Algoritmo de Detección de Manipulaciones Inter-Fotogramas en Vídeos Digitales

Inter-frame Manipulation Detection Algorithm in Digital Videos

TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2020–2021

Alina Burachok

Directores
Luis Javier García Villalba
Ana Lucila Sandoval Orozco

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Madrid, Septiembre de 2021

Agradecimientos

A mi madre y mis amigos, por apoyarme cuando más lo necesitaba.
A Ana, por su paciencia y tiempo.

Índice general

Índice de figuras VII

Lista de Acrónimos X

Abstract XII

Resumen XIII

1. Introducción 1
- 1.1. Motivación . . . 1
- 1.2. Contexto . . . 1
- 1.3. Objeto de la Investigación . . . 1
- 1.4. Plan de Trabajo . . . 2
- 1.5. Estructura del Trabajo . . . 4

2. Marco Conceptual 5
- 2.1. Proceso de Compresión . . . 5
- 2.2. Formatos de Compresión . . . 6
  - 2.2.1. Compresión MPEG . . . 6
  - 2.2.2. Compresión H.264 . . . 6
  - 2.2.3. Compresión H.265 . . . 7
  - 2.2.4. Diferencias entre Compresión H.264 y H.265 . . . 7
- 2.3. Estructura de Información en H.264 . . . 8
  - 2.3.1. Cadenas de Bits de Datos . . . 10
- 2.4. Redes Neuronales . . . 10
  - 2.4.1. Tipos de Redes Neuronales . . . 11
  - 2.4.2. Definición de una Neurona . . . 12
- 2.5. Redes Neuronales Convolucionales . . . 12

3. Técnicas de Manipulación en Vídeo 15
- 3.1. Técnicas de Manipulación Intrafotograma . . . 16
- 3.2. Técnicas de Manipulación Interfotograma . . . 16
- 3.3. Comparación de Trabajos Anteriores . . . 17
  - 3.3.1. Trabajos sobre la doble compresión . . . 17
  - 3.3.2. Trabajos sobre la Detección de Manipulaciones en Imágenes . . . 19

3.3.3. Trabajos sobre la Detección de Manipulaciones Intrafotograma . . . 19
3.3.4. Trabajos sobre la Detección de Manipulaciones Interfotograma . . . 19

4. Algoritmo de Detección Inter-Fotograma Propuesto 23
4.1. Tecnología Usada . . . 23
4.2. Generalidades de los Algoritmos Propuestos . . . 23
4.3. Estructura General . . . 24
4.4. Detección de Doble Compresión con los Tamaños de las Cadenas de Bits de Datos y los Macrobloques no Codificados Usando una Red Neuronal Convolucional . . . 25
4.5. Detección de Modificaciones en un Vídeo . . . 26
4.6. Experimentos . . . 30
4.6.1. Equipo de Pruebas . . . 30
4.6.2. Datasets . . . 31
4.6.3. Configuración de los Parámetros de la Red Neuronal Convolucional . 31
4.6.4. Validación de los Umbrales de la Red Neuronal Convolucional . . . 32
4.6.5. Validación de los Umbrales del Algoritmo Principal . . . 32
4.6.6. Detección de Compresión con la Red Neuronal Convolucional . . . 33
4.6.7. Detección de Manipulaciones con el Algoritmo Principal . . . 33

5. Conclusiones 35
5.1. Conclusiones . . . 35
5.2. Trabajos Futuros . . . 36

6. Introduction 37
6.1. Motivation . . . 37
6.2. Context . . . 37
6.3. Research Purpose . . . 37
6.4. Work Schedule . . . 38
6.5. Work Structure . . . 38

7. Conclusions 41
7.1. Conclusions . . . 41
7.2. Future Work . . . 42

Bibliografía 43

Índice de figuras

1.1. Diagrama de Gantt 3

2.1. Funcionamiento del grupo de imágenes 8
2.2. Esquema de capas en H.264 9
2.3. Esquema de la cadena de bits de datos 10
2.4. Esquema general de una red neuronal 11
2.5. Esquema de una neurona 12
2.6. Estructura general de una red convolucional 13

3.1. Clasificación de las manipulaciones de un vídeo 15
3.2. Duplicado de fotogramas 17
3.3. Borrado de fotogramas 17
3.4. Mezclado de fotogramas 18
3.5. Inserción de fotogramas 18

4.1. Perspectiva simplificada de los algoritmos propuestos 24
4.2. Esquema de la red neuronal convolucional 26
4.3. Esquema del algoritmo principal 27

6.1. Gantt Diagram 39

Lista de Acrónimos

AVC	*Advanced Video Coding*
CNN	*Convolutional Neural Network*
CTB	*Coding Tree Block*
CTU	*Coding Tree Unit*
DCT	*Discrete Cosine Transforme*
DST	*Discrete Sine Transforme*
ESD	*Extreme Studentized Deviate*
GOP	*Group of Pictures*
HEVC	*High Efficiency Video Coding*
MPEG	*Moving Picture Experts Group*
PIV	*Particle Image Velocimetry*
RBSP	*Raw Byte Sequence Payload*
ReLu	*Rectifier Linear Unit*

SODB *String of Data Bits*

TRF *Transformada Rápida de Fourier*

VFI *Velocity Field Intensity*

VPA *Variation of Prediction Artifact*

VPF *Variation of Prediction Footprint*

Abstract

Nowadays, mobile devices are very present in our daily lives. Every day a significant amount of images and videos are captured with this format. Because of this, the use of editing tools that allow the manipulation of the information captured with this medium is also increasing. Therefore, the processing of data from this source is of special interest in forensic analysis. Nowadays, mobile devices are very present in our daily lives. Every day a significant amount of images and videos are captured with this format. Because of this, the use of editing tools that allow the manipulation of the information captured with this medium is also increasing. Therefore, the processing of data from this source is of special interest in forensic analysis. Two algorithms are developed in this work that allow us to identify whether there is such manipulation in the videos from cell phones. The type of manipulation analyzed is the interframe. Both frame shuffling and frame insertion or cloning are detected. An algorithm based on a convolutional neural network is proposed to detect the double compression and the main algorithm that detects the manipulations in the video.

Keywords: Forensics Analysis, Digital Videos, Interframe, Macroblocks, H.264, Doble compression, Convolutional Neural Network.

Resumen

Actualmente los dispositivos móviles están muy presentes en nuestra vida cotidiana. Cada día se captura una cantidad importante de imágenes y vídeos con este formato. A causa de esto, también aumenta el uso de herramientas de edición que permiten la manipulación de la información capturada con este medio. Por lo que cobra especial interés en el análisis forense el procesamiento de los datos provenientes de esta fuente. En este trabajo se desarrollan dos algoritmos que permiten identificar si existe la dicha manipulación en los vídeos provenientes de los móviles. El tipo de manipulación analizado es el interfotograma. Se detecta tanto el barajado de fotogramas como la inserción o clonación de éstos. Se propone un algoritmo basado en una red neuronal convolucional para detectar la doble compresión y el algoritmo principal que detecta las manipulaciones en el vídeo.

Palabras clave: Análisis Forense, Vídeos Digitales, Interfotograma, Macrobloques, H.264, Doble compresión, Red Neuronal Convolucional.

Capítulo 1

Introducción

1.1. Motivación

Los avances imparables de la tecnología ofrecen muchas ventajas y facilidades para aumentar la comodidad de nuestra vida diaria pero también dan pie a un posible uso malintencionado de la misma en nuestra contra. En concreto, los smartphones son cada vez más comunes en nuestro entorno y su uso está extendido en cualquier parte del mundo. El desarrollo, la mejora de calidad de las cámaras y el fácil manejo de los teléfonos móviles han favorecido su uso sobre las cámaras de vídeo convencionales. Y gracias a estos cambios, las aplicaciones de edición de vídeos están en auge, permitiendo modificar cada vez con más detalle cualquier aspecto de un vídeo. A causa de esto, la enorme cantidad de información capturada por estos dispositivos necesita de métodos de verificación de la información robustos y fiables, ya que puede surgir la necesidad de usar estos vídeos en ámbitos jurídicos.

1.2. Contexto

Este Trabajo Fin de Grado ha sido realizado dentro del Grupo de Análisis, Seguridad y Sistemas (Grupo GASS, https://gass.ucm.es/, Grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades del proyecto de investigación THEIA (Techniques for Integrity and Authentication of Multimedia Files of Mobile Devices) con referencia FEI-EU-19-04.

1.3. Objeto de la Investigación

A raíz de la cantidad de información obtenida a través de las cámaras de los móviles, ha aumentado y mejorado el número de falsificaciones realizadas en los vídeos grabados con estas cámaras. Al contener información que puede ser crucial en el análisis forense, se necesita la confirmación de la veracidad de las grabaciones lo suficientemente fiable y robusta como para poder presentarse como una prueba.

En el trabajo se proponen dos algoritmos para detectar los distintos tipos de falsificaciones que puede sufrir un vídeo, uno de ellos basado en una de las técnicas de

detección más efectivas y actuales en el procesamiento de las imágenes que son las redes neuronales convolucionales.
Los objetivos específicos del trabajo son:

- Investigar las el estado del arte y las propuestas existentes más prometedoras en el ámbito de las manipulaciones de los vídeos.
- Desarrollar un algoritmo de detección de manipulaciones fiable y eficaz.
- Definir y entrenar una red neuronal convolucional capaz de detectar la doble compresión.
- Programar un algoritmo funcional para la detección de las manipulaciones de un vídeo.
- Testear con un *dataset* propio la validez de los algoritmos propuestos.

1.4. Plan de Trabajo

Este trabajo se ha desarrollado siguiendo tres fases:

- **Investigación**: Esta primera etapa empezó con la investigación general sobre todas las posibles manipulaciones que puede sufrir un vídeo. Se detectó que todos los vídeos modificados tenían como característica común la doble compresión, por lo que se decidió buscar un algoritmo eficaz para detectarla. Para eso, se profundizó en la compresión de los vídeos y los estándares más utilizados en la actualidad. Se decidió aplicar una red neuronal convolucional para detectar la doble compresión que funcionaría como filtro para mejorar los tiempos del algoritmo. Por otro lado, se hizo una compilación de la variedad de algoritmos recientes existentes para detectar las manipulaciones de imágenes y vídeos y se decidió utilizar el algoritmo propuesto, ya que cubre la mayoría de las manipulaciones posibles y muestra unos resultados muy efectivos. Se estudiaron varios cursos para familiarizarse con el lenguaje de programación *Python* usado para el trabajo y también se consultaron las APIs de los paquetes utilizados en el desarrollo.
- **Desarrollo**: La segunda etapa comenzó con el desarrollo de la red neuronal convolucional y el ajuste de las capas y las neuronas correspondientes. También se creó el conjunto de muestras para el entrenamiento de la red con el preprocesamiento necesario. Se buscó información sobre los paquetes predefinidos disponibles para la creación de redes neuronales. Después se codificó el segundo algoritmo con las correspondientes consultas de información sobre los paquetes necesarios de *Python*.
- **Experimentación**: En la etapa de pruebas se testearon los algoritmos con el posterior análisis de los resultados y las conclusiones obtenidas. Se repitieron los mismos experimentos usando distintas constantes en los algoritmos. Para la red neuronal, se utilizaron varias configuraciones en las capas y las neuronas para

seleccionar la red más eficaz. Para el segundo algoritmo, se repitieron las pruebas con variaciones en las constantes T_{dup} y T_{shuf}. Los vídeos manipulados usados para los experimentos contenían todas las combinaciones de manipulaciones posibles y todas las localizaciones.

- **Documentación**: Esta etapa comenzó al terminar de desarrollar los algoritmos, ya que se dio prioridad a éstos por su complejidad. Se recopiló información recogida en la primera etapa sobre todos los algoritmos vistos hasta el momento para el estado del arte y su correspondiente bibliografía. Se explicaron los algoritmos paso a paso y para terminar, se evaluaron los resultados obtenidos en las pruebas.

En la Tabla 6.1 y en la Figura 6.1 se detallan la lista de tareas realizadas y su desarrollo en el tiempo para este trabajo.

