IMF Smart Education

MÁSTER PRESENCIAL EN CIBERSEGURIDAD – 8ªEd.

DETECCIÓN DE MENSAJES OCULTOS EN IMÁGENES

TFM elaborado por: Alexia Cazón López

Tutor de TFM: Carlos Morales Diego

Madrid, 2023

TABLA DE CONTENIDO

ÍNDICE ILUSTRACIONES

ÍNDICE TABLAS

RESUMEN

Durante esta memoria se profundizará en los métodos más conocidos de esteganografía y qué funciones hay detrás de ellos para posteriormente poder aplicarlos a un conjunto de imágenes. Se analizarán los métodos de estegoanálisis correspondientes para detectarlos y los pondremos a prueba. Más allá de hacer esta revisión de los trabajos más relevantes, indagaremos sobre una nueva herramienta que tenga en cuenta todo lo mencionado en la primera parte del trabajo y como complementarlo con las nuevas tácticas de moda y que día a día se descubren e investigan en este campo. Así, se hará un repaso de los modelos más óptimos de machine learning y las funciones y fórmulas matemáticas más eficientes para extraer ciertas características de las imágenes que permitirán a la herramienta decidir sobre la detección de mensajes ocultos en imágenes. Durante las distintas partes se hará uso de Python y de distintos programas creados en este lenguaje que están disponibles en el siguiente GitHub: https://github.com/alexiacl/Esteganografia.

ABSTRACT

During this memory, the best-known steganography methods will be studied in depth and what functions are behind them to later be able to apply them to a set of images. The corresponding stegoanalysis methods will be analyzed to detect them and we will put them to the test. Beyond making this review of the most relevant works, we will inquire about a new tool that takes into account everything mentioned in the first part of the work and how to complement it with the new trends that are discovered and investigated every day in this field. Thus, a review will be made of which are the best machine learning models and the best functions and mathematical formulas to extract certain characteristics from the images that will ultimately allow a decision to be made on the detection of hidden messages in images. During the different parts, Python will be used and different programs created in this language that are available at the following GitHub: https://github.com/alexiacl/Esteganografia.

INTRODUCCIÓN

Para poder abordar este tema se ha de empezar preguntando como se puede llegar a ocultar contenido malicioso en imágenes y cómo se podría detectar para poder hacerle frente.

La primera pregunta tiene respuesta en la Esteganografía, disciplina que estudia la aplicación de tácticas que permiten encubrir objetos dentro de otros, por ejemplo, en imágenes y que estos no se perciban.

La segunda se debe plantear desde un punto de vista de mejora incesante y que sea capaz de estar adaptándose continuamente a los cambios y avances que surgen en esta parte de la ciberseguridad. Estas características de adaptación, mejora continua y aprendizaje mecánico son en los que se basa el Machine Learning. Es por ello, que es interesante afrontar la ocultación de mensajes o de contenido malicioso con una herramienta que esté basada en esta forma de inteligencia artificial.

La aproximación que se seguirá para detectar contenido malicioso en imágenes es la siguiente:

1. Investigación sobre las técnicas y tácticas de esteganografía y como hacer que salte la alarma mediante estegoanálisis.
2. Elección de características que se tendrán en cuenta para llegar a la conclusión de que existe contenido malicioso en las imágenes.
3. Clasificación de las imágenes en libres o no de contenido oculto con algoritmos basados en Machine Learning y programados en Python.

FUNDAMENTO TEÓRICO

En este capítulo se analizará la base conceptual, las teorías y los principios que sustentan la esteganografía y el estegoanálisis y que guiarán y supondrán un ejemplo para la creación de una herramienta con una base sólida para detectar contenido oculto en imágenes.

PRINCIPALES MÉTODOS DE ESTEGANOGRAFÍA

Antes se introdujo la Esteganografía y para poder hacerle frente se debe hablar también del Estegoanálisis, que hace referencia al conjunto de técnicas que estudian como detectar mensajes ocultos en algún objeto. Se puede abordar el Estegoanálisis desde distintos puntos, se empezará nombrando las distintas técnicas usadas para ocultar algún tipo de contenido en imágenes para después distinguir como detectarlas.

Existen distintas técnicas de esteganografía en imágenes: basadas en el dominio espacial (inserción del mensaje secreto en los bits menos significativos de la imagen), en el dominio de la frecuencia (inserción del mensaje oculto en zonas de alta frecuencia de la imagen mediante una transformada) y en el formato de la imagen (inserción del mensaje oculto en las cabeceras o al final del archivo).

A partir de esta clasificación se será capaz de decidir en qué propiedades de las imágenes hay que fijarse y cuáles deben de tenerse en cuenta para nuestra herramienta y así poder concluir si una imagen oculta mensajes o información maliciosa.

DOMINIO ESPACIAL

Para entender este método se introducirá el modelo RGB (Red, Green, Blue), en el que los colores rojo, verde y azul se mezclan en varias proporciones para formar una matriz de diferentes colores.

Cada uno de los componentes R, G y B son valores de 8 bits en el rango [0..255], siendo 255 el valor más grande que se puede almacenar (2^8 = 256 valores [0 ..255]). Así, el color de un píxel de una imagen es una combinación de los tres componentes.

A continuación, podemos ver la representación de estos tres colores en bits:

TABLA 1: VALORES RGB ROJO

R	1	1	1	1	1	1	1	1
G	0	0	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0

TABLA 2: VALORES RGB VERDE

R	0	0	0	0	0	0	0	0
G	1	1	1	1	1	1	1	1
B	0	0	0	0	0	0	0	0

TABLA 3: VALORES RGB AZUL

R	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0
B	1	1	1	1	1	1	1	1

Se puede ver un ejemplo sobre como la modificación de los dos últimos bits es imperceptible a la mirada humana.

TABLA 4: VALORES RGB VERDE ORIGINALES

R	0	0	0	0	0	0	0	0
G	1	1	1	1	1	1	1	1
B	0	0	0	0	0	0	0	0

TABLA 5: VALORES RGB VERDE MODIFICADOS

R	0	0	0	0	0	0	0	0
G	1	1	1	1	1	1	0	0
B	0	0	0	0	0	0	0	0

Es por ello por lo que una de las técnicas más usadas en esteganografía es ocultar información en estos últimos bits sin que sea perceptible en la imagen final.

Una manera de detectar esta ocultación de tipo LSB, “Last Significant Bit” es mediante el histograma de la imagen, el cual organiza los niveles de cierta característica de la imagen por la frecuencia en la que aparecen. En concreto, hace un conteo del número de píxeles que tienen el mismo nivel de gris.

En el histograma de la imagen modificada se puede apreciar que los pares de barras consecutivos tienden a obtener una altura similar. Esto sucede porque al sumar uno a las barras pares, provoca que den parte de sus valores a la siguiente barra, mientras que al restar uno a las barras impares, provoca que den parte de sus valores a la barra anterior. (Lerch Hostalot , s.f.)

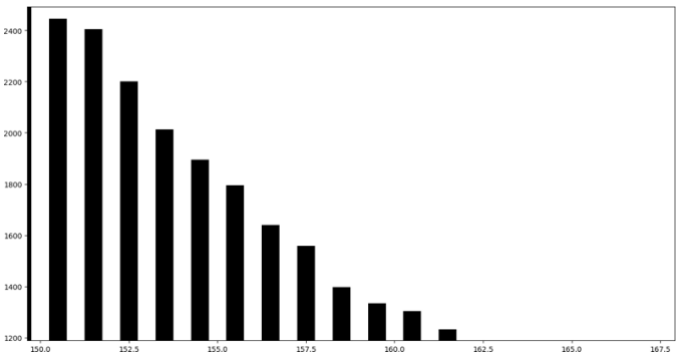


ILUSTRACIÓN 1: HISTOGRAMA DE LA IMAGEN ORIGINAL

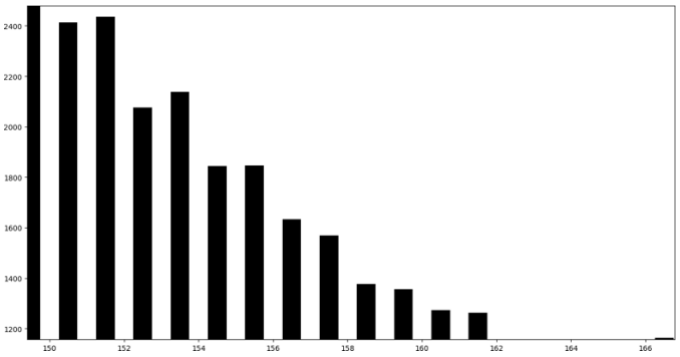


ILUSTRACIÓN 2: HISTOGRAMA DE LA ESTEGOIMAGEN

DOMINIO DE LA FRECUENCIA

Las series de Fourier sirven para encontrar las distintas funciones que componen una función periódica y continua a trozos a la que convergen, y que, además, son más sencillas que la función compleja inicial.

