Memoria de Prácticas Universitarias Extracurriculares

Universidad de Zaragoza
Estudiante: Julia Quero Pérez
NIP: 792310

Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

ÍNDICE

Datos personales y académicos

Nombre y apellidos	Julia Quero Pérez
Titulación	Programa Conjunto en Matemáticas-Ingeniería Informática
DNI	73131985N
Fecha de nacimiento	05/03/2003
Teléfono	601032380
Correo electrónico	792310@unizar.ess

Entidad donde realiza las prácticas

Centro de prácticas	UNIVERSIDAD DE ZARAGOZA. Escuela de Ingeniería y Arquitectura, laboratorios del Grupo DisCo. Zaragoza 50018 Zaragoza
Tutor de la entidad	Pedro Javier Álvarez Pérez-Aradros
Correo electrónico	alvaper@unizar.es
Teléfono	976552467
Tutor de la Universidad	Ricardo Julio Rodríguez Fernández
Departamento	Departamento de Informática e Ingeniería de Sistemas
Correo electrónico	rrodrigu@unizar.es

Objetivos propuestos

Las tareas a realizar durante las prácticas son las siguientes:

- Estudiar el uso de técnicas de IA al problema de la detección de malware.
- Creación de un dataset de referencia de código malware.
- Definición de un marco de características de interés para la detección de malware aplicando técnicas de machine learning.
- Programación y test de distintos modelos de clasificación y predicción para la detección de malware.
- Análisis de los resultados y conclusiones técnicas.
- Elaboración de un informe de prácticas.

Los objetivos propuestos sobre las tareas a realizar son los siguientes:

- Tratamiento de grandes volúmenes de información mediante programación.
- Diseño de redes neuronales convolutivas para el tratamiento de imágenes a partir de redes pre-entrenadas.
- Comprensión y aplicación del método del estudio de los n-gramas de las llamadas al sistema para la detección de malware.
- Generación de imágenes a partir de trazas de ejecución de malware; y estudio y propuesta de formatos para las mismas.

Para obtenerlos, se espera obtener competencia en las siguientes habilidades:

- Programación en Python usando varias de sus librerías (nltk, csv, json, PIL, Keras, InceptionV3, MobileNetV2, VGG16, VGG19, ResNet152 y otras comunes como collections, numpy, random o glob)
- Desarrollo de jupyter notebooks y trabajo en el entorno de Google Colab para la compilación y entrenamiento de redes neuronales.
- Uso de repositorios de GitHub.

Descripción detallada de las tareas realizadas

Sobre la generación de los n-gramas

A partir de un fichero csv en el que cada línea representa una traza, y en cuya primera columna está la clase de malware, en la segunda un identificador y de la tercera en adelante su traza de llamadas al sistema o categorías, genero un Counter de ngramas con la librería nltk.

Sobre las trazas de imágenes

Para la generación de las imágenes, se han tenido en cuenta los siguientes parámetros: la longitud del n-grama [n], la forma de elegir el color de las categorías [color], la forma de la imagen [formato], y la forma de distinguir dos categorías de n-gramas, una de mayor importancia que la otra [importancia].

La forma de generar las imágenes dada una traza es: primero se calculan sus n-gramas de longitud n. Luego se dividen sus n-gramas en dos listas, una con los considerados importantes que se encuentren presentes en la traza, y otra con el resto de los n-gramas presentes en la traza. Después, los n-gramas se codifican según el color y formato y se agregan a una imagen de 224x224 px con fondo blanco según el formato de la imagen, primero de mayor ocurrencia total a menor ocurrencia total los pertenecientes a la lista de

importantes, y después de igual manera los pertenecientes a la lista de los no importantes. En caso de haber más n-gramas que espacio en la imagen estos no se muestran y en caso de haber menos la zona sin ocupar permanece blanca.

Para el caso de imágenes con importancia afinidad, las imágenes se generan tomando tantas listas de importancia como número de categorías de virus haya en el dataset, y la imagen se particiona en tantas partes iguales como número de categorías de virus. Estas partes se van completando con dichas listas de importancia, una en cada parte, siguiendo el formato especificado.