Tabla 1.1: Programación del plan de trabajo

Número	Tarea
1	Investigación
1.1	Búsqueda de algoritmos anteriores
1.2	Selección de los algoritmos
2	Desarrollo
2.1	Desarrollo de la red neuronal
2.2	Desarrollo del algoritmo principal
3	Experimentación
3.1	Creación del dataset original
3.2	Creación del dataset con doble compresión
3.3	Creación del dataset con manipulaciones
3.4	Pruebas
4	Documentación
4.1	Memoria
4.2	Bibliografía

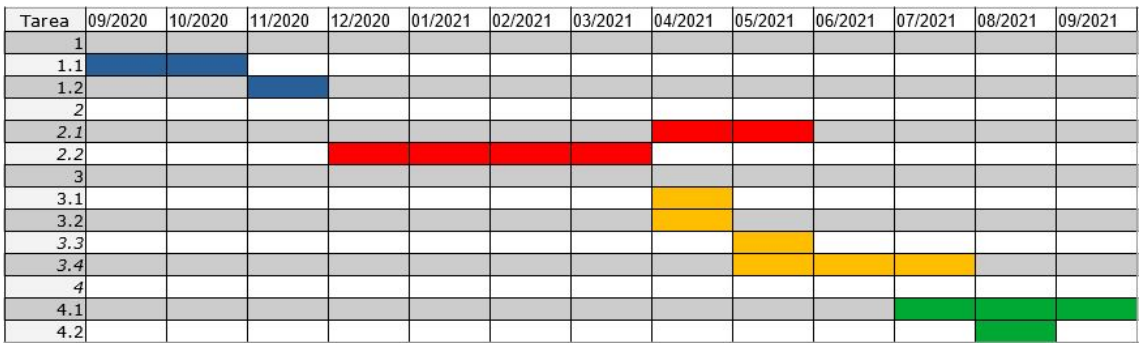


Figura 1.1: Diagrama de Gantt

1.5. Estructura del Trabajo

El trabajo está formado por 4 capítulos con la siguiente organización:

El Capítulo 2 introduce los conceptos básicos del proceso de compresión de un vídeo usados en los algoritmos presentados a continuación y los tipos de compresión más usados en la actualidad. También introduce las redes neuronales generales con algunos ejemplos y sus características y componentes básicos y con más detalle las redes neuronales convolucionales.

El Capítulo 3 detalla los dos grandes bloques de tipos de manipulación existentes en los vídeos y describe los trabajos anteriores realizados sobre la detección de manipulaciones en imágenes y vídeos usando distintos algoritmos.

El Capítulo 4 explica en detalle los conceptos básicos necesarios y los pasos de los algoritmos propuestos para la detección de manipulación en un vídeo grabado con un móvil. También detalla los experimentos realizados y los resultados de los mismos para probar los algoritmos propuestos.

El Capítulo 5 explica las conclusiones, la valoración general objetiva de los resultados, posibles mejoras para los algoritmos propuestos y futuros trabajos.

Los Capítulos 6 y 7 son las traducciones al inglés de los Capítulos 1 y 5.

Capítulo 2

Marco Conceptual

La idea principal de este Capítulo es presentar el proceso de la compresión de un vídeo y las partes básicas de las que se compone. La compresión consiste en aplicar las técnicas existentes sobre los vídeos para reducir y eliminar datos redundantes en éstos y poder almacenar o enviar la información resultante de forma más rápida y efectiva. El objetivo principal de las técnicas de compresión es reducir lo máximo posible el tamaño del fichero final preservando la calidad de la imagen. Existen diferentes técnicas que se han ido mejorando a lo largo del tiempo aunque la mayoría de herramientas relacionadas con la edición de vídeos siguen un estándar. Esto es necesario para permitir la compatibilidad e interoperabilidad entre distintos dispositivos y herramientas. En la Secciones 2.1 y 2.2 se explica el proceso de la compresión y los distintos códecs que se aplican a los vídeos. En la Sección 2.3 se especifica la estructura detallada del códec más usado. En la Sección 2.4 se detallan las redes neuronales existentes en la actualidad.

2.1. Proceso de Compresión

Durante el proceso de compresión general de un vídeo la información se separa en dos tipos, la redundante y la real. La información redundante es la que se elimina para disminuir el tamaño final. Esta información se puede recuperar a partir de la real que se deja intacta, aunque esto puede causar una disminución de la calidad final del vídeo. En algunos campos éste no es un resultado válido, por lo que surgen dos tipos de compresión que se adecuan a las distintas necesidades de los usuarios.

- **Con pérdida**: este tipo se usa para alcanzar tasas de compresión muy elevadas a costa de perder información redundante que no disminuya demasiado la calidad del vídeo al reconstruirlo. Están basados en la eliminación de todos los elementos que no son imprescindibles a la hora de reconstruir el vídeo original. Al descomprimir, se reconstruye una aproximación de los datos originales del vídeo. Este tipo normalmente se utiliza para la compresión de imágenes como por ejemplo JPEG.

- **Sin pérdida**: este tipo se usa para comprimir información que no puede disminuir su calidad. El tamaño final no difiere mucho del original, ya que no se puede eliminar

un número elevado de información. Al descomprimir, se reconstruyen exactamente los datos originales del vídeo.

2.2. Formatos de Compresión

Los estándares de vídeo actualmente más usados son *Moving Picture Experts Group* (MPEG), H.264 y el más reciente H.265.

2.2.1. Compresión MPEG

El códec MPEG-4 utiliza al igual que H.264, la predicción interfotograma para reducir la información del vídeo que se transmite entre las secuencias de fotogramas. La principal técnica es la codificación diferencial, en la que un imagen se compara con la imagen de referencia y sólo se procesan los píxeles que son distintos entre los dos fotogramas. Así, el tamaño de la información transmitida es mucho menos. Las secuencias de fotogramas codificadas con este método aparecen en el mismo orden que en el vídeo original [Com18].

Dependiendo de los tipos de imagen se usa [AF08]:

- **Codificación de imágenes I**: Este tipo de imagen se codifica sin ninguna referencia a imágenes anteriores o posteriores, se aplican sólo las técnicas de compresión de redundancia espacial y estadística.
- **Codificación de imágenes P**: Se codifican usando las técnicas de redundancia espacial, temporal y estadística. La imagen anterior tiene que ser I o B. Sólo se codificarán las partes de esta imagen que difieran de la anterior.
- **Codificación de imágenes B**: Se codifica con las imágenes precedente y consecutiva. Es similar a la codificación de las imágenes P.

2.2.2. Compresión H.264

Este códec, también llamado MPEG-4 *Advanced Video Coding* (AVC), consigue reducir hasta la mitad el tamaño final de los vídeos manteniendo la disminución de calidad a un nivel aceptable. A causa de esto se ha convertido en uno de los más usados por las herramientas de vídeos. El H.264 está basado, como sus predecesores, en un algoritmo que une la predicción y la transformación para reducir la correlación espacial. También usa la técnica de la predicción por compensación de movimiento para disminuir la información temporal transmitida. El estándar se compone de dos capas: La capa de la red de abstracción (NAL, Network Abstraction Layer) que abstrae los datos de salida para que sean compatibles con los canales de comunicación. La capa de codificación de vídeo (VCL, Video Coding Layer) que consiste en la secuencia de vídeo a codificar. Dentro de este códec podemos encontrar dos formas distintas de codificar: Intra o Inter.

Las innovaciones introducidas en este tipo y las que han hecho que sea el más usado son [Div18]:

- **Desbloqueo en bucle**: Esta técnica se aplica sobre cada fotograma. Con esta técnica los fotogramas de referencia contienen menores niveles de ruido, por lo que la eficiencia de la compresión mejora.
- **Bloques más pequeños, mejor predicción**: El tamaño de bloques usado en esta técnica es de 16x16. El tamaño pequeño resulta beneficioso en áreas que tienen una reducida definicón espacial, por lo que resulta muy útil para la definición estándar. También ofrece mayor flexibilidad en la predicción de intrafotogramas.
- **Más imágenes de referencia**: Permite que un fotograma tenga relación con fotogramas tanto anteriores como posteriores. Cualquier tipo de imágenes se pueden utilizar como referencia. Esto mejora las coincidencias encontradas durante la búsqueda del movimiento.

2.2.3. Compresión H.265

El H.265 también llamado MPEG-H Parte2 conocido como *High Efficiency Video Coding* (HEVC) es el sucesor del estándar actual H.264/MPEG-4 AVC. Su eficacia duplica la de H.264 a la hora de aplicar la compresión manteniendo el mismo nivel de calidad del vídeo. Este códec es una mejora considerable de las herramientas anteriores. Estima y compensa el movimiento de las secuencias de fotogramas para aprovechar las semejanzas entre las imágenes. El H.265 contiene nuevas estructuras como [OSLP19]:

- **Unidades de codificación nuevas**: *Coding Tree Block* (CTB) y *Coding Tree Unit* (CTU) con dimensiones que pueden variar entre 8x8 y 64x64 píxeles, lo que permite una mejor compresión.
- **Unidad de predicción (PU)**: dependiendo si se aplica el modo intra-fotograma o inter-fotograma (heredados de H.264), se utiliza un tamaño distinto.
- **Predicción Intra-fotograma**: soporta 33 modos direccionales.
- **Predicción Inter-fotograma**: se calculan los vectores de movimiento de los bloques adjuntos y con los resultados obtenidos se estima el movimiento.
- **Unidades de Transformación**: se añaden nuevas unidades como *Discrete Cosine Transforme* (DCT) y *Discrete Sine Transforme* (DST) .
- **Muestra de desplazamiento adaptativo**: mejora la calidad subjetiva de la imagen decodificada.

2.2.4. Diferencias entre Compresión H.264 y H.265

H.264 es el estándar usado hasta ahora por su eficiencia en la calidad y en la compresión de los vídeos pero con el aumento de calidad de los vídeos (4K y 8K) se necesita una mejora de este códec. Permite hasta 4K y 60fps. Disponible para Android y Iphone.

H.265 disminuye el ancho de banda necesario a la mitad para la transmisión del vídeo sin reducir la calidad del mismo. La calidad conseguida es igual a la que se obtiene

aplicando el códec H.264, obteniendo una importante disminución en el tamaño de los vídeos. Permite hasta 8K y 300fps. Disponible para Iphone (viene por defecto a partir de iOS 11) y Android (se puede usar a través de aplicaciones específicas como Kodi o Reproductor MX). La Tabla 2.1 presenta las principales diferencias descritas en [Hom16].

Tabla 2.1: Comparativa entre el estándar H.264 y el H.265

Comparativas	**H.264**	**H.265**
Nombre	H.264/MPEG4-AVC	H.265/MPEG-H
Mejoras introducidas	▪ Reducción de la tasa de bits entre un 40-50 % si se compara con MPEG-2 ▪ Compatible con alta definición	▪ Reducción de la tasa de bits entre un 40-50 % respecto al H.264 ▪ Aplicable en Ultra HD, 2K, 4K
Soporta 4K	Sí	Sí
Soporta hasta 8K	No	Sí
Soporta hasta 300fps	No	Sí

2.3. Estructura de Información en H.264

Este códec aplica una estructura jerárquica de capas al codificar las secuencias de vídeo [MMRR17]. Estas secuencias están dividas en grupos de imágenes, *Group of Pictures* (GOP), que a su vez están repartidas en porciones de distintos tamaños. Cada porción contiene diferentes macrobloques con la información relacionada de una parte de la imágen.

Las capas son las siguientes:

- **El grupo de imágenes, GOP**: representado en la Figura 2.1, es la unidad base de la codificación temporal. Contiene distintos tipos de imágenes. Cada vídeo se compone de uno o más GOP. Cada uno de ellos comienza siempre con un I-fotograma seguido de una secuencia de P-fotogramas y B-fotogramas. El tamaño de este grupo de imágenes es fijo en el vídeo pero puede variar si éste sufre una compresión.