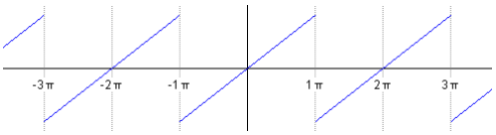


ILUSTRACIÓN 3: FUNCIÓN PERIÓDICA Y CONTINUA A TROZOS

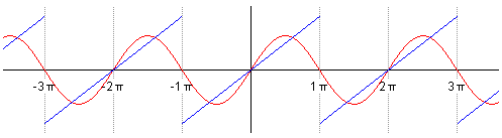


ILUSTRACIÓN 4: SERIES DE FOURIER

Se puede pensar en una imagen como una función que transmite información. Por lo explicado antes, se puede ver como un conjunto de funciones, en concreto, como una suma de senos y cosenos, es decir, como una serie de Fourier. Aquí, surge la transformada de Fourier, calculo matemático que desmonta una función para visualizarla con la frecuencia que la define. Por ello, una imagen se puede trazar en el dominio del espacio, que es como lo acostumbramos a visualizar comúnmente, haciendo uso de las coordenadas x e y, o en el dominio de la frecuencia, detallando con cuánta frecuencia se repiten ciertos patrones en una imagen.

ILUSTRACIÓN 5: IMAGEN EN DISTINTOS DOMINIOS

La transformada discreta del coseno o DCT, es una transformada basada en la transformada que antes definimos como discreta de Fourier, pero usando solo números reales. Expresa un vector finito de puntos en términos de una suma de funciones coseno, llevando una imagen del dominio del espacio al dominio de la frecuencia.

La técnica de la DCT fracciona una imagen en secciones de 8x8 píxeles para aplicarle la transformada discreta del coseno a cada una de ellas e insertar el mensaje secreto en los bits menos significativos de los componentes DCT.

Este reemplazamiento de bits produce una dependencia entre los valores de frecuencia de aparición afectando al histograma de la imagen, pues altera la frecuencia de los pares de coeficientes adyacentes.

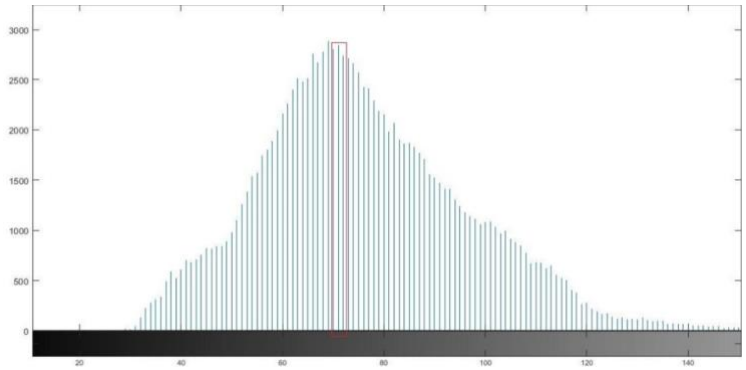


ILUSTRACIÓN 6: HISTOGRAMA IMAGEN ORIGINAL

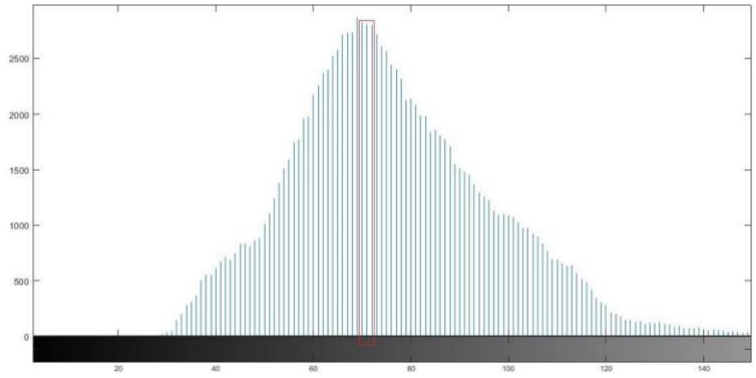


ILUSTRACIÓN 7: HISTOGRAMA ESTEGOIMAGEN

En este ejemplo se observa una representación del histograma de la imagen original y de la estegoimagen resaltando un extremo local máximo. Al insertar el mensaje se va a obtener que el histograma de los vecinos del punto máximo se suavizará para tener valores parecidos a la local mínima.

APROVECHANDO EL FORMATO DE LA IMAGEN

Una manera mucho más simple que las anteriores es aprovechando el formato de la imagen. Principalmente veremos los siguientes métodos.

Uno de ellos se basa en hacer uso del EOF, en sus siglas en inglés “End Of Life”, que es una señal que indica que no hay más información que ejecutar para visualizar la imagen. Esta técnica insertará el mensaje después de la etiqueta EOF y cuando se visualice con cualquier aplicación se ignorará todo lo que venga después de la marca, pero cuando se abra con un bloc de notas sí tendremos acceso al mensaje oculto. En la práctica, se haría ejecutando un comando específico del sistema operativo que copiará el mensaje de texto en el archivo de la imagen.

El segundo es agregando los datos de nuestro mensaje oculto en la información de archivo extendido de la imagen, EXIF, en sus siglas en inglés

“Exchangeable Image File Format”. El EXIF contiene todos los detalles sobre las especificaciones técnicas y los parámetros de disparo de las imágenes guardadas, pudiendo ocultar el mensaje en alguna de estas zonas.

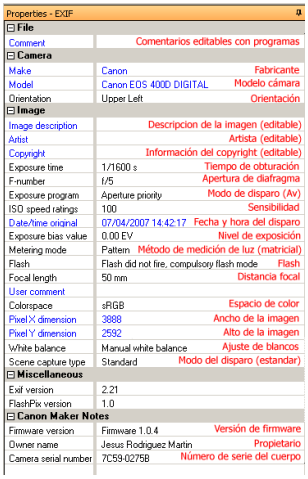


ILUSTRACIÓN 8: EXIF

Una manera sencilla de hacer frente a esta técnica es mediante la herramienta ExifTool, software de colaboración pública que permite leer y manipular metadatos mediante una interfaz de línea de comandos.

APLICANDO ESTEGANOGRAFÍA A UNA BASE DE DATOS DE IMÁGENES

Para poder poner en práctica los métodos analizados anteriormente se hará uso de una base de datos de imágenes obtenidas de Kaggle. A continuación, se puede ver una muestra de las imágenes a las que nos estamos refiriendo:

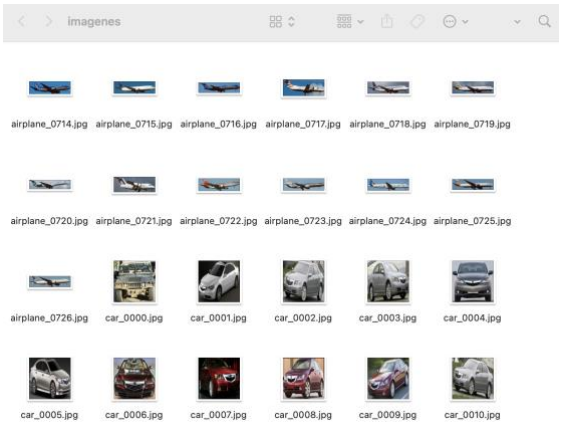


ILUSTRACIÓN 9: MUESTRA DE IMÁGENES

Además, se deberán pasar a otro formato ya que el método esteganográfico usado recoge las imágenes en .png. Para ello, se debe ejecutar jpgtopng.py que se puede encontrar en el GitHub mencionado. Tras esto, se habrá creado una nueva carpeta con las imágenes en el correspondiente formato:

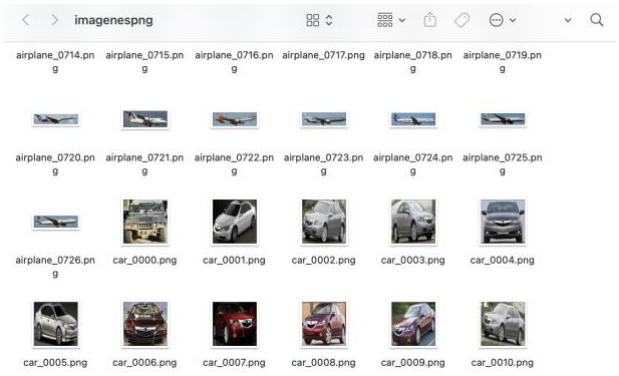


ILUSTRACIÓN 10: MUESTRA DE IMÁGENES .PNG

El objetivo ahora será insertar un mensaje oculto en ellas para poder después poner a prueba la herramienta con las estegoimágenes resultantes. El mensaje que se procederá a ocultar será una parte del primer capítulo del Quijote. En la siguiente imagen se puede ver una muestra del mensaje en cuestión:

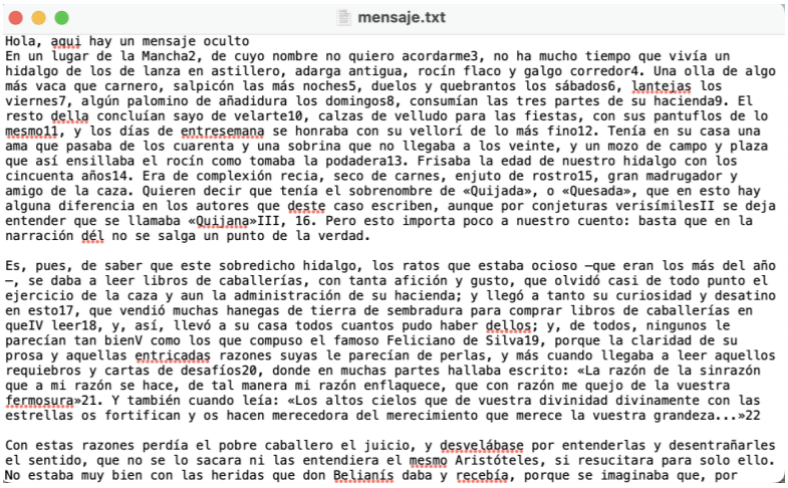


ILUSTRACIÓN 11: MENSAJE OCULTO

Para ocultar este mensaje se usará un proyecto de PyPI, repositorio de software del que pueden hacer uso diversas aplicaciones y que está elaborado en el lenguaje de programación Python. El proyecto en cuestión es “Steganography” y se usará el método LSBSteg para ocultar un archivo en la información del color de una imagen RGB. Más detalle sobre esto en la siguiente imagen:

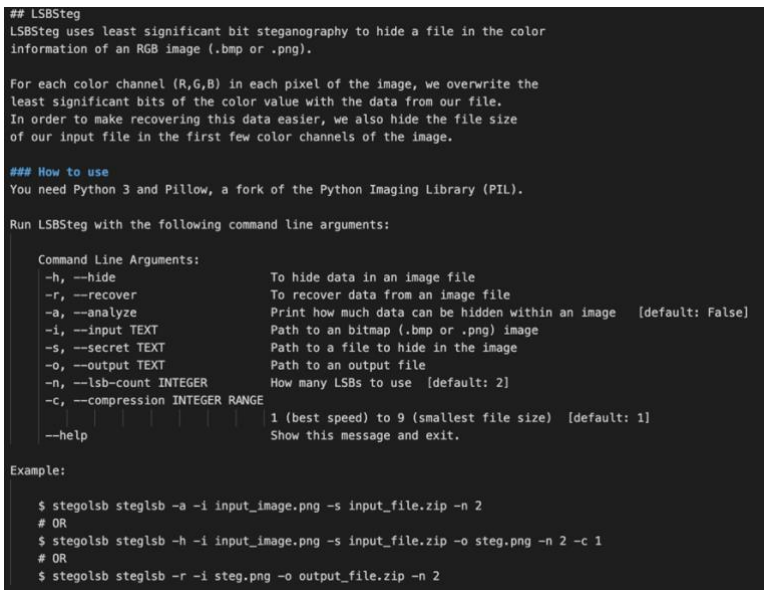


ILUSTRACIÓN 12: LSBSTEG

Además, se creará un programa para que automáticamente oculte el mensaje en todas las imágenes de la carpeta “imagenespng” y no tener que ejecutar ese comando imagen por imagen. Se puede ver más detalle sobre esto en imgtostego.py.

```
alexiacazon@MacAlexia Desktop % python3 imgtostego.py
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
Files read                      in 0.00s
5292 bytes hidden               in 0.00s
Image overwritten               in 0.00s
```

ILUSTRACIÓN 13: OCULTACIÓN DE MENSAJE EN LAS IMÁGENES

Y se obtendrá una carpeta con las estegoimágenes resultantes.

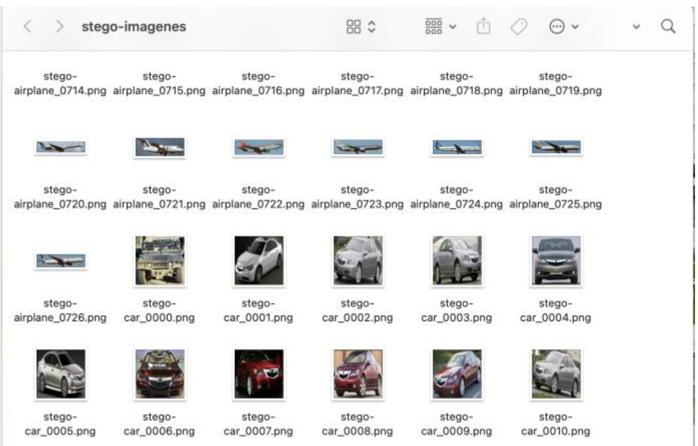


ILUSTRACIÓN 14: ESTEGOIMÁGENES RESULTANTES

PRINCIPALES MÉTODOS DE ESTEGOANÁLISIS

Los principales métodos de estegoanálisis incluyen el análisis visual, el análisis estadístico, el análisis de características, el análisis de frecuencia y el análisis de entropía. Cada uno de estos métodos tiene sus puntos positivos y negativos, y se utilizan en diferentes contextos y escenarios de estegoanálisis. En general, la combinación de múltiples métodos de estegoanálisis suele ser la más efectiva para detectar la presencia de información oculta en los datos.

POV

Tras estudiar los métodos estenográficos anteriores se puede concluir que un método efectivo para detectarlos es el análisis PoV, “Pair of Vales”, o en español, el análisis por par de valores. Se basa en que, al ocultar un mensaje en una imagen, usando el método de LSB o el del dominio de la frecuencia, se modifica el último bit de cada byte. Al hacer esto si se agrupan los 256 valores resultantes por pares contiguos ([0, 1], [2, 3], [4, 5], …, [252, 253], [254, 255]), sigue estando dentro del mismo par que antes de aplicarle la esteganografía, independientemente del nuevo valor del último bit. Veamos un ejemplo:

100 píxeles con valor 4

50 píxeles con valor 5

150 píxeles en el par [4, 5]

Se aplica el método esteganográfico y los cambios en los valores de los píxeles quedan así:

75 píxeles con valor 4

75 píxeles con valor 5

150 píxeles en el par [4, 5]

Por lo tanto, la distribución total de píxeles es constante en los pares de valores. Aplicado a la práctica, con la imagen que queremos estudiar se puede calcular la frecuencia que se espera de los valores de los distintos pares de píxeles, característica fundamental para poder realizar una hipótesis de tipo Chi cuadrado.

