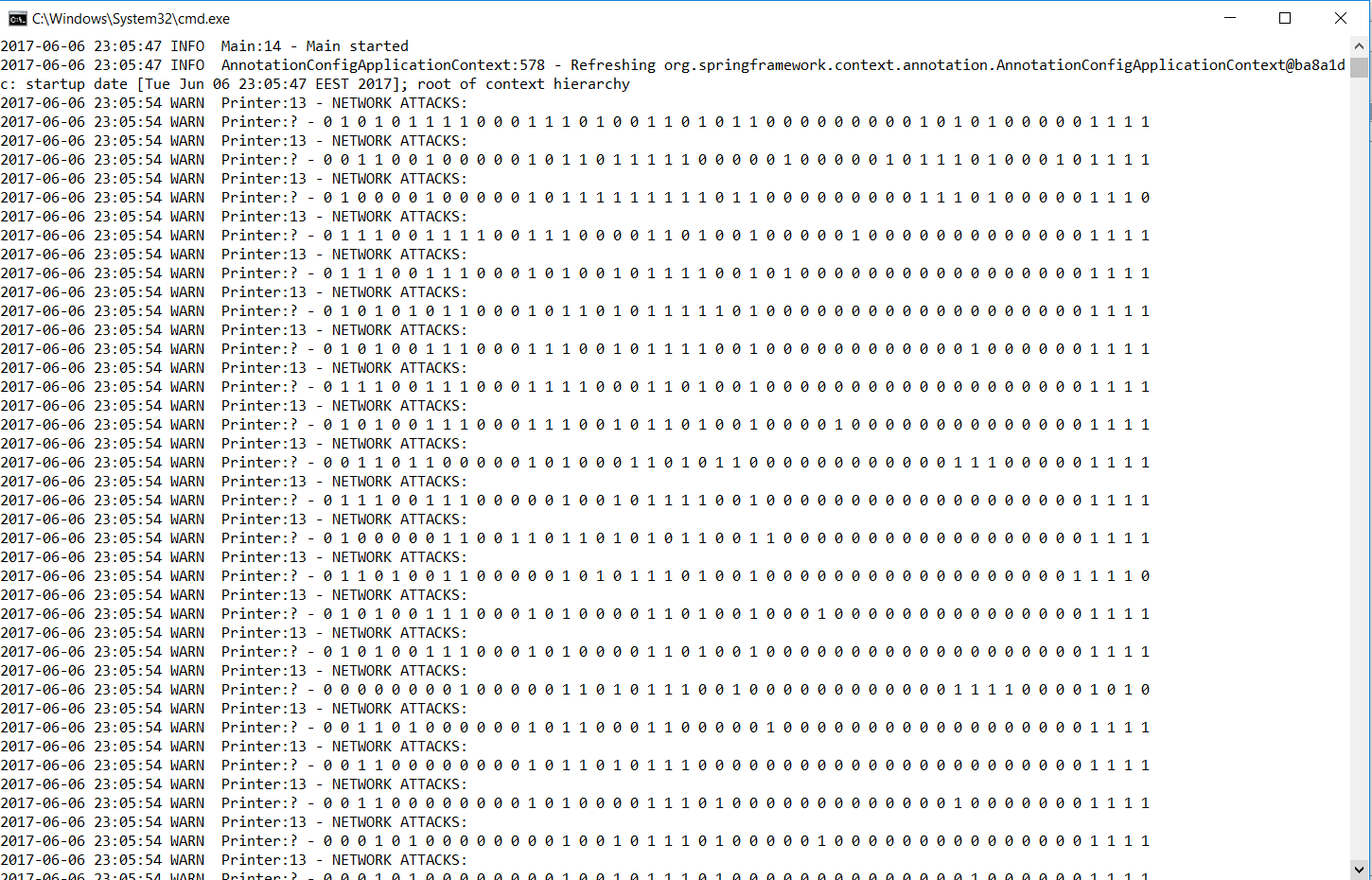


Рис. 2.2 Результати знаходження шкідливих сигнатур з консольним виводом

Результати евристичного аналізатора на основі нейронної мережі АРТ-1 можна аналізувати опираючись на дані, які представляються програмою шляхом логування. Шкідливі сигнатури знайдені за допомогою нейронної мережі будуть відображені у файлі *logs.log*(Рис 2.3 та Рис 2.4).