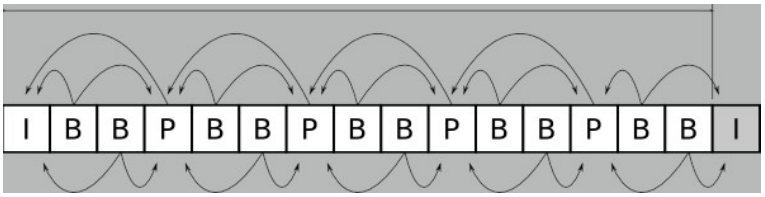


Figura 2.1: Funcionamiento del grupo de imágenes

- **Imagen**: Los distintos tipos de imágenes que componen un GOP son las siguientes:

- **Fotogramas con codificación intra (I-fotogramas)**: Fotogramas de referencia, no tienen relación ni dependencia con los otros tipos.
- **Fotogramas con codificación basadas en la predicción (P-fotogramas)**: contienen la información de la compensación de movimiento con la imagen anterior sin importar su tipo.
- **Fotogramas con codificación basadas en la predicción bidireccional (B-fotogramas)**: contienen la información variante de los fotogramas precedentes y siguientes a la misma.

Cada imagen a su vez se puede separar en las distintas partes representadas en la Figura 2.2.

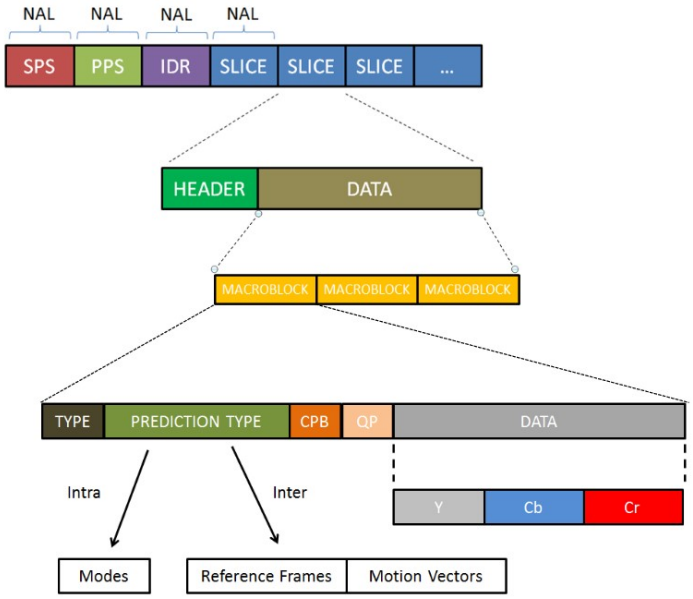


Figura 2.2: Esquema de capas en H.264

- **Porción**: Es una parte de la imagen original de tamaño variable. Se compone de los mismos tipos que las imágenes (I-porción, P-porción y B-porción). Cada porción se compone de un número de Macrobloques variable.
- **Macrobloque**: Un macrobloque es un bloque de información de la imagen sobre el que se realiza la compensación de movimiento con respecto de los demás bloques adyacentes. Hay muchos tipos distintos de macrobloques dependiendo de la información que abarcan y cómo se va a tratar en el proceso de compresión. Los tres principales son [SM18]:
  - **I-MB** (intra-coded): codificados sin predicción temporal
  - **P-MB** (predicted): codificados con predicción temporal

- **S-MB** (skipped): copiados directamente de otros fotogramas sin necesidad de vectores de movimiento. Los Skip-MB aparecen en los P-fotogramas y en los B-fotogramas. El códec estima los vectores de movimiento a partir de los macrobloques vecinos y los utiliza para calcular la predicción de movimiento compensado para los S-MB.

2.3.1. Cadenas de Bits de Datos

String of Data Bits (SODB) es una cadena de bits de datos que contiene la información del fotograma en el nivel más bajo. *Raw Byte Sequence Payload* (RBSP) conjuntamente con el primer bit, que indica el tipo de información contenida, forman la unidad NAL, que es la unidad básica utilizada en la estructura del flujo de bytes del códec H.264.

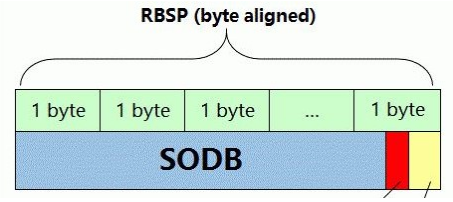


Figura 2.3: Esquema de la cadena de bits de datos

2.4. Redes Neuronales

Una red neuronal en la inteligencia artificial es un modelo computacional que emula el funcionamiento del cerebro humano. Sus características más destacadas son su capacidad de aprendizaje, el aprendizaje paralelo y la generalización, con lo que resultan muy efectivas en muchos campos para resolver distintos tipos de problemas.

En la Figura 2.4 se puede observar un ejemplo de red neuronal simple compuesta por 3 capas. Las capas de una red neuronal son un conjunto de neuronas interconectadas que procesan los valores de entrada y calculan una salida. La red neuronal se divide en tres tipos de capas, Capa de Entrada, recibe la información del entorno que se va a procesar, Capa de Salida, devuelve el resultado, Capa Oculta, no tiene conexión con el entorno, realiza los cálculos necesarios para clasificar los datos de entrada.

El entrenamiento de las redes es la parte más importante, ya que el aprendizaje que obtienen es la base para clasificación de la información de entrada. Con ello mejoran su rendimiento y funcionalidad y disminuyen la tasa de error ajustando los pesos de cada neurona con coeficientes óptimos.

Existen dos paradigmas fundamentales de aprendizaje de las redes [AARP13]:

- **Supervisado**: este proceso consiste en el entrenamiento de la red mediante un agente externo que controla el aprendizaje. Este agente es el encargado de determinar las respuestas que debería obtener la red para cada entrada predefinida.

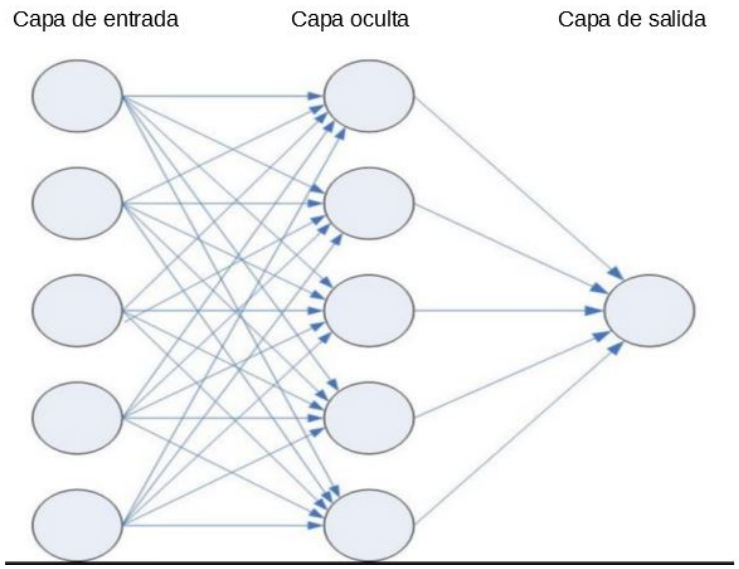


Figura 2.4: Esquema general de una red neuronal

- **No supervisado**: el proceso se basa en el aprendizaje de asociaciones entre los patrones. La salida muestra el porcentaje de similitud entre la información entrante y todo lo aprendido hasta entonces.

2.4.1. Tipos de Redes Neuronales

Las posibilidades de definir una cantidad enorme de estructuras que forman las redes ha dado lugar a multitud de estructuras diferentes. A continuación se muestran las más comunes:

- **Perceptrón**: es el ejemplo más simple de una red de aprendizaje supervisado. Compuesta sólo por una capa de entrada y otra de salida. Su funcionamiento es básico, suma todos los valores leídos de la capa de la entrada aplicando los pesos asignados anteriormente y procesa el resultado con una función de activación que genera el resultado final. El entrenamiento de esta red se realiza de forma iterativa para obtener los pesos sinápticos que mejor ajusten el resultado al esperado.

- **Red Neuronal Prealimentada**: la Figura 2.4 representa la estructura general de esta red, donde todas las neuronas están interconectadas con las neuronas de la siguiente capa, no existen ciclos y pueden existir capas ocultas. El Perceptrón, anteriormente explicado, pertenece a este tipo de redes.

- **Red neuronal Convolucional**: es una red neuronal que se utiliza para la identificación de distintas características en las entradas que posteriormente se procesan en base a las matrices bidimensionales, por lo que este tipo de redes es muy efectivo para la clasificación de imágenes. Se compone de varias capas ocultas especializadas. Tiene una parte importante de preprocesamiento que facilita los cálculos y el tiempo de ejecución necesario para la clasificación del resultado.

- **Red Neuronal Recurrente**: en estas redes cada neurona de las capas ocultas recibe como entrada su propia salida con un ajuste del retraso, esto permite almacenar y enviar información temporal entre las neuronas dándoles la posibilidad de reconocer y predecir patrones cambiantes en el tiempo.

2.4.2. Definición de una Neurona

El elemento principal de una red neuronal es la neurona [BS17]. Un ejemplo de ésta se puede observar en la Figura 2.5. Consta de un número variable de entradas (x), cada una con un peso asignado(w), una regla de propagación(h), la función de activación(a) y la salida(y). La entrada es el estímulo externo que recibe y procesa la neurona. El peso es un valor asignado a cada neurona que se aplica a la entrada y que se modifica según la red va aprendiendo. La regla de propagación indica hacia donde se envía la información de salida. Los distintos tipos son: propagación posterior, propagación hacia atrás, realimentación y otros. La función de activación es la responsable de computar la salida de una neurona en base a una entrada o un conjunto de entradas. La salida se calcula aplicando la función de activación sobre el peso y la entrada.

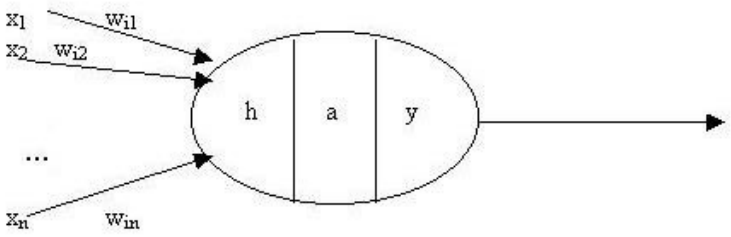


Figura 2.5: Esquema de una neurona

2.5. Redes Neuronales Convolucionales

Este tipo específico de redes son muy potentes en el procesamiento y el análisis de las imágenes [HJS+17]. Como se puede observar en la Figura 2.6, son redes multicapa que se forman a partir de múltiples capas convolucionales de una o más dimensiones y capas de reducción alternadas y que terminan en capas de conexión total cuyo objetivo principal es clasificar la información en base a las características extraídas. Como entrada se utilizan las imágenes del vídeo, preprocesadas anteriormente para reducir el tiempo y el coste de los cálculos a ejecutar. En las capas convolucionales se realiza la operación del producto escalar entre la capa anterior y los filtros o kernel que genera como resultado un mapa de características. La función de activación óptima para las redes convolucionales es la *Rectifier Linear Unit* (ReLu), elimina los valores negativos y deja los valores positivos.

Las capas de reducción se utilizan para disminuir el número de parámetros al conservar sólo las características más comunes para que el número de neuronas necesarias en la siguiente capa no se dispare.

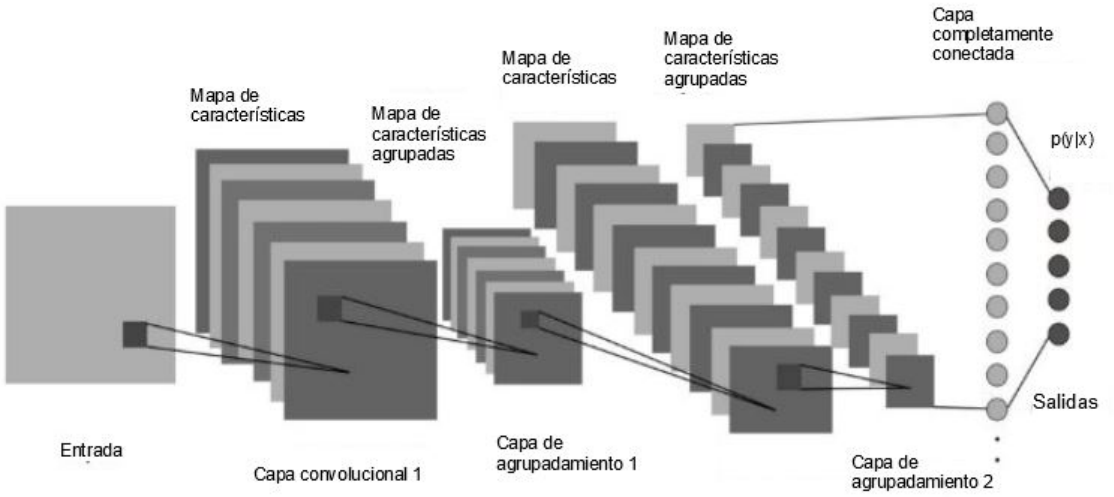


Figura 2.6: Estructura general de una red convolucional

Por último, la capa completamente conectada, a la que se aplica la función Softmax que se emplea para normalizar los resultados, clasifica la salida. Se compone de tantas neuronas como clases de clasificación existen para ésta.