Primero, se calcula el valor Chi cuadrado con la siguiente fórmula:

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

Con el resultado se calcula la probabilidad de que se haya ocultado un mensaje, es decir, el p-valor. Si el resultado de p es 0 o tiende a 0, significa que las barras son muy dispares las unas a las otras, lo que conlleva a suponer que existe la posibilidad de que haya información oculta en la imagen, en cambio, si tiende a 1, las barras de aparición son semejantes.

Se pondrá a prueba este método con la siguiente imagen original:

ILUSTRACIÓN 15: IMAGEN ORIGINAL PARA POV

La imagen resultante de añadirle un mensaje oculto es la siguiente que se ve a continuación:

ILUSTRACIÓN 16: ESTEGO IMAGEN PARA POV

Como se puede observar, ante la mirada humana es imposible ver las diferencias entre ambas. Se crea un script de Python para detectar la esteganografía mediante el método de PoV que puede encontrarse como PoV.py en el GitHub y el resultado es el siguiente:

```
alexiacazon@MacAlexia Desktop % python3 PoV.py /Users/alexiacazon/Desktop/stego-imagenes/stego-airplane_0000.png
p-valor 0.0
La imagen podría contener esteganografía.
```

ILUSTRACIÓN 17: EJECUCIÓN POV

Al ser el p-valor igual a 0 este método afirma que es muy probable que la imagen tenga información oculta.

EXIFTOOL

ExifTool es un programa gratis y de colaboración abierta para leer, escribir y modificar metadatos de imágenes, audios, videos y otros ficheros. Servirá para detectar mensajes ocultos en los metadatos de una imagen.

Se hará uso de la siguiente imagen para ocultar información aprovechando el formato de la imagen:

ILUSTRACIÓN 18: IMAGEN ORIGINAL PARA EXIFTOOL

Con esta herramienta se pueden leer los metadatos de esta imagen que de otra manera no podríamos:

```
alexiacazon@MacAlexia ~ % exiftool /Users/alexiacazon/Desktop/gato1.jpg
ExifTool Version Number         : 12.56
File Name                       : gato1.jpg
Directory                       : /Users/alexiacazon/Desktop
File Size                       : 79 kB
File Modification Date/Time     : 2019:10:05 03:02:56+02:00
File Access Date/Time           : 2023:02:21 12:51:00+01:00
File Inode Change Date/Time     : 2023:02:21 12:50:58+01:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 300
Y Resolution                    : 300
Exif Byte Order                 : Big-endian (Motorola, MM)
Make                            : Canon
Camera Model Name               : Canon EOS 80D
Exposure Time                   : 1/160
F Number                        : 5.6
ISO                             : 160, 0
Date/Time Original              : 2018:10:19 13:34:20
Flash                           : Unknown (16 0)
Focal Length                    : 103.0 mm
Lens Model                      : TAMRON 16-300mm F/3.5-6.3 Di II VC PZD B016
Image Width                     : 640
Image Height                    : 426
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Aperture                        : 5.6
Image Size                      : 640x426
Megapixels                      : 0.273
Shutter Speed                   : 1/160
Focal Length                    : 103.0 mm
Light Value                     : 11.6
Lens ID                         : TAMRON 16-300mm F/3.5-6.3 Di II VC PZD B016
```

ILUSTRACIÓN 19: EXIFTOOL

Ahora se procede a ocultar un comentario en la imagen, una de las muchas opciones de exiftool y se obtendrán los siguientes resultados:

```
alexiacazon@MacAlexia ~ % exiftool -comment="esto es un mensaje oculto" /Users/alexiacazon/Desktop/gato1.jpg
    1 image files updated
alexiacazon@MacAlexia ~ % exiftool /Users/alexiacazon/Desktop/gato1.jpg
ExifTool Version Number         : 12.56
File Name                       : gato1.jpg
Directory                       : /Users/alexiacazon/Desktop
File Size                       : 79 kB
File Modification Date/Time     : 2023:02:21 12:54:40+01:00
File Access Date/Time           : 2023:02:21 12:54:42+01:00
File Inode Change Date/Time     : 2023:02:21 12:54:40+01:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 300
Y Resolution                    : 300
Exif Byte Order                 : Big-endian (Motorola, MM)
Make                            : Canon
Camera Model Name               : Canon EOS 80D
Exposure Time                   : 1/160
F Number                        : 5.6
ISO                             : 160, 0
Date/Time Original              : 2018:10:19 13:34:20
Flash                           : Unknown (16 0)
Focal Length                    : 103.0 mm
Lens Model                      : TAMRON 16-300mm F/3.5-6.3 Di II VC PZD B016
Comment                         : esto es un mensaje oculto
Image Width                     : 640
Image Height                    : 426
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Aperture                        : 5.6
Image Size                      : 640x426
Megapixels                      : 0.273
Shutter Speed                   : 1/160
Focal Length                    : 103.0 mm
Light Value                     : 11.6
Lens ID                         : TAMRON 16-300mm F/3.5-6.3 Di II VC PZD B016
```

ILUSTRACIÓN 20: RESULTADOS EXIFTOOL

APLICANDO ESTEGOANÁLISIS A UNA BASE DE DATOS DE ESTEGOIMÁGENES

Para este apartado se hará uso de las mismas carpetas que se introdujeron en el apartado de Aplicando esteganografía a una base de datos de imágenes. En este caso, haremos uso del mismo proyecto pero se ejecutará StegDetect, que proporciona un método para detectar esteganografía simple en imágenes.

```
<a name = "StegDetect"></a>
## StegDetect
StegDetect provides one method for detecting simple steganography in images.

### How to Use
You need Python 3 and Pillow, a fork of the Python Imaging Library (PIL).

Run StegDetect with the following command line arguments:

    Command Line Arguments:
    -i, --input TEXT         Path to an image
    -n, --lsb-count INTEGER  How many LSBs to display  [default: 2]
    --help                   Show this message and exit.
```

ILUSTRACIÓN 21: STEGDETECT

Ejecución:

```
alexiacazon@MacAlexia stego_lsb % stegolsb stegdetect -i /Users/alexiacazon/Desktop/stego-imagenes/stego-airplane_0000.png
Runtime: 0.02s
```

ILUSTRACIÓN 22: EJECUCIÓN STEGDETECT

Se guardará una imagen con los pixeles que contienen un mensaje oculto. En nuestro caso el resultado fue el siguiente:

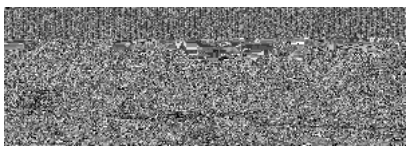


ILUSTRACIÓN 23: PÍXELES CON MENSAJE OCULTO

CREACIÓN DE LA HERRAMIENTA

La herramienta que se propone en esta memoria resulta de un conjunto de los métodos estudiados en los primeros capítulos más una parte fundamental que será escoger las características que tendrá en cuenta nuestro algoritmo de clasificación para decidir si la imagen esconde o no contenido malicioso.

CONCEPTOS PREVIOS

Para poder decidir qué características se tendrán en cuenta se debe introducir primero varios conceptos.

FUNCIÓN CARACTERÍSTICA

Una función de masa de probabilidad calcula la probabilidad de que una variable que debe ser aleatoria discreta tome cierto valor. Es decir, a cada punto de su dominio lo corresponde con la probabilidad de que tenga cierto valor. A partir de ahora nos referiremos a la función de masa de probabilidad como FMP.

Cuando se agrega el mensaje oculto a una imagen se está modificando cada píxel de ésta, este grado de modificación será una variable aleatoria discreta con su FMP correspondiente f_Δ. Si $h_o$es el histograma de la imagen original y $h_s$el histograma de la stego imagen, se tiene la siguiente relación:

$$h_s = h_o * f_\Delta$$

Sea i el píxel y f_i el componente de frecuencia en ese píxel se tiene la siguiente relación:

$$H_s[f_i] = DFT(h_s[i]) = H_o[f_i] * F_\Delta[f_i]$$

Así, la función característica, FC, será el conjugado de la transformada discreta de Fourier DFT del histograma de la imagen original, $|H(f_i)|$, y gracias a ella se podrá aplicar métodos analíticos más adelante.