Los valores que pueden tomar cada parámetro, ordenados por parámetro, son:
n

- 3
- 4
- 5
- 6

importancia

- [min_sup] Toma como n-gramas importantes aquellos que aparezcan en más del 60% de las trazas pertenecientes a su misma categoría.
- [contrast] Siguiendo el método presentado en [2]: primero toma los n-gramas que aparezcan en más del 60% de todas las trazas (minimum support del 60%). Después, calcula el contraste, como indicado en el artículo, y toma los media/2 n-gramas con mayor contraste, donde media es la media de n-gramas por traza.
- [afinidad] Es una variante del método presentado en [2]: primero toma los n-gramas que aparezcan en más del 60% de todas las trazas (minimum support del 60%). Después, calcula el contraste, y selecciona los media/(número de categorías de virus en el dataset) n-gramas con mayor contraste de cada categoría.
- [relevant] Siguiendo el método presentado en [1] de n-gramas relevantes: para cada categoría genera una lista tomando los media/9 n-gramas que aparecen en más trazas del dataset completo. Después, elimina aquellos n-gramas en la intersección de las listas y crea la lista de n-gramas importantes con la unión restante.

formato

- [cat_id] Cada categoría de un n-grama se presenta como un cuadrado de lado óptimo. Estos cuadrados se colocan por filas (de arriba a abajo) de izquierda a derecha.
- [cat_aa] Cada categoría de un n-grama se presenta como un cuadrado de lado óptimo. Estos cuadrados se colocan por columnas (de izquierda a derecha) de arriba a abajo.
- [trans_id] Cada n-grama se presenta como un cuadrado de lado óptimo, en el que cada categoría del n-grama aporta la misma transparencia al cuadrado. Estos cuadrados se colocan por filas (de arriba a abajo) de izquierda a derecha.
- [trans_aa] Cada n-grama se presenta como un cuadrado de lado óptimo, en el que cada categoría del n-grama aporta la misma transparencia al cuadrado. Estos cuadrados se colocan por columnas (de izquierda a derecha) de arriba a abajo.
- [diag_sup] Cada categoría de un n-grama se presenta como un cuadrado de lado óptimo. Estos cuadrados comienzan a colocarse desde la esquina superior izquierda hacia abajo en diagonal, después completan en diagonal la imagen por debajo de la

diagonal principal hasta la esquina inferior izquierda y después por encima de la diagonal principal hasta la esquina superior derecha.

- [diag_inf] Cada categoría de un n-grama se presenta como un cuadrado de lado óptimo. Estos cuadrados comienzan a colocarse desde la esquina inferior izquierda hacia arriba en diagonal, después completan en diagonal la imagen por encima de la diagonal principal hasta la esquina superior izquierda y después por debajo de la diagonal principal hasta la esquina inferior derecha.
- [mix] Cada n-grama de la lista de n-gramas importantes se presenta como un cuadrado de lado óptimo, en el que cada categoría del n-grama aporta la misma transparencia al cuadrado. Estos cuadrados se colocan por filas (de arriba a abajo) de izquierda a derecha. Cada categoría de un n-grama de la lista de n-gramas no importantes se presenta como un cuadrado de lado óptimo, que se coloca por filas (de arriba a abajo) de izquierda a derecha.

color

- [equid] Se divide el espectro del negro al blanco en tantos tramos de igual longitud como número de categorías haya en la lista. El color se toma como el límite inferior del rango, de forma que todos son equidistantes. Tras esta división, los colores se asignan por el orden de la lista del más negro al más blanco.
- [equiran] Se divide el espectro del negro al blanco en tantos tramos de igual longitud como número de categorías haya en la lista. El color se toma como el límite inferior del rango, de forma que todos son equidistantes. Tras esta división de colores, se asignan de forma aleatoria entre las categorías de la lista.
- [rand] A cada categoría se le asigna un color aleatorio distinto al de las demás y al blanco.

Otros valores para dichos parámetros que se consideraron estudiar pero finalmente descartaron, son:

formato

- [banda] Para n=3 se propone por cada n-grama, obtener 3 cuadrados: uno central con igual transparencia para cada categoría, uno superior para las 2 primeras categorías del n-grama con igual transparencia, y otro inferior para las dos últimas categorías del n-grama con igual transparencia. Estos cuadrados se colocarían en forma de banda tridiagonal como en [diag_ing]: los cuadrados de un n-grama comienzan a colocarse desde la esquina inferior izquierda hacia arriba en diagonal, después completan en diagonal la imagen por encima de la diagonal principal hasta la esquina superior izquierda y después por debajo de la diagonal principal hasta la esquina inferior derecha. Para el resto de n no había una propuesta fija.

colores

- [wsi] Asignar a cada categoría valores del 0 al 254 según su afinidad al tratamiento de “web”, “sincronización” e “interfaz”, que se asignaría respectivamente a los colores rojo, verde y azul.
- [reord] Fijar una ordenación específica para la lista de categorías, y aplicar el método de selección de color de [equid] sobre esa ordenación de la lista.

Se han generado trazas para todas las permutaciones posibles de los valores de los parámetros, y se ha obtenido el weighted-average f1-score de 3 entrenamientos de modelo para cada una de las 2 trazas (Kim, Alain) y cada una de las 5 redes pre-entrenadas estudiadas (InceptionV3, MobileNetV2, ResNet152, VGG16 y VGG19).