Capítulo 3

Técnicas de Manipulación en Vídeo

En este Capítulo se detallan los distintos procesos de falsificaciones que existen actualmente. Las manipulaciones que se pueden aplicar sobre un vídeo se pueden dividir en dos grandes bloques que son intrafotograma e interfotograma. Estos se proceden a describir con más detalle en los apartados siguientes. Se puede observar la clasificación general en la Figura 3.1. En la Secciones 3.1 y 3.2 se explican los dos tipos de falsificaciones posibles en los vídeos. En la Sección 3.3 se detallan los trabajos anteriores más relacionados con los algoritmos propuestos en este trabajo.

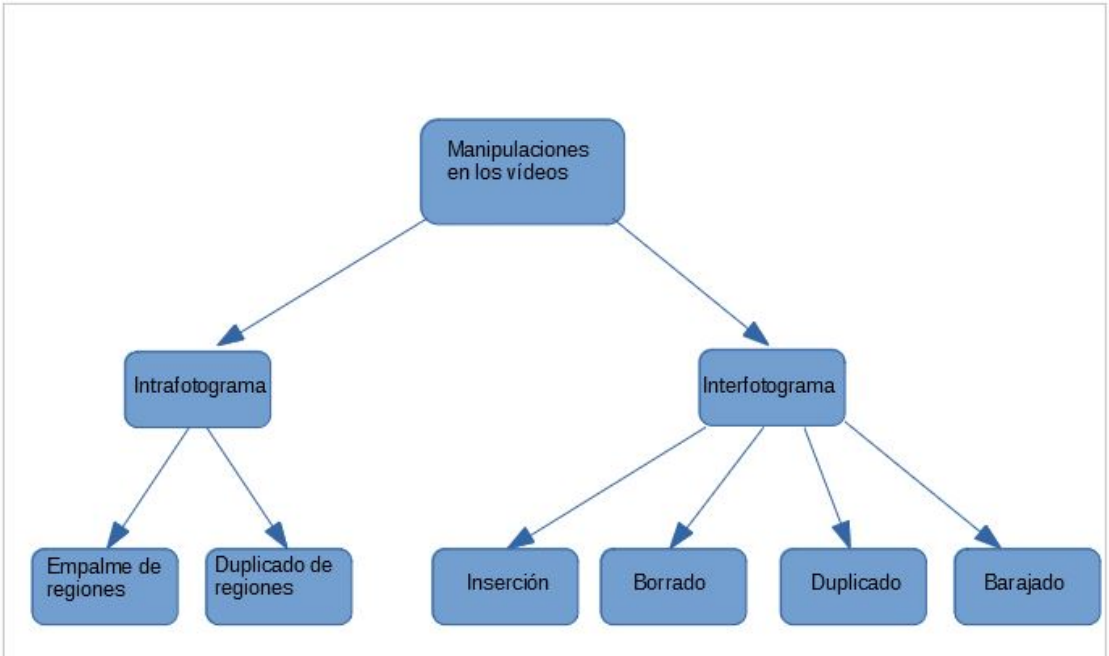


Figura 3.1: Clasificación de las manipulaciones de un vídeo

3.1. Técnicas de Manipulación Intrafotograma

Las manipulaciones Intra-fotograma se producen cuando se modifica una sección del fotograma, ya sea duplicando partes del mismo fotograma o clonando partes de otros. Estas manipulaciones se pueden aplicar a nivel de píxel, insertando o duplicando partes de la imagen, o a nivel de fotograma, recortando algún objeto específico al disminuir el tamaño del fotograma.

Las posibles falsificaciones son las siguientes [HHLH08]:

- **Duplicado de regiones**: Se produce cuando se copia una región de un fotograma concreto y se pega en el mismo fotograma pero en una ubicación distinta. Este método normalmente se utiliza para aumentar o disminuir el número de objetos que aparecen en la región tratada o mover los mismos cambiando su ubicación en el mismo fotograma.

- **Empalme de regiones**: Se produce cuando se copia una región de un fotograma y se pega en otro fotograma distinto del vídeo. Este método sirve para mover los posibles objetos que aparecen en la región entre fotogramas apareciendo antes o después en el tiempo que en el vídeo original.

3.2. Técnicas de Manipulación Interfotograma

Las manipulaciones Inter-fotograma son las que se producen al modificar el orden original de los fotogramas del vídeo, ya sea añadiendo o eliminando fotogramas. Esto sirve para añadir o quitar elementos específicos del vídeo original al tratar secuencias completas donde aparecen estos elementos. Los distintos tipos existentes están representados en las Figuras que aparecen a continuación.

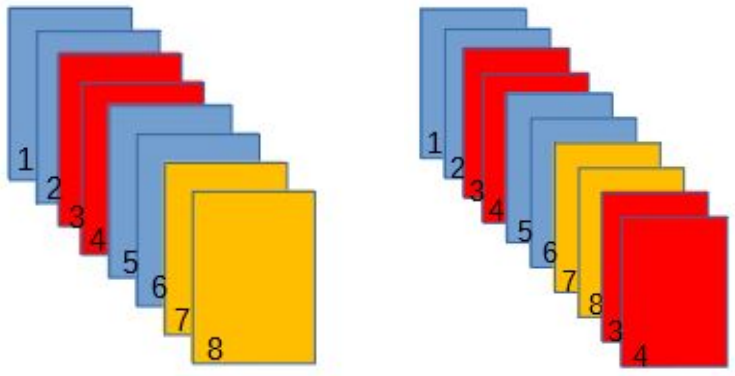
Las posibles falsificaciones son las siguientes [WJWS14]:

- **Duplicado**: El duplicado o la clonación se produce cuando se copian los fotogramas de un vídeo y se insertan en el mismo en una ubicación distinta. Este método sirve para añadir elementos del vídeo original en otro instante temporal del mismo. Se puede observar en la Figura 3.2.

- **Eliminación**: Borrado de uno o varios fotogramas del vídeo en cualquier ubicación. Se aplica para eliminar del vídeo original secuencias en las que aparece uno o varios elementos concretos que no se desea que se muestren en el vídeo. Se representa en la Figura 3.3.

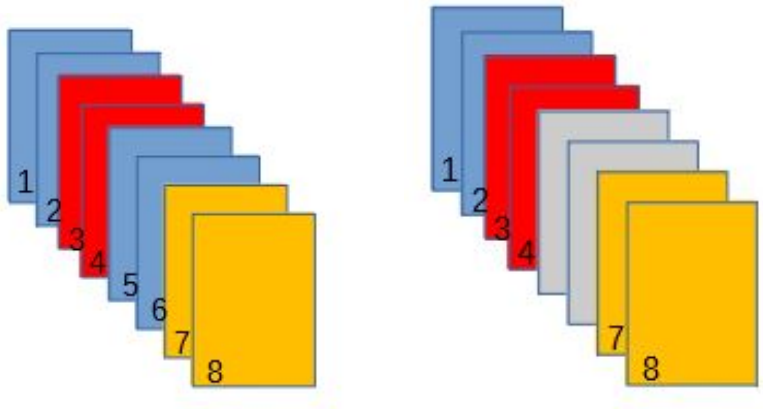
- **Mezclado**: Otra forma de duplicado pero con alteración en el orden de los fotogramas al copiarlos en el vídeo. Lo que se consigue con esta técnica es alterar en el tiempo los elementos del vídeo original. Se muestra en la Figura 3.4.

- **Inserción**: Inclusión de uno o varios fotogramas de un vídeo ajeno al vídeo original. Esto se aplica normalmente para añadir elementos a una secuencia del vídeo original. Aparece representada en la Figura 3.5.



(a) Vídeo original (b) Duplicado de fotogramas

Figura 3.2: Duplicado de fotogramas



(a) Vídeo original (b) Borrado de fotogramas

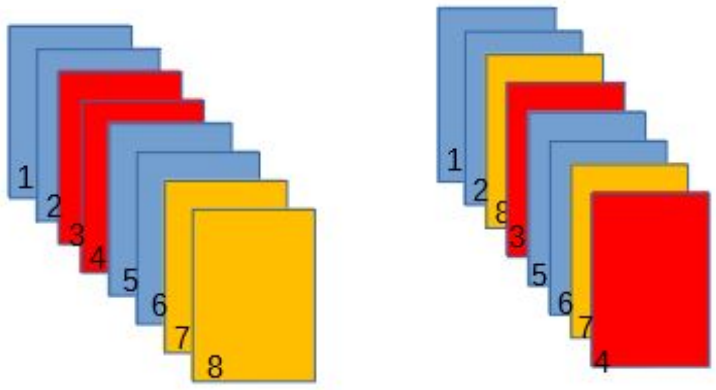
Figura 3.3: Borrado de fotogramas

3.3. Comparación de Trabajos Anteriores

Se han realizado muchas investigaciones relacionadas con la detección de manipulaciones tanto en imágenes como en vídeos. En este apartado se explican algunos de ellos y aparecen resumidos en la Tabla 3.1.

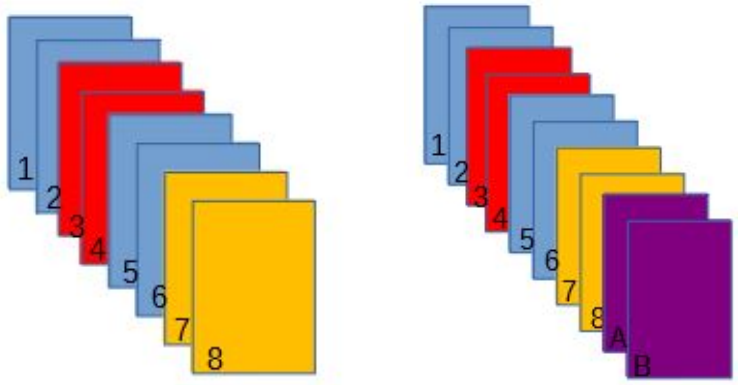
3.3.1. Trabajos sobre la doble compresión

En [HSJW13]se usan las estadísticas de Markov, se aplica la transformada discreta del coseno sobre todas las direcciones de las imágenes más algunos cálculos y utiliza las características obtenidas para la detección de la doble compresión en vídeos MPEG-4. Las pruebas realizadas sobre 30 vídeos del Dataset YUV demuestran que la tasa de aciertos del algoritmo está cerca del 90 %. La principal limitación de este algoritmo es el uso de diferentes parámetros en la doble cuantización que inevitablemente introduce errores de redondeo en los resultados.



(a) Vídeo original (b) Mezclado de fotogramas

Figura 3.4: Mezclado de fotogramas



(a) Vídeo original (b) Inserción de fotogramas

Figura 3.5: Inserción de fotogramas

En [HSJW15] se combina *Variation of Prediction Footprint* (VPF) con la medición de los artefactos de bloque para detectar la doble compresión en los vídeos con compresión MPEG-4. También estima el posible tamaño del GOP. Funciona para varios tipos de codificación, entre ellos MPEG-4 y H.264. En los resultados observados sobre 14 vídeos del Dataset YUV, se obtiene que el funcionamiento de VPF es ineficiente al analizar vídeos con mucho movimiento o vídeos que hayan sufrido otras manipulaciones interfotograma aunque el rango de detección sigue siendo mayor que el 90 %.