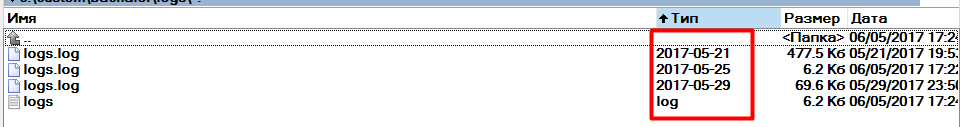


Рис 2.3 Результати роботи програми залоговані й розбиті на файли за датами



Рис 2.4. Результати роботи програми записані у файл

Наступним кроком для отримання релевантних результатів випробування, вихідні дані з нейронної мережі треба відфільтрувати, процес фільтрування зображено на рис.2.5. У якості допоміжної програми використовується NotePad++. Зліва – результати отримані в консолі (або log файлі), справа – всі сигнатури, які були подані на вхід нейронної мережі для розпізнавання.

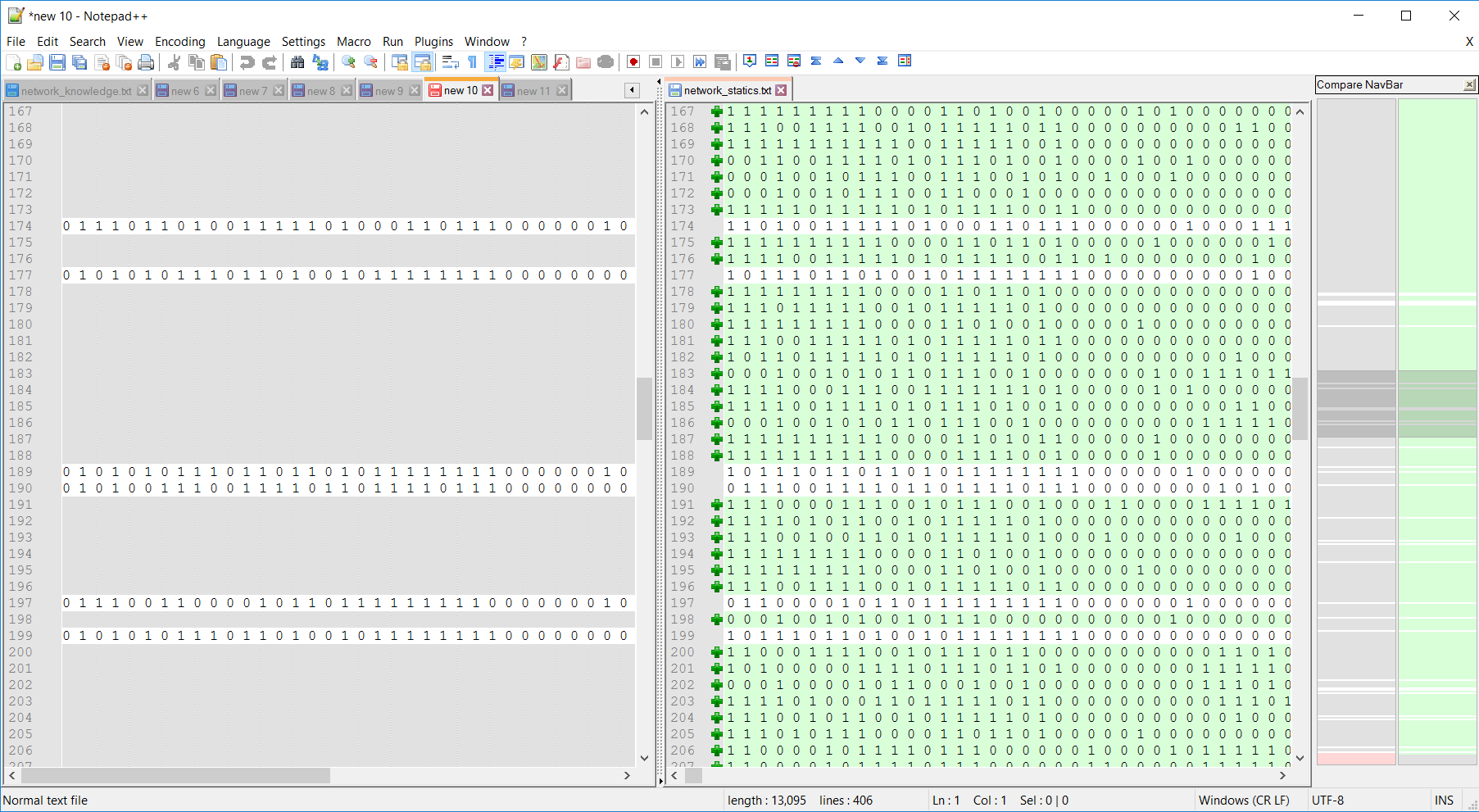


Рис 2.5 Фільтрування результатів за допомогою NotePad++

Таким чином отримані результати використані для подальшого аналізу, експериментального підбору оптимального коефіцієнту подібності, визначення ефективності нейронної мережі в залежності від вхідних даних. **2.4 Аналіз вихідних даних системи ідентифікації**