TRANSFORMADA DE WAVELET DISCRETA

Una Wavelet es una pequeña onda o señal de duración limitada, lo que significa que su actividad está concentrada alrededor de un punto y tiene promedio cero:

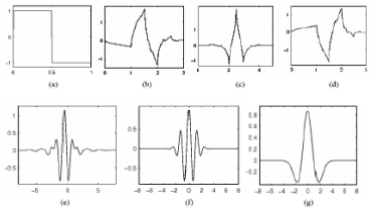


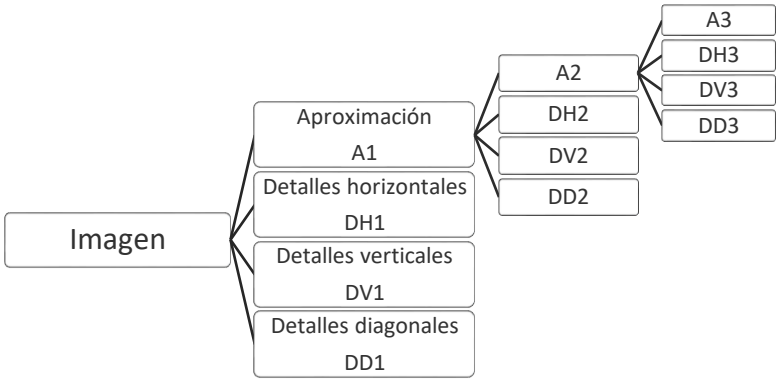
ILUSTRACIÓN 24: WAVELET

La Transformada de Wavelet Discreta, conocida por sus siglas DWT, transforma un vector de información de longitud n $(a_1, a_2, \dots, a_n)$ en otro vector de coeficientes wavelets de la misma longitud $(c_1, c_2, \dots, c_n)$, usando las funciones que hemos introducido antes como wavelets. Si se calcula la DWT de una imagen se tendrá como resultado un conjunto de coeficientes DWT que al clasificarlos por las sucesivas bandas permitirán que mejore la resolución de la imagen.

Con otras palabras, dada una imagen, si se le aplica la DWT, lo que se está haciendo es ir aplicando filtros a la imagen original creando componentes de ésta de resolución inferior. Cada filtro permite fraccionar la imagen en coeficientes con información sobre los de detalles horizontales, verticales y

diagonales. A partir de estos coeficientes, se puede reconstruir la imagen original. El diagrama sería algo así:

TABLA 6: COEFICIENTES DWT

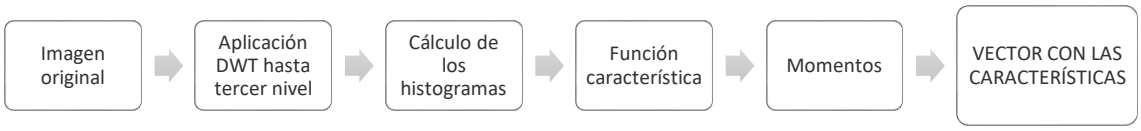


Además, se sabe que estos coeficientes DWT son independientes, lo que permite hacer uso de ellos para la obtención de características de una imagen. Otro término del que se hará uso son los momentos de una variable, que en estadística se refieren a los valores esperados de ciertas funciones de esa variable y son definidos de la siguiente manera:

$$M_n = \frac{\sum_{j=1}^{\frac{N}{2}} f_j^n |H(f_i)|}{\sum_{j=1}^{\frac{N}{2}} |H(f_i)|}$$

Así, se calcularán los momentos para $n = 1,2,3$ de la función característica de las sub-bandas obtenidas en la DWT y de ahí se obtendrán las características finales. El esquema final sería el siguiente:

TABLA 7: ESQUEMA OBTENCIÓN DE CARACTERÍSTICAS



OBTENCIÓN DE LAS CARACTERÍSTICAS DE LAS IMÁGENES

Este programa servirá para calcular los momentos de una imagen hasta el nivel 3 y para almacenarlos en un csv.

En el GitHub se pueden encontrar como momentos.py y stegomomentos.py

Se usarán para crear un csv con los momentos de un conjunto de imágenes y otro csv con el conjunto de estegoimágenes.

Tras hacer esto, se juntarán ambos csv, el de imágenes y el de estegoimágenes, en uno, que será la base de datos final para nuestro algoritmo de clasificación. A continuación, se pueden ver los momentos calculados para una pequeña muestra inicial de imágenes:

	m00	m10	m01	...	mu02	mu21	mu12
0	4.0	170.000000	12088.000000	...	1.000000	3.074092e-09	0.000000e+00
1	2.0	2520.000000	1914.000000	...	0.333333	2.970919e-07	9.778887e-08
2	2.0	2524.000000	1906.000000	...	0.333333	1.369044e-07	-6.180163e-08
3	2.0	2076.000000	1672.000000	...	0.333333	3.390814e-07	8.055940e-08
4	2.0	4.000000	1914.000000	...	0.333333	1.136868e-12	0.000000e+00
5	5.5	69.833333	2170.833333	...	2.578283	1.519896e-01	1.519896e-01
6	223.0	113879.500000	181212.833333	...	3464.372073	-3.882562e+03	-5.618876e+02
7	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
8	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
9	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
10	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
11	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
12	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
13	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
14	2.0	224.000000	1062.000000	...	0.333333	6.439222e-10	1.117587e-08
15	2.0	30.000000	1096.000000	...	0.333333	6.821210e-13	1.164153e-09
16	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
17	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
18	2.0	30.000000	1096.000000	...	0.333333	6.821210e-13	1.164153e-09
19	4.0	824.000000	1862.000000	...	1.666667	0.000000e+00	1.792796e-08
20	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
21	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
22	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
23	2.0	364.000000	846.000000	...	0.333333	2.051820e-09	1.047738e-09
24	2.0	594.000000	1800.000000	...	0.333333	1.746230e-08	2.305023e-08
25	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
26	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
27	14.0	23653.500000	42278.500000	...	13.339286	1.371207e+00	1.821045e+00
28	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
29	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
30	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
31	0.0	0.000000	0.000000	...	0.000000	0.000000e+00	0.000000e+00
32	1.0	12.000000	832.333333	...	0.055556	-2.222222e-02	-4.656613e-10
33	2.0	3582.000000	5722.000000	...	0.333333	2.159737e-06	2.223998e-06

ILUSTRACIÓN 25: CARACTERÍSTICAS DE UNA MUESTRA DE IMÁGENES

ALGORITMO DE CLASIFICACIÓN

Dentro del Machine Learning existen diversos algoritmos, pero para problema planteado se hará uso de los algoritmos de clasificación. Estos algoritmos encuentran patrones en los datos que se les da y los clasifican en grupos para luego comparar nuevos datos y ubicarlos en cada grupo. En el problema planteado se distinguen dos grupos: libre de contenido oculto o no. Como se ha visto en el apartado anterior para cada técnica de ocultación conocida se puede generar una técnica de detección y lo que interesa es combinar estos modelos de detección en uno y así la herramienta mejorará la precisión.

Aquí entran en juego combinar varios métodos, esto ayuda a mejorar la eficacia de los modelos de Machine Learning. Se basan en un proceso mediante el cual se construyen varios modelos de Machine Learning para resolver un problema particular.

Uno de los algoritmos más usados dentro de los métodos combinados es el de los bosques aleatorios. Para entender este término se introducirán primero los árboles de decisión.

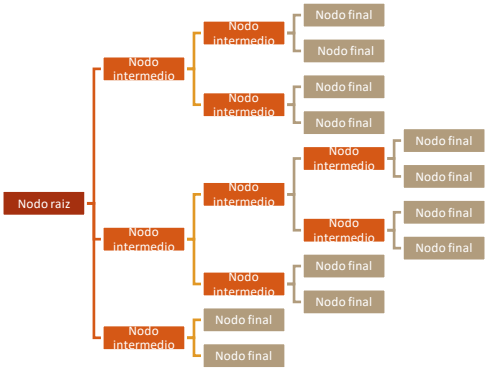
ÁRBOLES DE DECISIÓN

Los componentes de un árbol de decisión son los siguientes:

- Nodo raíz: Decisión que se intenta tomar, en el caso propuesto sería una imagen y a partir de ella comprobar si tiene algún contenido oculto.
- Nodo intermedio: Pregunta sobre una observación que indaga más en detalle, las únicas respuestas válidas son sí o no, por lo que cada nodo intermedio se dividirá en dos a su vez.