En [HJS+17] la detección de la doble compresión se realiza usando una *Convolutional Neural Network* (CNN) que detecta los I-fotogramas transformados en P-fotogramas. Los resultados obtenidos al analizar 21 vídeos YUV indican un tasa de aciertos del 96 %. Se necesita una mejora en el preprocesamiento del vídeo para asegurar la extracción de toda la información útil de los I-fotogramas reubicados.

3.3.2. Trabajos sobre la Detección de Manipulaciones en Imágenes

En [BS17] el análisis de las imágenes se realiza con una red neuronal convolucional entrenada previamente para detectar manipulaciones en imágenes JPEG. Dependiendo de la definición de los tipos de capas en la red el rendimiento mejora hasta el 98 % de aciertos. El mayor problema de las redes es su arquitectura, se tiene que definir correctamente el número de capas y el tipo y la cantidad de neuronas utilizadas por capa para que el aprendizaje sea efectivo.

En [CPV17] se propone como algoritmo la redefinición de descriptores locales residuales usando redes neuronales convolucionales Se utiliza una red neuronal convolucional basada en la redefinición de descriptores locales residuales en las imágenes para detectar las manipulaciones. El porcentaje de aciertos es del 94 %. La principal limitación de este algoritmo es que el proceso es muy lento y costoso en recursos, por lo que es poco eficaz en vídeos de un tamaño considerable.

3.3.3. Trabajos sobre la Detección de Manipulaciones Intrafotograma

En [HHLH08] se propone como algoritmo utilizar el ruido residual como una característica de clasificación para localizar las regiones falsificadas. Se extrae el ruido y se aplica un clasificador Bayesiano para distinguir las posibles manipulaciones. El algoritmo es sensible a la compresión, cuanto mayor es el ratio de compresión peor es la detección de las manipulaciones.

En [YYG18] la detección de manipulaciones intra-frame en los vídeos se realiza con una CNN. Consiste en pasar el vídeo por tres capas de preprocesamiento antes de aplicar la red neuronal, lo que amplifica la señal residual que dejan las modificaciones. Este algoritmo detecta las manipulaciones en los fotogramas del vídeo pero no obtiene su localización exacta aunque su tasa de aciertos es de un 97 %.

3.3.4. Trabajos sobre la Detección de Manipulaciones Interfotograma

En [CJS12] se propone un sistema basado en la consistencia del flujo óptico que detecta la inserción y la eliminación de fotogramas en el vídeo. Basándose en que los fotogramas adyacentes tienen el flujo óptico consistente entre ellos se utiliza esta información para localizar las posibles variaciones indicadoras de posibles manipulaciones. Se utiliza como base el dataset KTH aplicándole las modificaciones necesarias. El ratio de aciertos de inserción está en el 95 %, el de eliminación alcanza el 89 %. Los vídeos con mucho movimiento disminuyen la tasa de aciertos.

En [WJWS14] se propone el uso de la consistencia del campo de la velocidad para detectar la eliminación y el duplicado de fotogramas. Se aplica la prueba de *Extreme Studentized Deviate* (ESD) para identificar los tipos de falsificación y localizar las posiciones manipuladas en los vídeos falsificados. El dataset usado es uno propio. El número de aciertos está en el 96 %. El algoritmo es sensible a la compresión, cuantas más veces haya sufrido el vídeo el proceso de la compresión menor es el número de aciertos.

En [WLZM14] se propone el método basado en la detección de inconsistencias en los coeficientes de correlación de los valores de las imágenes al transformarlas a la escala de

grises. Destaca para las manipulaciones de inserción y borrado. El dataset usado es uno propio. Tiene un porcentaje alto de aciertos, el 96 %, si los vídeos tienen un fondo estático pero baja la tasa de aciertos si hay más movimiento.

En [LH15] las manipulaciones se detectan en este algoritmo usando los momentos opuestos de cromaticidad de Zernike y con el consiguiente análisis de las características del grosor. Esto consiste en la extracción de puntos anormales de grosor combinándolo después con la correlación del momento de Zernike. El dataset usado es el SULFA y uno propio. El número de aciertos del algoritmo sube al 97 %. Tiene ciertas limitaciones con los vídeos con mucho movimiento.

Tabla 3.1: Comparativa entre trabajos anteriores

Referencia	En que se basa	Que detecta	Dataset	Resultado	Limitaciones
[HHLH08]	Correlación del ruido residual	Manipulación intrafotograma de los vídeos	Vídeos manipulados MPEG-2	94 %	Sensible al ratio de compresión
[CJS12]	Consistencia del flujo óptico	Manipulación interfotograma de los vídeos	KTH	95 %	Videos con mucho movimiento
[HSJW13]	Estadísticas de Markov	Doble Compresión	YUV	90 %	Errores de redondeo en la doble cuantización
[WLZM14]	Consistencia en la correlación de coeficientes de la escala de grises	Manipulación interfotograma de los vídeos	Vídeos manipulados	96 %	Disminuye la eficacia con los fondos no estáticos
[WJWS14]	Consistencia del campo de velocidad	Manipulación interfotograma de los vídeos	Vídeos manipulados MPEG-2	96 %	Sensible al ratio de compresión
[LH15]	momentos opuestos de cromaticidad de Zernike y características del grosor	Manipulación interfotograma de los vídeos	SULFA y Vídeos manipulados	97 %	Vídeos con mucho movimiento
[HSJW15]	Medición de los artefactos de bloques con variación de predicción de la huella	Doble compresión	YUV	92 %	Vídeos con mucho movimiento
[BS17]	Redes neuronales convolucionales	Manipulación en las imágenes	Imágenes JPEG manipuladas	98 %	Construcción de una red eficiente
[CPV17]	Redefinición de descriptores locales residuales usando CNN	Manipulación en imágenes	Imágenes manipuladas	94 %	Tiempo de procesamiento muy lento
[YYG18]	Detección de objetos usando CNN	Manipulación intrafotograma de los vídeos	SYSU-OBJFORG	97 %	No detecta la localización del objeto introducido en el fotograma
[HJS+17]	Detección de I-fotogramas reubicados con una CNN	Doble compresión	YUV	96 %	Preprocesamiento muy lento

Capítulo 4

Algoritmo de Detección Inter-Fotograma Propuesto

En este capítulo se describen las técnicas que se van a usar para detectar las distintas modificaciones que pueden afectar a un vídeo. En la Sección 4.1 se explica la tecnología que se ha utilizado para el desarrollo de los algoritmos propuestos. En la Sección 4.2 se explican las generalidades necesarias para entender los algoritmos que se explican en este Capítulo. En la Sección 4.3 se muestra el esquema general de los algoritmos propuestos para detectar las modificaciones de un vídeo manipulado. En la Sección 4.4 se procede a desarrollar el algoritmo de la detección de la doble compresión basado en una red neuronal convolucional. En la Sección 4.5 se explica el procedimiento del algoritmo principal de la detección de manipulaciones en un vídeo paso a paso. En la Sección 4.6 se evalúan los algoritmos mostrando detalladamente todos los resultados obtenidos.

4.1. Tecnología Usada

El desarrollo de este algoritmo se ha realizado en la aplicación PyCharm con el lenguaje Python, versión 3.5. Se han utilizado varias librerías para el desarrollo del algoritmo propuesto como ffmpeg, para el análisis en profundidad de los fotogramas, openpiv, para la extracción de los vectores de *Velocity Field Intensity* (VFI) tanto horizontales como verticales, y cv2, para la extracción de fotogramas del vídeo. También se ha utilizado la librería numpy para las funciones matemáticas. Se ha utilizado la librería keras para la creación de las capas del modelo de red neuronal convolucional presentado en este algoritmo.

4.2. Generalidades de los Algoritmos Propuestos

Los conceptos básicos necesarios para el algoritmo descrito en este apartado son los siguientes:

- ***Variation of Prediction Artifact* (VPA)**: Un vídeo, al sufrir una manipulación y ser recomprimido, puede transformar algunos de sus I-fotogramas a P-fotogramas.

Estos fotogramas contienen más información que un P-fotograma normal, por lo que si se extraen los macrobloques del mismo, se pueden utilizar para detectar la manipulación que le ha llevado a ese estado. Para ello se compara el número de I-MBs y S-MBs del fotograma transformado con los P-fotogramas anterior y posterior usando la ecuación (4.9).

- ***Particle Image Velocimetry* (PIV)**: Es una técnica de análisis no intrusiva que recoge la visualización cualitativa y cuantitativa del flujo de partículas. El desplazamiento de los grupos de estas partículas se calcula evaluando la correlación cruzada de partes más pequeñas de cada imagen en los espacios de tiempo consecutivos.

- **ESD**: Esta técnica permite detectar valores atípicos en un flujo de datos que se aproxima a una distribución normal. Se basa en el test de Grubb's. Cada vez que se detecta un valor atípico, éste se extrae y se vuelve a aplicar el algoritmo hasta que no existan más puntos.

4.3. Estructura General

Los pasos propuestos en este trabajo para detectar manipulaciones en los vídeos son dos: el primero consiste en el procesamiento del vídeo seleccionado por una red neuronal convolucional, explicado detalladamente en la Sección 4.4. Si la red detecta la doble compresión se utiliza como indicativo de una posible manipulación, por lo que se procede a analizar el vídeo con el algoritmo principal que detecta la posible manipulación, detallado en la Sección 4.5. El esquema se expone en la Figura 4.1.

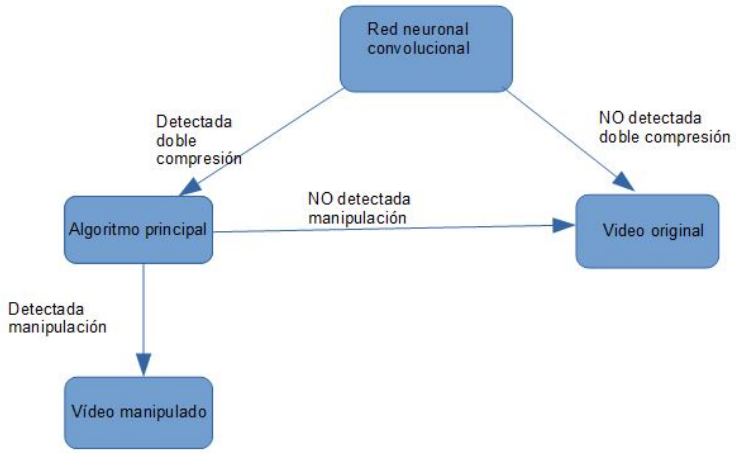


Figura 4.1: Perspectiva simplificada de los algoritmos propuestos

4.4. Detección de Doble Compresión con los Tamaños de las Cadenas de Bits de Datos y los Macrobloques no Codificados Usando una Red Neuronal Convolucional

Cualquier vídeo, al sufrir una modificación de cualquier tipo con las herramientas de edición actuales, vuelve a ser comprimido. El proceso deja un rastro que se puede identificar.

Aunque detectar la doble compresión no asegura que se haya producido alguna falsificación en el vídeo, sirve de base para aplicar las demás técnicas forenses. Cada vídeo se compone de uno o más GOP, compuestos por los tres tipos de imágenes existentes explicadas anteriormente.

Se pueden distinguir dos tipos de detección de doble compresión: con la misma estructura GOP o con distinta estructura.

Los pasos generales de esta técnica son los siguientes:

1. *Preprocesamiento del vídeo de entrada*

   Convertir todas las imágenes a Escala de grises

   Aplicar el filtro Gaussiano de paso bajo

   Restar al vídeo en escala de grises el resultado del vídeo después del filtro Gaussiano

2. *Procesamiento con la red neuronal convolucional*

3. *Clasificación según la salida de la red neuronal convolucional*

Los pasos detallados del algoritmo, representados en la Figura 4.2, son:

En la primera parte del preprocesamiento del vídeo de entrada, todos los fotogramas del mismo son convertidos a escala de grises. A continuación se dividen en grupos de tres, dónde el fotograma procesado será el intermedio. Esta división es solapada para que todos los fotogramas se procesen y no se pierda información.

Después, a todos los fotogramas se les aplica el filtro Gaussiano de paso bajo y se calcula el valor absoluto de la diferencia del vídeo resultante al aplicarle la escala de grises y el vídeo filtrado.