Для правильного функціонування системи ідентифікації стану експериментальним шляхом підібрано оптимальний коефіцієнт подібності.

Перший етап. За базові знання нейронної мережі взято 100 сигнатур типу Backdoor. Початковий коефіцієнт – 0.6, оскільки являється мінімальним допустимим для нейронної мережі. На вхід для розпізнавання подано сигнатури безпечних файлів, що означає жодна з сигнатур не повинна бути розпізнана. Оскільки на вхід подано сигнатури лише безпечних файлів, то очікуваний результат – пуста консоль. Якщо на консоль виведені сигнатури – це означає, що сигнатури були помилково розпізнані. (Рис 2.6)

Таблиця 2.2

**Експериментальний підбір оптимального коефіцієнту подібності для Backdoor**

|  |  |
| --- | --- |
| Коефіцієнт подібності | Кількість помилкових розпізнавань безпечних сигнатур |
| 0,6 | 15 |
| 0,7 | 10 |
| 0,8 | 1 |
| 0,84 | 0 |

Відсутності помилкових спрацьовуванні вдалося досягти при коефіцієнті 0,84 для сигнатур типу Backdoor.

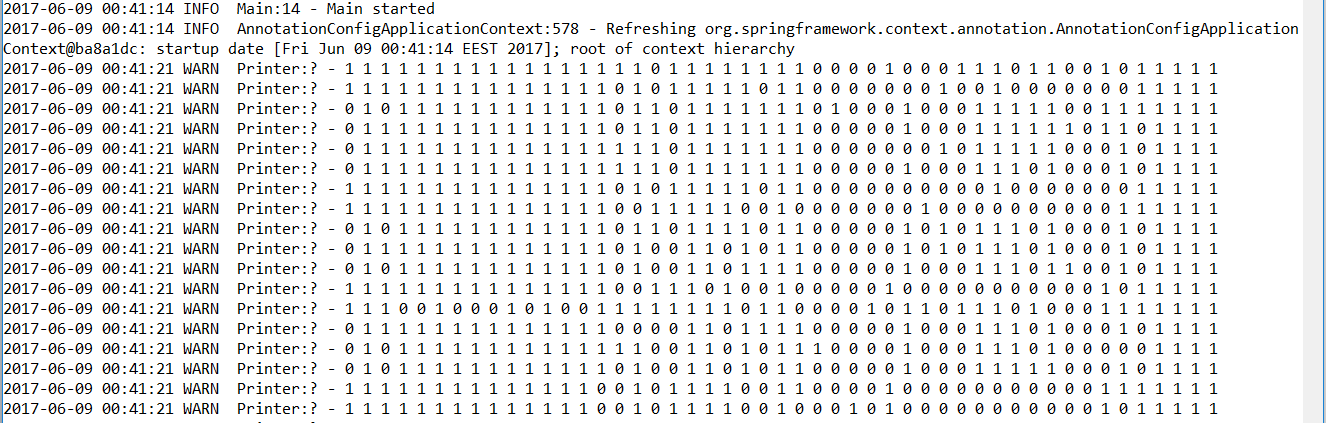


Рис 2.6. Помилкові спрацьовування при коефіцієнті 0,6

Другий етап. За базові знання нейронної мережі взято 100 сигнатур типу Trojan. Початковий коефіцієнт – 0.6, оскільки являється мінімальним допустимим для нейронної мережі. На вхід для розпізнавання подано сигнатури безпечних файлів. Помилкові спрацьовуванні стали відсутні при коефіцієнті 0,7 для сигнатур типу Trojan.

Таблиця 2.3

**Експериментальний підбір оптимального коефіцієнту подібності для Trojan**

|  |  |
| --- | --- |
| Коефіцієнт подібності | Кількість помилкових розпізнавань безпечних сигнатур |
| 0,6 | 10 |
| 0,65 | 3 |
| 0,7 | 0 |

Третій етап. Базові знання – 100 сигнатур типу Worm. Початковий коефіцієнт – 0.6. На вхід для розпізнавання подано сигнатури безпечних файлів. Помилкові спрацьовуванні стали відсутні при коефіцієнті 0,97 для сигнатур типу Worm.