- Rama: Conexión que une el nodo raíz con los nodos intermedios y también une estos últimos con otros dos hasta los nodos finales.
- Nodo final: Son los nodos cuya función es indicar la clasificación definitiva, en el caso propuesto libres de contenido oculto o no.

TABLA 8: ARBOLES DE DECISIÓN



Estos modelos se generan por reglas binarias (en nuestro caso responderían a las observaciones con un si o un no) con las que se podrá responder a las observaciones en función de las características y predecir así el valor final que tomará (en nuestro libre de contenido oculto o no). Se puede ver un ejemplo simple para entenderlo mejor a continuación:

TABLA 9: ÁRBOL DE DECISIÓN APLICADO A ESTEGANOGRAFÍA

Así, un bosque aleatorio está formado por la creación de varios árboles de decisión y los combina para obtener una predicción más precisa.

ÁRBOLES DE DECISIÓN EN PYTHON

Para desarrollar nuestro árbol de decisión se definirán dos tipos de variables. La variable objetivo o destino, que es la que predice el resultado libre de contenido oculto o no, que en el programa se definirá como ‘output’ y podrá valer ‘0’ o ‘1’ respectivamente. Y las variables predictoras, que se corresponden con las características de las imágenes y las obtendremos de un csv que contendrá los momentos de diversas imágenes y estegoimágenes. Por lo tanto, serán los valores respectivos de los momentos 'm00', 'm10', 'm01', 'm20', 'm11', 'm02', 'm21', 'm12', 'mu20', 'mu11', 'mu02', 'mu21', 'mu12'.

Se definirán así dos matrices: X, con las características de las imágenes, e Y, con los valores del ‘output’.

TABLA 10: MOMENTOS DE LAS IMÁGENES

id	m00	m10	...	mu21	mu12
0	4	170	...	3.074092e-09	0
1	2	2520	...	2.970919e-07	9.778887e-08
2	2	2524	...	1.369044e-07	-6.100163e-08
...	...	...	...	...	...

TABLA 11: OUTPUT HERRAMIENTA

id	output
0	1
1	1
2	1
...	...

El algoritmo que se usará para ir particionando nuestro árbol hasta el nodo final será la entropía. La entropía mide la incertidumbre de una fuente de información y puede ser considerada como una medida de esta que nos sirve para reducir o eliminar la incertidumbre. En general, una característica que puede ayudar a discriminar más objetos, en nuestro caso imágenes, tiende a reducir más la entropía, y por tal motivo, debe ser seleccionado como un nodo intermedio para la siguiente subdivisión. Así, se buscan los árboles que tengan más pequeña la entropía en sus nodos intermedios. Para calcularla se usa la siguiente fórmula:

$$Entropia(S) = \sum_{i=1}^{n} -p_i \log_2 p_i$$

Donde S es la colección de imágenes, p_i la probabilidad de los posibles valores e i el posible output de las imágenes.

El concepto opuesto a la entropía es la ganancia de información, si la entropía disminuye, la ganancia de la información aumenta, y viceversa. Por lo tanto, la

construcción de un árbol de decisión se trata de encontrar características que devuelvan la mayor cantidad de información.

Por otro lado, se definen dos conjuntos de datos, un conjunto de entrenamiento, con el que el algoritmo aprenderá y un conjunto de prueba con el que el algoritmo decidirá si la imagen tiene contenido oculto o no.

Se usará la librería scikit para la construcción del árbol de decisión con el método DecisionTreeClassifier. Este, toma las dos matrices que se han definido anteriormente como entrada. El objetivo es crear un árbol de clasificación que prediga el valor de la variable 'output' gracias al aprendizaje que irá adquiriendo con las tomas de decisiones que se deducen de las características de las imágenes haciendo uso de la entropía.

Se puede encontrar el programa correspondiente en el GitHub como arbol.py.

RESULTADOS

En esta sección por un lado se analizarán y evaluarán los resultados del algoritmo de clasificación con dos muestras diferenciadas: una inicial con un número pequeño de imágenes y otra final mucho más grande y por lo tanto, fidedigna. Y por otro lado, se finalizará con los resultados generales de la herramienta.

RESULTADOS ALGORITMO DE CLASIFICACIÓN

El árbol se irá subdividiendo hacia la izquierda, si la condición se cumple y hacia la derecha en caso contrario. La información que contiene cada nodo es la siguiente:

- Condición: en nuestro caso relacionado con el valor de cierto momento, aparece solo si es un nodo intermedio donde se toma alguna decisión.
- Entropía: como medida de incertidumbre que se ha explicado anteriormente.
- Samples: número de muestras, en el caso propuesto, imágenes que satisfacen las condiciones necesarias para llegar a este nodo.
- Value: cuántas muestras de las distintas clases llegan a este nodo, [<muestras con output='0'>, <muestras con output='1'>].
- Class: '0' si no tiene contenido oculto y '1' en caso contrario.

Se ha probado la herramienta para una primera pequeña muestra inicial para más tarde extrapolarla a una base mucho más grande de imágenes. El conjunto de imágenes y estegoimágenes son los siguientes:

ILUSTRACIÓN 26: MUESTRA INICIAL IMÁGENES ORIGINALES

ILUSTRACIÓN 27: MUESTRA INICIAL ESTEGOIMÁGENES

Con estos datos el árbol de decisión resultante es el siguiente:

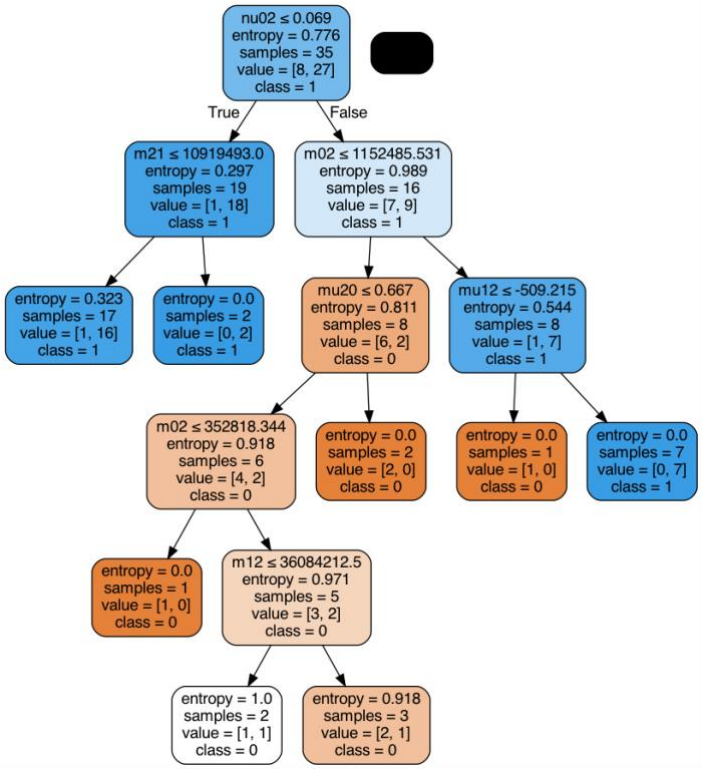


ILUSTRACIÓN 28: ÁRBOL DE DECISÓN MUESTA INICIAL

Y si se imprime por pantalla los momentos de las imágenes a predecir, su predicción y la exactitud, los resultados son los siguientes:

```
ids
      m00          m10          m01          m20          m11  ...      nu02          nu30          nu21          nu12          nu03
22    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
2     2.0  2524.000000  1906.000000  3.185288e+06  2.405372e+06  ...  0.083333  1.685874e-07  2.420151e-08 -1.078367e-08  4.214685e-08
49    5.5  9848.833333  15733.833333  1.763628e+07  2.817453e+07  ...  0.085232 -4.271889e-03  2.142341e-03  2.142150e-03 -4.271835e-03
26    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
33    2.0  3582.000000  5722.000000  6.415362e+06  1.024810e+07  ...  0.083333  3.371748e-07  3.817912e-07  3.931511e-07  1.348699e-06
44  157.0  163094.666667  130328.166667  1.694380e+08  1.353906e+08  ...  0.048571 -5.901611e-02 -9.791995e-03 -1.586116e-04  5.093308e-04
30    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
50    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
32    1.0    12.000000   832.333333  1.441667e+02  9.988000e+03  ...  0.055556  4.547474e-13 -2.222222e-02 -4.656613e-10  7.407427e-03
27   14.0  23653.500000  42278.500000  3.996346e+07  7.143104e+07  ...  0.068058 -2.725397e-03  1.869747e-03  2.483136e-03  7.564428e-04
3     2.0  2076.000000  1672.000000  2.154888e+06  1.735536e+06  ...  0.083333  1.685874e-07  5.992755e-08  1.424102e-08  4.214685e-08
29    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
47    2.0    84.000000  6044.000000  3.528333e+03  2.538480e+05  ...  0.083333  5.144879e-12  2.717139e-10  9.219623e-09  1.348699e-06
41    2.0  1054.000000  1816.000000  5.554583e+05  9.570320e+05  ...  0.083333  2.107342e-08  1.596970e-08  2.057952e-10  4.214685e-08
39    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
21    0.0     0.000000     0.000000  0.000000e+00  0.000000e+00  ...  0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00

[16 rows x 21 columns]
prediccion [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
Accuracy: 0.8125
```

ILUSTRACIÓN 29: PREDICCIONES MUESTRA INICIAL DE IMÁGENES

Podemos verlo ahora para una muestra de 1696 imágenes y otras tantas con mensaje oculto. Las muestras finales del conjunto de imágenes y estegoimágenes son las siguientes:

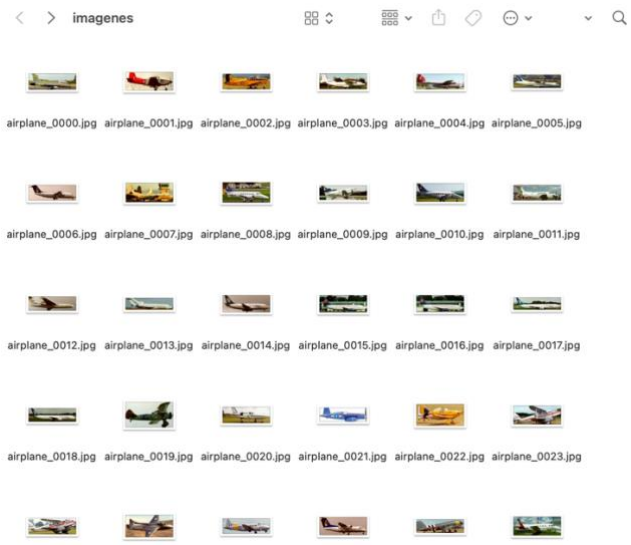


ILUSTRACIÓN 30: MUESTRA FINAL IMÁGENES ORIGINALES

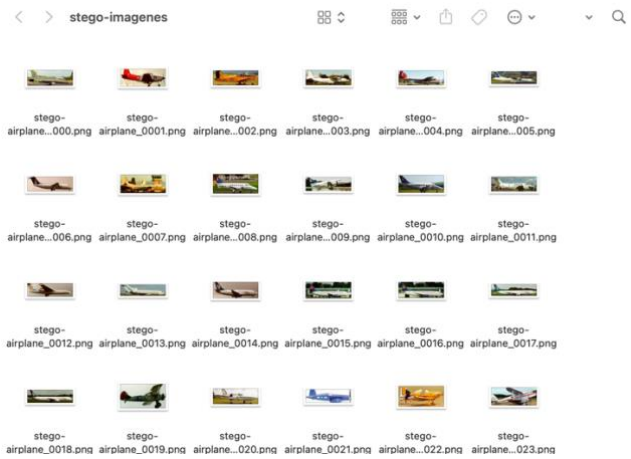


ILUSTRACIÓN 31: MUESTRA FINAL ESTEGOIMÁGENES

Con estos datos el árbol de decisión resultante es el siguiente:

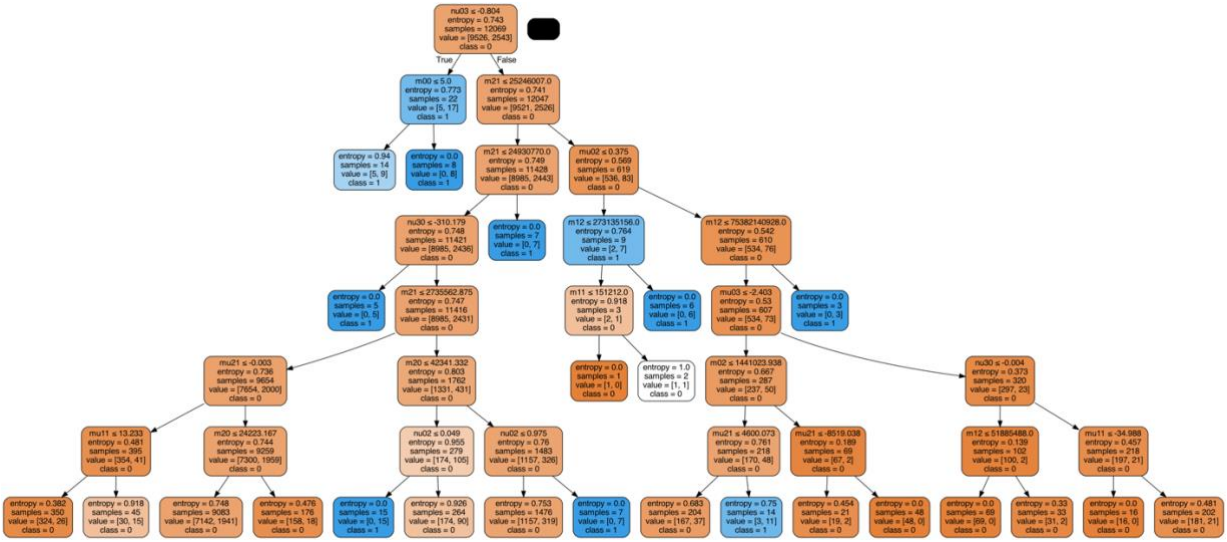


ILUSTRACIÓN 32: ÁRBOL DE DECISIÓN MUESTRA FINAL

En este caso, la cantidad de datos con las que se entrena el algoritmo es mayor por lo que se puede hacer más profundo el árbol. Los resultados de los momentos de las imágenes a predecir, su predicción y la exactitud son los siguientes:

```
alexiacazon@MacAlexia Desktop % python3 arbol.py
ids
          m00          m10          m01          m20  ...          nu30          nu21          nu12          nu03
13669   0.0     0.000000     0.000000     0.000000  ...  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
15564   0.0     0.000000     0.000000     0.000000  ...  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
1546    0.0     0.000000     0.000000     0.000000  ...  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
12099   0.0     0.000000     0.000000     0.000000  ...  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
12545   2.0   190.000000   194.000000   18050.333333  ...  4.115903e-11  7.074209e-12  3.408482e-11  8.231806e-11
...     ...          ...          ...          ...  ...           ...           ...           ...           ...
10049   0.0     0.000000     0.000000     0.000000  ...  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
4547    2.0   334.000000   230.000000   55778.333333  ...  3.292723e-10  4.373147e-11  9.067849e-11  8.231806e-11
8775   84.5  5914.833333  7662.000000  415891.750000  ... -3.853093e-02 -7.669246e-03  3.209467e-03  1.612141e-03
8562    1.5    61.833333   119.166667    2549.250000  ... -1.627766e-02 -1.037888e-02  1.418695e-03  2.837391e-03
8589    5.5    34.000000   503.333333     217.416667  ... -2.226223e-02 -1.012408e-02  1.645229e-01 -3.247730e+00

[5173 rows x 21 columns]
prediccion [0 0 0 ... 0 0 1]
Accuracy: 0.7858109414266383
```

ILUSTRACIÓN 33: PREDICCIÓN MUESTRA FINAL DE IMÁGENES

RESULTADOS FINALES DE LA HERRAMIENTA

Se finaliza este capítulo de la memoria resumiendo los resultados de los principales métodos usados para la herramienta:

1. PoV: método que permite detectar esteganografía de tipo LSB y de dominio de la frecuencia. En general se considera una técnica bastante efectiva para detectar la presencia de información oculta en imágenes.
2. Exiftool: técnica que permite detectar mensajes ocultos en los metadatos de las imágenes, ofrece resultados reales y acordes con la información encubierta pues en general muestra tal cual el mensaje oculto.
3. StegDetect: en general, la librería StegDetect se considera una herramienta bastante efectiva para detectar la presencia de información oculta en imágenes mediante el análisis de patrones estadísticos, aunque su precisión exacta puede variar en diferentes casos de uso.
4. Momentos característicos de la imagen: mediante el análisis de ciertas características estadísticas de la imagen se ha podido entrenar un árbol de decisión que ha arrojado resultados de alrededor al 80% de precisión.