El procesamiento del vídeo se realiza en la red neuronal. Los grupos de imágenes anteriores se usan como entrada de la red. La red se compone de 5 capas convolucionales, cada una de ellas seguida de una capa de normalización, de la función de activación ReLu y de una capa de agrupamiento promedio. Por último, se añade la capa de agrupamiento promedio global y la capa completamente conectada.

La salida de la red neuronal es un vector de probabilidades para cada fotograma que indica si son I-P fotogramas o P-P fotogramas.

Como último paso, se clasifica cada fotograma evaluándolo con un valor límite predefinido para conseguir el resultado final.

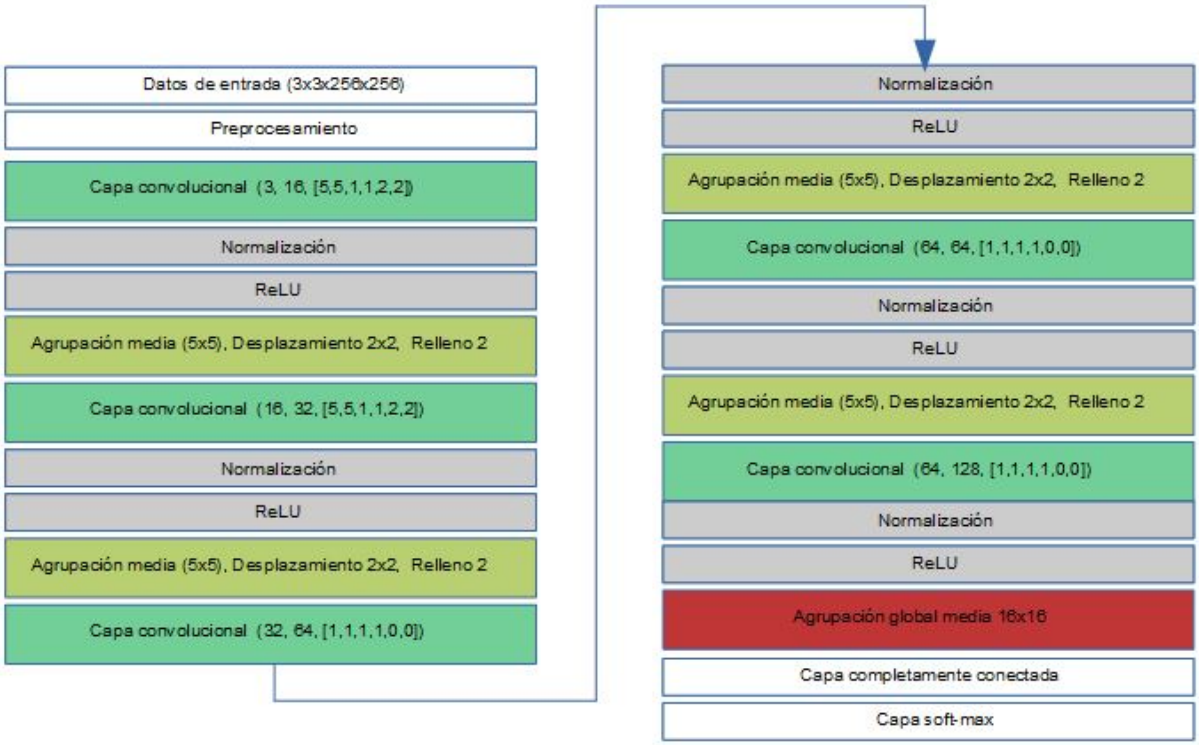


Figura 4.2: Esquema de la red neuronal convolucional

4.5. Detección de Modificaciones en un Vídeo

Los posibles tipos de modificaciones inter-fotograma que se pueden realizar sobre un vídeo y que detecta el algoritmo siguiente son la inserción, el duplicado y el mezclado de fotogramas basado en [WJSW14] y [SM18].

Los pasos generales, que se pueden observar en la Figura 4.3, son los siguientes:

1. *Identificación de la localización de los candidatos de falsificación*

   Cálculo de las secuencias horizontales y verticales de VFI entre fotogramas sucesivos

   Cálculo de VFI de ejemplo

   Cálculo de las secuencias de factor relativo

   Asignación de puntos anormales detectados usando el algoritmo ESD

2. *Validación de la localización de los candidatos*

   Comprobación de la cardinalidad

   VPA detectados con la cardinalidad

3. *Clasificación de la falsificación inter-fotograma*

   Test doble de duplicado

Test simple de duplicado

Test doble de barajado

Test simple de barajado

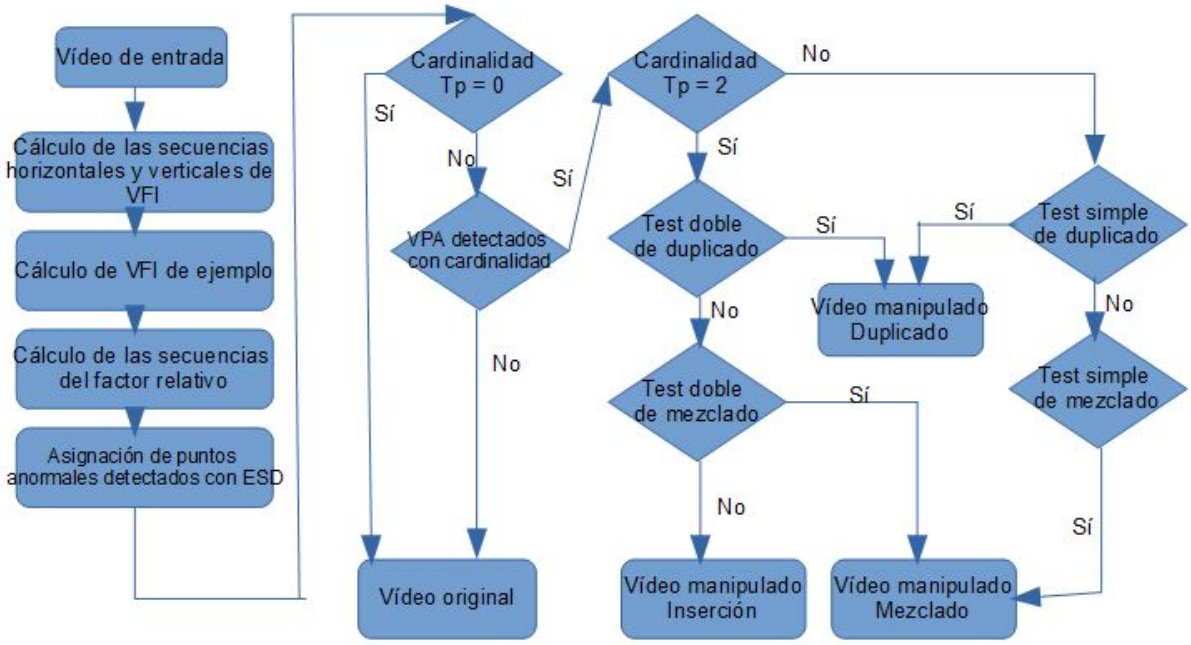


Figura 4.3: Esquema del algoritmo principal

Pasos detallados del algoritmo:

- **Identificación de la localización de los candidatos de falsificación**

  El análisis de las inconsistencias obtenidas de fotogramas consecutivos proporciona las localizaciones de los posibles candidatos. Se divide en 4 etapas diferenciadas.

  Cálculo de secuencias horizontales y verticales de VFI entre fotogramas sucesivos

  Se realiza utilizando la técnica PIV, explicada anteriormente. En esta técnica se calcula el VFI, que es el desplazamiento de los grupos de partículas entre el tiempo t y $t+1$ evaluando la correlación cruzada de subimágenes. Para esto se utiliza la *Transformada Rápida de Fourier* (TRF). La transformada se aplica a una ventana de interrogación del tamaño de 16x16 píxeles y un factor de solapamiento del 75 % con la siguiente ecuación (4.1), donde $I(i,j,t)$ y $I(i,j,t+1)$ representan la ventana de interrogación. Las F^{-1} y F^{-1} representan la TRF y la TRF inversa. En la ecuación (4.2), Re es la parte real de del resultado de la ecuación anterior. Los vectores $u(i,j,t)$ y $v(i,j,t)$ representan el VFI en el tiempo t.

$$R_c\left(u,v\right) = F^{-1}(F\left(I\left(i,j,t\right)\right)\left[F\left(I\left(i,j,t+1\right)\right)\right]) \tag{4.1}$$

$$(u(i,j,t), v(i,j,t)) = argmax_{u,v} Re\{R_c(u,v)\} \tag{4.2}$$

Las ecuaciones (4.3) y (4.4) representan el VFI calculado horizontal y vertical respectivamente en el espacio de tiempo t.

$$VFI_h = \sum_i \sum_j |u(i,j,t)| \tag{4.3}$$

$$VFI_v = \sum_i \sum_j |v(i,j,t)| \tag{4.4}$$

Cálculo de VFI de ejemplo

Este paso se utiliza para reducir los falsos positivos causados por el ruido de la cámara. Se obtienen al calcular el VFI máximo de secuencias de 3 fotogramas. Como resultado de estas ecuaciones (4.5) y (4.6) obtenemos las secuencias SVFI horizontales y verticales en el espacio de tiempo t.

$$SVFI_h(t') = max\left(VFI_h(t-1), VFI_h(t), VFI_h(t+1)\right) \tag{4.5}$$

$$SVFI_v(t') = max\left(VFI_v(t-1), VFI_v(t), VFI_v(t+1)\right) \tag{4.6}$$

Cálculo de las secuencias de factor relativo

Para obtener las secuencias de factor relativo horizontales y verticales se usan las anteriormente calculadas SVFI siguiendo las fórmulas (4.7) y (4.8). El factor relativo se utiliza para revelar los posibles picos anormales en el vídeo para distinguir entre los distintos tipos de manipulación.

$$RF_h(t') = \frac{SVFI_h(t-1) + SVFI_h(t+1)}{SVFI_h(t-1) \times SVFI_h(t+1)} \times SVFI_h(t') \tag{4.7}$$

$$RF_v(t') = \frac{SVFI_v(t-1) + SVFI_v(t+1)}{SVFI_v(t-1) \times SVFI_v(t+1)} \times SVFI_v(t') \tag{4.8}$$

Asignación de puntos anormales detectados usando el algoritmo ESD

Se aplica el algoritmo ESD, explicado anteriormente, para detectar los puntos anormales de las secuencias calculadas. Esto se usa para distinguir los distintos tipos de manipulación realizada sobre el vídeo.

- **Validación de la localización de los candidatos**

  El valor cardinal de *tp* denota el número de puntos contradictorios detectados en el vídeo. A partir de este valor se realizan las distintas comprobaciones que permitirán descartar la posible falsificación o detectar su tipo.

Comprobación de la cardinalidad

Si el valor cardinal de *tp* es cero se puede deducir que el vídeo no se ha manipulado, es el original. Si es distinto, se procederá a analizarlo más en profundidad.

VPA detectados con la cardinalidad

Se aplica el algoritmo VPA explicado anteriormente. Al tener localizado el primer punto de manipulación se extraen los I-MBs y los S-MBs de los fotogramas contiguos y se aplica la ecuación (4.9) tomando como n el P-fotograma que aparece en el punto de manipulación, i() como los I-MBs del fotograma correspondiente y s() como los S-MBs del mismo fotograma. Los fotogramas $(n-1)$ y $(n+1)$ son los P-fotogramas anterior y siguiente al marcado por *tp*.

$$[i(n-1) < i(n) \wedge [i(n) > i(n+1)] \wedge [s(n-1) > s(n)] \wedge [s(n) < s(n+1)]] \quad (4.9)$$

- **Clasificación de la falsificación inter-fotograma**

  Al localizar la ubicación de la posible manipulación sólo queda detectar que tipo de manipulación que se ha realizado. Para esto se utilizan los siguientes test.

