Reporte de proyecto final

Redes neuronales 2024-2
Hernández Vela Daniel

Delimitación del problema

Detección de URLs maliciosas

En la actualidad existe una creciente cantidad de ataques y amenazas en internet que utilizan URLs maliciosas como medio para distribuir malware, phishing, fraudes, etc. El objetivo de este proyecto es clasificar URLs maliciosas o potencialmente peligrosas con un enfoque de no tener que acceder a su contenido, es decir, basándose meramente en el texto que las conforma (en este caso, se clasifica entre URLs seguras, defacement, phishing y malware), esto se logrará extrayendo información relevante del texto de una URL, las variables de entrada que se proponen son las siguientes:

- Tamaño del URL
- Densidad de vocales presentes en el URL
- Densidad de consonantes presentes en el URL
- Densidad de símbolos presentes en el URL
- Densidad de vocales seguidas de consonantes
- Densidad de consonantes seguidas de vocales
- Densidad de vocales seguidas de vocales
- Densidad de consonantes seguidas de consonantes
- Densidad de dígitos seguidos de dígitos
- Contiene la cadena "www" o no
- Densidad de mayúsculas
- Tamaño del path
- Número de consultas realizadas

Dadas las cuatro clases en que se clasificarán las URLs, las variables de salida son:

- Benign (representado con "1"), no tiene una intención maliciosa detrás.
- Defacement (representado con "2"), dirige a un sitio web alterado por un atacante.
- Phishing (representado con "3"), dirige a un sitio web diseñado para engañar a usuarios con fines maliciosos.
- Malware (representado con "4"), dirige a un sitio web que distribuye software malicioso.

Conjunto de entrenamiento

El dataset utilizado es "Malicious URL dataset", el cual se puede encontrar en Kaggle (véase la bibliografía). Es importante mencionar que se realizaron cuatro modificaciones al dataset original:

1. El dataset consiste de un listado de URLs en formato string junto con su respectiva etiqueta, dado que más información no es proporcionada es necesario realizar un proceso de extracción de información para

obtener los datos de entrada antes mencionados, este proceso se puede observar en el notebook del proyecto.
2. El dataset es una recopilación de otros dataset que se pueden encontrar públicamente en internet, en la sección de discusión del dataset en Kaggle un usuario mencionó que existe un error en el etiquetado de algunas URL en el dataset, específicamente en aquellas obtenidas del dataset "phishstorm" (véase la bibliografía), por lo que es necesario realizar un proceso de reetiquetado, este trabajo se realiza descargando el dataset "phishstorm" para luego media un script en Python generar un nuevo dataset que será utilizado para el entrenamiento.
3. Durante el proceso de reetiquetado se encontró que existen entre 10 y 20 filas en el archivo que parecen estar mal codificadas y causan errores al trabajar con el dataset, por lo que estas fueron removidas manualmente.

Una vez obtenido el dataset final, se realiza la extracción de información resultando en 14 columnas, que serán las entradas de la red neuronal. El dataset se divide en 70% para el conjunto de entrenamiento y 30% para el conjunto de prueba, se utilizó estratificación con el objetivo de que el desbalance en los tipos de URL no represente una dificultad aún mayor durante el entrenamiento.

Nota

Al realizar pruebas con el entrenamiento de la red se encontró que el modificar el conjunto de entrenamiento de tal manera que las URLs de tipo "benign" sean reducidas a la mitad resulta en una mejora significativa para los resultados de la red, el porcentaje que se encontró adecuado fue 50% pues reducirlo menos causa que la red tenga una dificultad importante para entrenarse, y reducirlo más puede causar que la red no tenga una buena generalización con URLs fuera del conjunto de entrenamiento.

Arquitectura de la red

Se optó por utilizar una red neuronal multicapa pues esta arquitectura es adecuada para procesar la información numérica obtenida a partir de la extracción de información, dado que la información extraída del dataset no crece exponencialmente, incluso al alimentar la red con aún más información resulta ser suficiente, y provee una mayor facilidad en su implementación y ajuste.