Таблиця 2.4

**Експериментальний підбір оптимального коефіцієнту подібності для Worm**

|  |  |
| --- | --- |
| Коефіцієнт подібності | Кількість помилкових розпізнавань безпечних сигнатур |
| 0,6 | 29 |
| 0,7 | 14 |
| 0,8 | 17 |
| 0,9 | 10 |
| 0,97 | 0 |

**Оптимальний коефіцієнт розпізнавання** Таблиця 2.5

|  |  |
| --- | --- |
| Тип сигнатури | Оптимальний коефіцієнт |
| Backdoor | 0.84 |
| Trojan | 0.7 |
| Worm | 0.97 |

На рис. представлено оптимальний коефіцієнт подібності, згруповані за типом шкідливих сигнатур. Оптимальним коефіцієнтом вважається той, за яким відсутні помилкові розпізнавання (Рис 2.7).

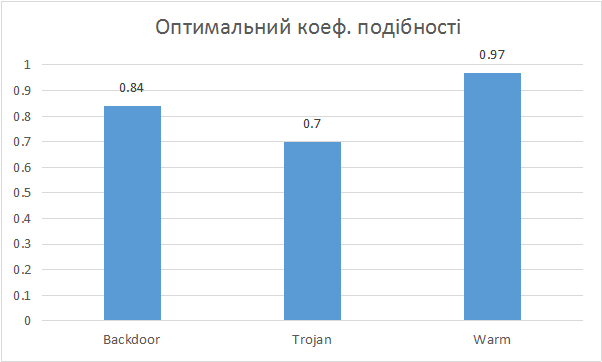


Рис 2.7. Оптимальний коеф. подібності

Оптимальне положення коефіцієнту подібності обирається на основі середнього значення за типами вірусів.

**ВИСНОВОК**

В даній роботі проаналізовано методи побудови антивірусних програм. Розглянуто методи побудови евристичних сканерів на базі різних типів нейронних мереж.

Було проаналізовано РЕ-структуру шкідливого та безпечного програмного забезпечення а саме: 290 файлів типу Worm, 1050 файлів типу Trojan, 1153 файлів типу Backdoorта 1000 безпечних файлів та виділено їх ознаки.Проведено сканування 300 шкідливих та 1000 безпечних програм та отримано набір вхідних даних для навчання нейронної мережі.

Розроблено програмну модель евристичного аналізатора на базі нейронної мережі АРТ-1, знайдено оптимальні коефіцієнти подібності, та проведено тестування розробленої системи виявлення комп'ютерних вірусів.

В процесі роботи евристичний аналізатор обробляє вхідні дані за алгоритмом, визначеним архітектурою нейронної мережі АРТ-1, визначає сигнатури та їх модифікації на основі попередньо навчених сигнатур. Опираючись на отримані знання в процесі навчання та аналізуючи нові вхідні дані нейронна мережа поповнює свої знання безперервно.Дослідження показали, що оптимальне положення обирається на основі середнього значення за типами вірусів.  
 За результатами експериментів виявлено, що програма може ідентифікувати близько 85 відсотків шкідливих сигнатур, майже з нульовим помилковим результатом. Отримані результати досліджень показали можливість використання розробленої системи ідентифікації шкідливого програмного забезпечення у якості допоміжного методу в системах захисту інформації

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1 . Мак-Каллок У. С., Питтс В. Логическое исчисление идей, относящихся к нервной активности // Автоматы / Под ред. К. Э. Шеннона и Дж. Маккарти. — М.: Изд-во иностр. лит., 1956.

2 . Розенблатт Ф. Принципы нейродинамики: персептроны и теория механизмов мозга = Principles of Neurodynamic: Perceptrons and the Theory of Brain Mechanisms. — М.: Мир, 1965.  
3. Барцев С. И., Охонин В. А. Адаптивные сети обработки информации. — Красноярск: Ин-т физики СО АН СССР, 1986. Препринт N 59Б.

4. Яхъяева Г.Э. Основы теории нейронных сетей. — М.: Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2006.

**ІНТЕРНЕТ-РЕСУРСИ**  
http://www.deeplearningbook.org

https://neuralnet.info/book/