  Test doble de los duplicados

  En este paso del algoritmo se tienen dos puntos de posible manipulación que indican el inicio y el fin de los fotogramas duplicados en el vídeo. Para localizar estos fotogramas se utiliza una ventana deslizante del mismo tamaño que hay entre los puntos t_{p1} y t_{p2}. Se aplica la correlación de VFI entre la ventana y el clip del vídeo que se está comprobando y se detecta si es el conjunto de fotogramas copiado si se cumple que la correlación es mayor que el valor límite T_{dup}.

  Se utiliza la comparación con el VFI porque es mucho más rápida que la comparación directa de fotogramas.

  Test simple de los duplicados

  Es un caso especial de duplicados. Sólo se tiene un punto de manipulación que ocurre si el segundo punto coincide con el inicio del vídeo o el final del vídeo. El vídeo se divide en dos partes, C_1 y C_2, tomando como el punto de separación el valor de t_p. El tamaño de los clip, l, se deduce de los fps del vídeo. Se calcula el VFI de C_1 y C_2. Con esto se localiza la manipulación pero no se conoce el tamaño exacto de los fotogramas manipulados que puede ser mayor que l. Por lo que se necesita repetir el proceso pero aumentando el tamaño del clip en el que se ha detectado la manipulación. Se añaden los l siguientes fotogramas a C_1 o a C_2. Esto se repite hasta que la correlación queda por debajo del valor límite T_{dup}. El siguiente paso es dividir los últimos l fotogramas añadidos al clip y volver a calcular la correlación. Este proceso se repite en bucle hasta encontrar todos los fotogramas duplicados.

  Test doble de barajado

Al igual que en el test doble de duplicados, se tienen dos puntos de manipulación que indican el posible inicio y fin de los fotogramas barajados. Se guardan en dos vectores los VFI horizontales y verticales de todas las combinaciones posibles de fotogramas que están en la ventana de desplazamiento y se ordenan. Este paso también se aplica para los fotogramas entre t_{p1} y t_{p2}. Después se aplica la correlación entre los dos grupos de fotogramas. Si ésta supera el valor límite T_{shuf} se detecta la manipulación de barajado.

Test simple de barajado

Si el test simple de duplicados da negativo, se analiza para el barajado simple. Al tener sólo un punto de manipulación, el inicio o el final del vídeo son los candidatos para ser el segundo punto. Se elige la parte con menor número de fotogramas. Ahora que se tienen dos puntos de manipulación, se usa el algoritmo del Test doble de barajado.Si el test simple de duplicados da negativo, se analiza para el barajado simple. Al tener sólo un punto de manipulación, el inicio o el final del vídeo son los candidatos para ser el segundo punto. Se elige la parte con menor número de fotogramas. Ahora que se tienen dos puntos de manipulación, se usa el algoritmo del Test doble de barajado.

Test doble de inserción

Si el Test doble de duplicados y el test doble de barajado han dado negativo se detecta la manipulación de inserción de fotogramas, ya que es la única posibilidad que queda con dos puntos de manipulación.

4.6. Experimentos

Se han realizado pruebas separadas para la red neuronal convolucional de detección de la doble compresión y para el algoritmo de detección de manipulaciones que se explican a continuación.

4.6.1. Equipo de Pruebas

En la Tabla 4.1 se detalla la especificación del equipo utilizado para los experimentos expuestos en este capítulo.

Tabla 4.1: Especificaciones del equipo

Recurso	Descripción
Sistema Operativo	Ubuntu 16.04
Memoria RAM	16 GB
Procesador	AMD Ryzen 5 1400
Gráficos	Nvidia GTX 1050
Arquitectura	64 bits
Disco Duro	1 TB HDD

4.6.2. Datasets

Los Datasets utilizados en las pruebas de detección, en el entrenamiento y el testeo de la red neuronal han sido creados con la herramienta Openshot Video Editor. Se han utilizado vídeos grabados con móviles a los que se les han aplicado las distintas manipulaciones. En la Tabla 4.2 se especifican los distintos vídeos utilizados para los experimentos.

Para la red neuronal convolucional se grabaron 10 vídeos originales con varios móviles. Los mismos vídeos se comprimieron con el códec H.264 para generar los vídeos con la doble compresión. 6 de los vídeos originales y otros 6 de los comprimidos se utilizaron como entrenamiento y el testeo para la red neuronal. Los demás sirvieron como dataset para el experimento.

Para el algoritmo propuesto se grabaron 9 vídeos originales. A éstos se les aplicaron las manipulaciones de inserción, duplicado y mezclado, y a su vez, estas manipulaciones se colocaron en distintas ubicaciones: al inicio, a la mitad del vídeo y al final. A cada vídeo sólo se le aplicó una falsificación a la vez.

Tabla 4.2: Dataset

Vídeos	**Manipulación**
10 vídeos	Originales
10 vídeos	Doble compresión
9 vídeos	Inserción Inicio
9 vídeos	Inserción Medio
9 vídeos	Inserción Final
9 vídeos	Originales
9 vídeos	Duplicado Inicio
9 vídeos	Duplicado Medio
9 vídeos	Duplicado Final
9 vídeos	Mezclado Inicio
9 vídeos	Mezclado Medio
9 vídeos	Mezclado Final

4.6.3. Configuración de los Parámetros de la Red Neuronal Convolucional

En las primeras capas convolucionales se han utilizado kernels de 5x5. En las siguientes se ha ido disminuyendo el tamaño a 3x3 hasta terminar con 1x1.

En las capas completamente conectadas la función de activación óptima ha sido la ReLu, es una función que sólo se activa si los valores de entrada son positivos anulando los valores negativos. Es de las mejores para el tratamiento de imágenes ya que los valores negativos no son importantes para éstas.

Se ha utilizado el optimizador Adam a la hora de compilar la red neuronal convolucional para los cálculos de los pesos de cada neurona de la red.

4.6.4. Validación de los Umbrales de la Red Neuronal Convolucional

En la Tabla 4.3 se muestran las distintas configuraciones de entrenamiento testeadas con la red neuronal y sus correspondientes resultados de detección de la compresión. Se tomaron como valores de referencia para el inicio de las pruebas los recomendados en [HJS+17] y se ajustaron a la red neuronal propuesta quedando como la mejor configuración los valores de 5 epochs y 25 de batch size. No se realizaron pruebas con valores mucho más grandes a causa de las limitaciones del equipo, ya que al aumentar el tamaño de la red se necesita mucha más potencia de cálculo.

Tabla 4.3: Pruebas con distintas configuraciones de la red neuronal convolucional

Epochs	Batch size	% de Detección
3	25	66,67 %
4	25	66,67 %
5	25	83,33 %
3	30	66,67 %
4	30	83,33 %
5	30	50 %

4.6.5. Validación de los Umbrales del Algoritmo Principal

En la Tabla 4.4 se detallan los resultados alcanzados al aplicar el algoritmo al dataset definido anteriormente con los valores de los umbrales $T_{dup} = 0.93$ y $T_{shuf} = 0.91$. Al ser los umbrales menos restrictivos la detección aumenta en algunos de los experimentos pero en general disminuye la efectividad, por lo tanto se decide descartar estos valores.

Tabla 4.4: Resultados de las pruebas con umbrales $T_{dup} = 0.93$ y $T_{shuf} = 0.91$

Manipulación en el vídeo	Ubicación de la manipulación	% de Detección
Duplicado	Inicio	64,45 %
Duplicado	Medio	75.56 %
Duplicado	Fin	87.78 %
Barajado	Inicio	64,45 %
Barajado	Medio	87.78 %
Barajado	Fin	76,67 %
Inserción	Inicio	64,45 %
Inserción	Medio	64,45 %
Inserción	Fin	76,67 %

En la Tabla 4.5 se detallan los resultados logrados al aplicar el algoritmo al dataset definido anteriormente con los valores de los umbrales $T_{dup} = 0.97$ y $T_{shuf} = 0.96$. Al aumentar los umbrales demasiado el algoritmo pierde mucha eficacia, algunos porcentajes de aciertos se mantienen pero en la mayoría de los casos disminuyen significativamente, por lo que se decide descartar también estos valores.

Tabla 4.5: Resultados de las pruebas con umbrales $T_{dup} = 0.97$ y $T_{shuf} = 0.96$

Manipulación en el vídeo	Ubicación de la manipulación	% de Detección
Duplicado	Inicio	65,5 %
Duplicado	Medio	76,67 %
Duplicado	Fin	65,5 %
Barajado	Inicio	55,5 %
Barajado	Medio	55,5 %
Barajado	Fin	65,5 %
Inserción	Inicio	68,67 %
Inserción	Medio	64,45 %
Inserción	Fin	76,67 %

Después de varias pruebas con distintos valores que sirven como umbrales para la detección se han escogido $T_{dup} = 0.96$ y $T_{shuf} = 0.95$ al ser los valores límites con mejores resultados.

4.6.6. Detección de Compresión con la Red Neuronal Convolucional

Se usaron los vídeos del Dataset originales y los comprimidos con la codificación H.264 para los experimentos.

Los vídeos comprimidos del Dataset se separaron en tres grupos, donde cada grupo se utilizó para una etapa concreta del proceso. El primer grupo se utilizó para el entrenamiento de la red neuronal, el segundo grupo como el test de control de la red y el tercer grupo es el que se usó como la prueba final.

El porcentaje de aciertos en la detección de la doble compresión es del 83,33 %.

4.6.7. Detección de Manipulaciones con el Algoritmo Principal

Se han utilizado vídeos con distintos tipos de falsificaciones y para cada tipo de manipulaciones de fotogramas distintas localizaciones y con distinto número de fotogramas.

Los vídeos del Dataset se pueden separar en tres grupos, donde cada grupo está compuesto por un tipo de manipulación específico: inserción, duplicado y mezclado, y dentro de cada grupo por la localización de los fotogramas modificados en el vídeo.

En la Tabla 4.6 se detallan los resultados obtenidos.

Se puede observar que el porcentaje de detección disminuye al tratar de identificar las manipulaciones ubicadas en la parte central del vídeo. También que la manipulación del Duplicado es la que tiene el mejor porcentaje de aciertos en el algoritmo propuesto, mientras que el Barajado es la que tiene el porcentaje más bajo. Comparando las ubicaciones, la manipulación al final del vídeo es la que tiene el mejor porcentaje.

Tabla 4.6: Resultados de detección de manipulaciones

Manipulación en el vídeo	**Ubicación de la manipulación**	**% de Detección**
Duplicado	Inicio	76.66 %
Duplicado	Medio	75.56 %
Duplicado	Fin	87.78 %
Barajado	Inicio	64,45 %
Barajado	Medio	64,45 %
Barajado	Fin	76,67 %
Inserción	Inicio	68.67 %
Inserción	Medio	64,45 %
Inserción	Fin	76,67 %

Capítulo 5

Conclusiones

5.1. Conclusiones

En el estado del arte se procede a recopilar la información de los distintos algoritmos recientes que tratan las manipulaciones en los vídeos y en las imágenes al igual que también la doble compresión como marca de una posible manipulación. Al observarse la posible mejora si se unen varios algoritmos distintos se propone la solución de aplicar dos algoritmos a un vídeo para optimizar la tasa de aciertos. Se elige la red neuronal convolucional por ser uno de los mejores algoritmos para tratar tantos imágenes como vídeos.

El primer algoritmo elegido es, la anteriormente nombrada, red neuronal convolucional que detecta la doble compresión en los vídeos en los que sirve como un primer indicio de posibles manipulaciones. Si no se detecta nada se entiende que el vídeo no está modificado. Si se detecta doble compresión no significa que habrá manipulación obligatoriamente sino que la probabilidad es más alta.

El segundo algoritmo se basa en la identificación de los posibles puntos de manipulación usando VFI y ESD a los que a continuación se les aplican varios tests dependiendo del cardinal de los puntos de posible manipulación que se hayan encontrado y se llega a la conclusión de si el vídeo tratado está manipulado o en caso contrario solo tiene doble compresión.

Antes de proceder a evaluar la red neuronal se entrena con varios vídeos originales y otros comprimidos para mejorar el aprendizaje pero evitando el sobreentrenamiento. Al pasar varias pruebas se ajustan el número de capas y neuronas existentes en la red dejándola preparada para los experimentos. El dataset usado es uno propio compuesto por 10 vídeos originales y 10 con doble compresión, de los cuales 3 son utilizados para el entrenamiento, otros 3 para el testeo y los últimos 4 para la evaluación de la red.