Resultados obtenidos

Se ha obtenido el weighted-average f1-score para cada traza resultante de la media de 3 entrenamientos con la red indicada.
La red pre-entrenada InceptionV3 es con la que mejores resultados se obtuvieron, por lo que se priorizaron las pruebas con esta. Para la realización de las pruebas se intentó combinar los parámetros de forma que se pudiera abstraer cuándo estos influían en el resultado.
Así, se podría concluir, pendiente confirmación por la obtención de todas las combinaciones posibles, que los mejores valores para cada parámetro son los siguientes:
Para la longitud de los n-gramas:
El n-grama que mejor resultados obtiene es el de longitud 4, como se observa en la mayoría de pruebas realizadas con ambos datasets. Varios resultados mejoran con n=5, pero para la longitud 6 vuelven a disminuir, siendo en ocasiones mejores que para n=3 y en ocasiones peores.

Para la importancia:
Con diferencia, los mejores resultados para la selección de los n-gramas que se consideren importantes para cada familia de malware se obtienen con el valor [min_sup]. El resto de métodos obtienen resultados diversos dependiendo del formato seleccionado para la construcción del dataset de imágenes, por lo que no parece claro abstraer una ordenación entre ellos.

Para el formato:
En general, la colocación horizontal y luego vertical de los colores ([cat_id], [trans_id]) muestran mucho mejores resultados que las inversas ([cat_aa], [trans_aa]). Los métodos con transparencia ([trans_id], [trans_aa]) son mejores que los que no la usan, a excepción de [diag_inf], que muestra los mejores resultados en la mayoría de pruebas. El método [mix] ha probado no ser una mejora en la selección del formato.

Para el color:
Se observa que los resultados son mejores cuando el método de selección de color asigna colores equidistantes en la escala RGB, como son [equid] y [equiran]. Para algún caso se obtienen mejores resultados con [rand], pero al no conocer los colores asignados no se puede abstraer qué hizo que se obtuvieran mejores resultados.

Aprendizaje relacionado con la titulación

Las tareas asignadas a la práctica se corresponden con las acciones formativas del plan de estudios. Permiten adquirir y desarrollar competencias específicas de la titulación y competencias transversales tales como la capacidad de aprendizaje, comunicación, autonomía, trabajo en equipo, iniciativa, capacidad de adaptación, habilidades interpersonales y preocupación por la calidad.
Durante el desarrollo de la práctica se han aplicado los siguientes conocimientos obtenidos en la titulación:

- Programación básica (Programación I, Programación II, Estructuras de datos y algoritmos)
- Manejo de la terminal y mecanización del trabajo (Sistemas Operativos)

Además, se han obtenido los siguientes conocimientos:

- Tipos de redes neuronales, diseño y aplicaciones (Inteligencia artificial)
- Tipos de malware, comportamiento y métodos para su reconocimiento (Seguridad Informática)

Bibliografía

[1] Dai, J., Guha, R., & Lee, J. (2009). Efficient virus detection using dynamic instruction sequences. *Journal of computers*, *4*(5). https://doi.org/10.4304/jcp.4.5.405-414

[2] Reddy, D. K. S., & Pujari, A. K. (2006). N-gram analysis for computer virus detection. *Journal in Computer Virology*, *2*(3), 231–239. https://doi.org/10.1007/s11416-006-0027-8

Contenidos IASAC. (s/f). Unizar.es. Recuperado el 30 de septiembre de 2023, de https://unidigitaliasac.unizar.es/contenidos-iasac

csv — Lectura y escritura de archivos CSV. (s/f). Python documentation. Recuperado el 30 de septiembre de 2023, de https://docs.python.org/es/3/library/csv.html

NLTK :: nltk package. (s/f). Nltk.org. Recuperado el 30 de septiembre de 2023, de https://www.nltk.org/api/nltk.html

collections — Container datatypes. (s/f). Python documentation. Recuperado el 30 de septiembre de 2023, de https://docs.python.org/3/library/collections.html

Clasificación de imágenes. (s/f). TensorFlow. Recuperado el 30 de septiembre de 2023, de https://www.tensorflow.org/tutorials/images/classification?hl=es-419

Kaspersky IT Encyclopedia. (s/f). Kaspersky.com. Recuperado el 30 de septiembre de 2023, de https://encyclopedia.kaspersky.com/

json — Codificador y decodificador JSON. (s/f). Python documentation. Recuperado el 30 de septiembre de 2023, de https://docs.python.org/es/3/library/json.html

Pyllow (s/f). PyPI. Recuperado el 30 de septiembre de 2023, de https://pypi.org/project/Pillow/

Layer weight regularizers. (s/f). Keras.Io. Recuperado el 30 de septiembre de 2023, de https://keras.io/api/layers/regularizers/

EarlyStopping. (s/f). Keras.Io. Recuperado el 30 de septiembre de 2023, de https://keras.io/api/callbacks/early_stopping/