Es importante tener en cuenta que ninguna herramienta de esteganálisis puede detectar el 100% de los casos de información oculta, ya que siempre existe la posibilidad de que los algoritmos de esteganografía se hayan diseñado para evadir la detección. Por lo tanto, esta herramienta usa varios métodos de esteganálisis y técnicas de análisis estadístico para mejorar la precisión en la detección de información oculta.

CONCLUSIONES

Tras un proceso de investigación sobre métodos y técnicas de esteganografía y estegoanálisis se ha sido capaz de ponerlos a prueba y conocer cómo funcionan. Esto ha servido durante la memoria para tener en cuenta cómo se ejecuta hoy en día esta práctica y para tomar estas ideas en la creación de la herramienta posteriormente. El machine learning ha sido la tecnología perfecta para poder desarrollarla y hacerla capaz de detectar si una imagen se trata realmente de la original o una modificada. Basándose en un aprendizaje continuo, entrenándose con una pequeña muestra primero y otra mucho más grande después y con capacidades de mejora indefinida. El modelo más visual de esta forma de inteligencia artificial han sido los árboles de decisión que nos han dado la respuesta a la incógnita de este trabajo: ¿Cómo detectar mensajes ocultos en imágenes?

Con este método basado en inteligencia artificial y junto con las técnicas de estegoanálisis más relevantes se ha conseguido lograr eficiencia y precisión además de una herramienta de fácil uso por parte de técnicos de los distintos sectores de la ciberseguridad.

La esteganografía plantea importantes desafíos para la seguridad y la privacidad de la información y junto con el esteganálisis son áreas en constante evolución y que los investigadores están desarrollando continuamente para mejorar la ciberseguridad. La conciencia y la comprensión de estas ciencias son importantes para mantener la integridad y la confidencialidad en una amplia variedad de contextos.

BIBLIOGRAFÍA

Berg, G., Davidson, I., Duan, M.-Y., & Goutam, P. (2003). Searching For Hidden Messages: Automatic Detection of Steganography. *ACTAS DE LA XV CONFERENCIA DE APLICACIONES INNOVADORAS DE LA INTELIGENCIA ARTIFICIAL* (págs. 51-56). Albany: John Riedl y Randy Hill.

Bismita Choudhury, R. D. (2015). *A Novel Steganalysis Method Based on Histogram Analysis.* Obtenido de Department of Computer Science and Engineering and Information Technology, Don Bosco College of Engineering and Technology, Guwahati 781017, Assam, India: https://www.researchgate.net/publication/282889667_A_Novel_Steganalysis_Method_Based_on_Histogram_Analysis?enrichId=rgreq-aa9350ee496d59932e25e8f9cef8a7b0-XXX&enrichSource=Y292ZXJQYWdlOzI4Mjg4OTY2NztBUzo2MTQ1MjI4ODA1NDQ3NzVAMTUyMzUyNTA4MDczMQ%3D%3D&el=1_x

Carlos L. Velasco-Bautista , Julio C. López-Hernánde, Mariko Nakano-Miyatake, & Héctor M. Pérez-Meana. (2007). *Esteganografía en una imagen digital en el dominio DCT.* Obtenido de Instituto Politécnico Nacional: https://www.redalyc.org/pdf/614/61411403.pdf

Center for Statistics and Applications in Forensic Evidence and Center for Survey Statistics and Methodology, Iowa State University. (Marzo de 2021). *Stego App DB*. Obtenido de Stego App DB: https://data.csafe.iastate.edu/StegoDatabase/

Chamorro, A. G. (2010). *Estegoanálisis en imágenes digitales.* México: Instituto Politécnico Nacional.

Diaz, J. (2015). *GitHub.* Obtenido de GitHub: https://github.com/INTECOCERT/Varios

Diaz, J. (2015). *Github-INTECOCERT.* Obtenido de Github: https://github.com/INTECOCERT/Varios/blob/master/stego/imgchi2.py

Díaz, J. (2015). *INCIBE-CERT*. Obtenido de INCIBE: https://www.incibe-cert.es/blog/esteganografia-y-estegoanalisis-basicos

G., F., M., S., J.L., F. M., & S., D.-C. (2004). Determinación de parámetros de la transformada de Wavelet para la clasificación de señales del diagnóstico scattering thomson. *XXV Jornada de Automática.* Ciudad Real.

Hamid, N., Yahya, A., Ahmad, R., & Al-Qershi, O. (2012). Técnicas de esteganografía de imágenes: una visión general. *Revista Internacional de Ciencias de la Computación y Seguridad*, Volume 6, Issue 3.

Hernández, J. C. (2009). *Estegoanálisis de imágenes digitales usando técnica de reconocimiento de patrones.* México: Instituto Politécnico Nacional.

Hostalot, D. L. (2015-2023). *Esteganografía LSB en imágenes y audio*. Obtenido de https://daniellerch.me/stego/lab/intro/lsb-es/

Jan Kodovský, J. F. (s.f.). *Department of Electrical and Computer Engineering, Binghamton University, NY, 13902, USA.* Obtenido de http://dde.binghamton.edu/kodovsky/pdf/TIFS-2011-ensemble.pdf

jessica0x73. (2019). *An Investigation into Steganalysis Techniques in Media File Forensics and the Use of Machine Learning in Identifying Affected Files*. Obtenido de GitHub: https://github.com/jessica0x73/steganalyse

Kodovský , J., Holub, V., & Fridrich, J. (2011). Clasificadores de conjunto para estegoanálisis de medios digitales. *Transacciones IEEE sobre análisis forense y seguridad de la información*, 432-444.

Lerch Hostalot , D. (s.f.). *daniellerch.me*. Obtenido de https://daniellerch.me/stego/lab/intro/lsb-es/

Liua, Q., Sung, A., Chen, Z., & Xu, J. (2008). Feature mining and pattern classification for steganalysis of LSB matching steganography in grayscale images. *Pattern Recognition 41*, 56-66.

Nieto, N. &. (2008). El uso de la transformada wavelet discreta en la reconstrucción de señales senosoidales. *Scientia et technica, 14(38)*, 381-386.

Perez, M. R. (2013). *Blind Steganalysis Method for Detection of Hidden Information in Images.* Obtenido de https://inaoe.repositorioinstitucional.mx/jspui/bitstream/1009/243/1/RodriguezPM.pdf

Qian, Y., Dong, J., Wang, W., & Tan, T. (2016). Aprendizaje y transferencia de representaciones para estegoanálisis de imágenes utilizando redes neuronales convolucionales. *Conferencia internacional IEEE de 2016 sobre procesamiento de imágenes*, (págs. 2752-2756). China.

T.S.R. Krishna Prasad, Y.V.N. Tulasi, & V. Ra. (2011). *Disclosure of Hidden Messages using Classifier based on Statistical Moments of Wavelet Characteristic Function.* Obtenido de International Journal of Electronics and Communication Engineering: https://www.ripublication.com/irph/ijece/ijecev4n2__10.pdf

Vahid, M., & Raschka, S. (2020). *Python machine learning.* Marcombo.