Las características de la red son las siguientes:

- Tres capas ocultas de tamaño 50, 50 y 25 respectivamente
- Activación reLU para las tres capas ocultas, Softmax en la capa de salida
- Función de error Cross Entropy

Durante el entrenamiento se utilizaron números diferentes de capas ocultas y de neuronas para estas, como resultado de varias pruebas tres capas dan mejores resultados para este problema, es interesante notar que aumentar la cantidad de capas ocultas y de neuronas no solo no representa una mejora en el desempeño de la red, si no que esto incluso causó que el entrenamiento se viera atascado en un punto donde el error dejaba de disminuir completamente.

Entrenamiento

Para el entrenamiento se utilizaron los siguientes hiperparámetros:

- Optimizador Adam
- Learning rate = 0.02

- Épocas de entrenamiento: 10000

El optimizador Adam resulto ser mucho más efectivo que SGD y ligeramente más efectivo que Adagrad, logrando reducir el error a una velocidad mayor, la gráfica de error muestra más picos que en el caso de los otros optimizadores mencionados, sin embargo, la gráfica no tiene variaciones excesivas y la clasificación que se observa en la matriz de confusión y el porcentaje de precisión para el conjunto de prueba resultan ser mejores.

En el caso del Learning rate, mientras mayor es el valor, los resultados son peores pues la red no logra una generalización adecuada, un valor menor al mencionado no representa una mejora significativa por lo que el rango de $[0.01, 0.02]$ es el más adecuado.

Con lo anterior, 5000 épocas de entrenamiento pueden ser suficientes pues en las pruebas aproximadamente a partir de la época 4000 la disminución del error disminuye significativamente su velocidad, sin embargo, al hacer uso de CUDA para optimizar los tiempos de entrenamiento se utilizaron 10000 épocas, esto para determinar si representa una mejora que valga la pena, se observó que aún se logra reducir el error lentamente pero sin estancarse.

Evaluación final

La red ya entrenada logró una precisión de aproximadamente 88.75%, dada la matriz de confusión se puede observar que la red sorpresivamente obtuvo buenos resultados al distinguir entre el tipo de URL "benign" y "defacement", pues el identificar una URL de tipo "defacement" muchas veces conlleva explorar el sitio web al que dirige, lo que indica que en general existen patrones entre las URLs de los sitios que son alterados por los atacantes. Tuvo más dificultad al distinguir entre URLs de tipo "benign" y "phishing", lo cual tiene sentido considerando que para crear este tipo de URL los atacantes recurren a ingeniería social con el objetivo de hacer pasar URLs maliciosas por benignas, sin embargo, obtuvo resultados decentes. En el caso del tipo malware se obtuvieron buenos resultados a pesar de ser la clase con menor cantidad de URLs en el dataset, este tipo de URLs en ocasiones pueden resaltar a simple vista pues estas tienen características comunes que las diferencian de los otros tipos, por ejemplo, su longitud comúnmente más larga, cantidad de dígitos numéricos que aparecen, cantidad de símbolos, etc.

Al realizar predicciones para URLs que no se incluyen dentro del dataset, es posible determinar intuitivamente algunas de sus clases, por ejemplo, google.com (benign), wikipedia.org (benign), usgoogle-com.net (phishing), etc. Dado lo anterior, se puede observar que la red puede llegar a fallar en URLs que son cortas y no proporcionan tanta información (google.com, x.com, etc.), sin embargo, para URLs más extensas la red tienen un mejor desempeño, como se observa en la visualización de predicciones en el conjunto de prueba.

Finalmente, a pesar del buen desempeño de la red, aún existen áreas de oportunidad. Una manera de mejorar la precisión y lograr que la red sea mejor generalizando puede ser recopilar más datos, especialmente URLs de tipos maliciosos, así como tratar de extraer más características en las URLs que puedan proporcionar más información útil a la red, por lo que se continuará trabajando en este proyecto en el futuro recopilando datos, buscando más información útil extraible y experimentando con otras arquitecturas de redes neuronales.