La evaluación del segundo algoritmo se realiza con un dataset propio compuesto por 9 vídeos originales a los que se aplican las distintas manipulaciones en distintas localizaciones. Cada vídeo solo contiene una manipulación en una localización concreta. Al contener dos umbrales, para llegar a los valores óptimos, se realizan las pruebas con distintos valores hasta ajustarlos para los mejores resultados.

Al aplicar primero la red neuronal convolucional a los vídeos de entrada para detectar

la doble compresión se busca descartar todos los vídeos que tienen una compresión simple para acelerar los tiempos del proceso completo, ya que no se llega a ejecutar el algoritmo principal que es el más costoso. El segundo algoritmo trata de detectar las manipulaciones utilizando distinta información obtenida de las relaciones entre fotogramas consecutivos aplicándoles distintos cálculos.

Después de analizar los resultados de las pruebas se ha llegado a la resolución de que el conjunto de los algoritmos propuestos permite detectar las manipulaciones pero necesita mejorar la tasa de los aciertos para ser más fiable, ya que comparado con los aciertos de los algoritmos de trabajos anteriores no llega al mismo ratio.

Este algoritmo está limitado en su uso para la detección de más de una manipulación en el vídeo analizado, ya que sólo detecta la primera manipulación existente. También depende de la longitud del vídeo, para los más grandes el tiempo de procesamiento se hace muy costoso.

5.2. Trabajos Futuros

El análisis forense, al ser un tema tan actual y presente en una gran variedad de situaciones, necesita ser fiable y robusto por lo que se deberían utilizar los sistemas más avanzados y rápidos como las redes neuronales para tratar vídeos.

Los resultados de las pruebas nos muestran que en general los algoritmos propuestos son bastante buenos pero siempre mejorables. Por eso, a continuación se ofrece una lista de las posibles mejoras:

- Se podría reemplazar el algoritmo de detección propuesto por una red neuronal que permita detectar el tipo de manipulación y el lugar exacto dónde se ha encontrado ésta para precisar más en detalle toda la información que puede ser útil.
- Se deberían tratar de mejorar los tiempos de procesamientos para los vídeos más grandes, ya que al procesar tantos cálculos pueden ser demasiado costosos.
- Otra posible ampliación del trabajo actual podría ser la detección de varias manipulaciones realizadas en el mismo vídeo, ya que actualmente solo se detecta la primera que se encuentre.
- Como última propuesta, añadir la posibilidad de diferenciar los tipos de las manipulaciones detectadas.

Capítulo 6

Introduction

6.1. Motivation

The unstoppable advances in technology offer many advantages and facilities to increase the comfort of our daily life but also give rise to a possible malicious use of it against us. In particular, smartphones are becoming more and more common in our environment and their use is widespread all over the world. The development, improved quality of cameras and the easy handling of cell phones have favored their use over conventional video cameras. And thanks to these changes, video editing applications are booming, allowing any aspect of a video to be modified in more and more detail. Because of this, the enormous amount of information captured by these devices requires robust and reliable methods of information verification, as the need may arise to use these videos in legal settings.

6.2. Context

This Final Degree Project has been carried out within the Analysis, Security and Systems Group (Group GASS, https://gass.ucm.es/, Group 910623 of the catalog of groups recognized by the UCM) as part of the activities of the research project THEIA (Techniques for Integrity and Authentication of Multimedia Files of Mobile Devices) with reference FEI-EU-19-04.

6.3. Research Purpose

As a result of the amount of information obtained from cell phone cameras, the number of forgeries made on videos recorded with these cameras has increased and improved. As they contain information that can be crucial in forensic analysis, there is a need for confirmation of the veracity of the recordings that is reliable and robust enough to be presented as evidence.

In this work, two algorithms are proposed to detect the different types of falsifications that a video can suffer, one of them based on one of the most effective and current detection techniques in image processing, which are the convolutional neural networks.

6.4. Work Schedule

This work has been developed in three phases:

- **Investigation**: This first phase began with a general investigation of all the possible manipulations that a video can undergo. It was detected that all the modified videos had double compression as a common characteristic, so it was decided to look for an efficient algorithm to detect it. For this purpose, we studied the compression of videos and the most currently used standards. It was decided to apply a convolutional neural network to detect the double compression that would work as a filter to improve the algorithm times. On the other hand, a compilation of the variety of recent existing algorithms to detect image and video manipulations was made and it was decided to use the proposed algorithm, it covers most of the possible manipulations and shows very effective results. Several courses were studied to become familiar with the *Python* programming language used for the work and the APIs of the packages used in the development were also consulted.

- **Development**: The second phase began with the development of the convolutional neural network and the adjustment of the layers and the corresponding neurons. The set of samples for training the network with the necessary preprocessing was also created. Information was sought on the predefined packages available for neural network creation. Then the second algorithm was coded with the corresponding queries for information about the necessary *Python* packages.

- **Experimentation**: In the testing stage, the algorithms were tested with subsequent analysis of the results and conclusions obtained. The same experiments were repeated using different constants in the algorithms. For the neural network, various configurations in the layers and neurons were used to select the most effective network. For the second algorithm, the tests were repeated with variations in the T_{dup} and T_{shuf} constants. The manipulated videos used for the experiments contained all possible combinations of manipulations and all locations.

- **Documentation**: This stage started when finishing the development of the algorithms, since priority was given to them due to their complexity. Information collected in the first stage was compiled on all the algorithms seen so far for the state of the art and their corresponding bibliography. The algorithms were explained step by step and finally,the results obtained in the tests were evaluated.

The list of tasks performed and their development over time for this work are detailed in Table 6.1 and Figure 6.1.

6.5. Work Structure

The rest of the paper is organized in 4 chapters with the structure discussed below:

Chapter 2 introduces the basic concepts of the compression process of a video used in the algorithms presented below and the types of compression most commonly used

Tabla 6.1: Work plan scheduling

Number	Task
1	Investigation
1.1	Search of previous algorithms
1.2	Selection of the algorithms
2	Development
2.1	Development of the neural network
2.2	Development of the main algorithm
3	Experimentation
3.1	Creation of the original dataset
3.2	Creation of the double compression dataset
3.3	Creation of the dataset with manipulations
3.4	Testing
4	Documentation
4.1	Memory
4.2	Bibliography

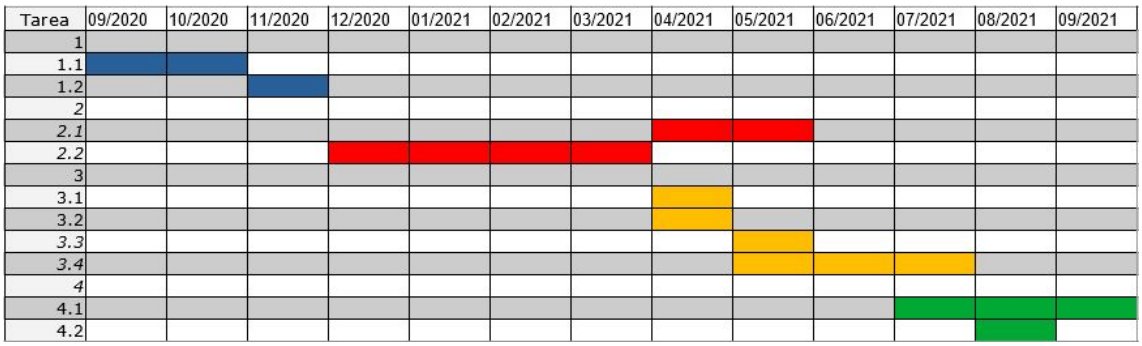


Figura 6.1: Gantt Diagram

today. It also introduces general neural networks with some examples and their basic characteristics and components and in more detail convolutional neural networks.

Chapter 3 details the two major blocks of manipulation types existing in videos and describes previous work done on image and video manipulation detection using different algorithms.

Chapter 4 explains in detail the basic concepts required and the steps of the proposed algorithms for manipulation detection in a video recorded with a cell phone. It also details the experiments performed and the results of the experiments to test the proposed algorithms.

Chapter 5 explains the conclusions, the overall objective assessment of the results, possible improvements for the proposed agorithms and future work.

Capítulo 7

Conclusions

7.1. Conclusions

In the state of the art we proceed to collect information on the various recent algorithms that deal with video and image manipulations as well as double compression as a mark of possible manipulation. Observing the possible improvement by combining several different algorithms, the solution of applying two algorithms to a video to optimize the hit rate is proposed. The convolutional neural network is chosen because it is one of the best algorithms for dealing with both images and videos.

The first algorithm chosen is the previously named convolutional neural network that detects double compression in the videos in which it serves as a first indication of possible manipulations. If nothing is detected it is understood that the video is not modified. If double compression is detected, it does not mean that there will necessarily be manipulation, but that the probability is higher.

The second algorithm is based on the identification of possible manipulation points using VFI and ESD to which then several tests are applied depending on the cardinal of the possible manipulation points that have been found and the conclusion is reached as to whether the treated video is manipulated or otherwise only has double compression.

Before proceeding to evaluate the neural network, it is trained with several original videos and other compressed videos to improve learning but avoiding overtraining. After passing several tests, the number of layers and neurons in the network is adjusted, leaving it ready for the experiments. The dataset used is a proprietary one composed of 10 original videos and 10 with double compression, of which 3 are used for training, another 3 for testing and the last 4 for network evaluation.

The evaluation of the second algorithm is performed with a proprietary dataset composed of 9 original videos to which the different manipulations are applied in different locations. Each video contains only one manipulation at a specific location. As it contains two thresholds, in order to reach the optimal values, tests are performed with different values until they are adjusted for the best results.

By first applying the convolutional neural network to the input videos to detect the double compression, we try to discard all the videos that have a simple compression to speed up the time of the whole process, since we do not get to run the main algorithm

which is the most expensive one. The second algorithm tries to detect the manipulations using different information obtained from the relationships between consecutive frames by applying different calculations to them.

After analyzing the results of the tests, we have reached the resolution that the set of the proposed algorithms allows to detect the manipulations but needs to improve the hit rate to be more reliable, since compared with the hits of the algorithms of previous works it does not reach the same ratio.

This algorithm is limited in its use for the detection of more than one manipulation in the analyzed video, since it only detects the first existing manipulation. It also depends on the length of the video, for larger ones the processing time becomes very expensive.

7.2. Future Work

Forensic analysis, being such a topical subject and present in a wide variety of situations, needs to be reliable and robust so the most advanced and fast systems such as neural networks should be used to process videos.

The results of the tests show that in general the proposed algorithms are quite good but can always be improved. A list of possible improvements is given below:

- The proposed detection algorithm could be replaced by a neural network to detect the type of manipulation and the exact place where it was found in order to pinpoint in more detail all the information that could be useful.
- Efforts should be made to improve processing times for larger videos, as processing so many computations can be too costly.
- Another possible extension of the current work could be the detection of several manipulations performed on the same video, as currently only the first one found is detected.
- As a last proposal, the possibility of differentiating the types of manipulations detected.

Bibliografía

[AARP13] D. L. Alfonzo Azuaje and I. M. Romero Piñero. Aprendizaje Supervisado y no Supervisado, 2013.

[AF08] E. M. Aguilar Fernández. Decodificador de vídeo MPEG-2 en MATLAB y análisis del bitstream, 2008.

[BS17] B. Bayar and M. C. Stamm. Design Principles of Convolutional Neural Networks for Multimedia Forensics. In *Media Watermarking, Security, and Forensics*, pages 77–86. Society for Imaging Science and Technology, November 2017.

[CJS12] J. Chao, X. Jiang, and T. Sun. A Novel Video Inter-frame Forgery Model Detection Scheme Based on Optical Flow Consistency. In *The International Workshop on Digital Forensics and Watermarking 2012*, pages 30–31. Springer, November 2012.

[Com18] Axis Communications. Video Compression, 2018.

[CPV17] D. Cozzolino, G. Poggi, and L. Verdoliva. Recasting Residual-based Local Descriptors as Convolutional Neural Networks: an Application to Image Forgery Detection. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, pages 159–164, June 2017.

[Div18] Divx. Un nuevo estándar de vídeo digital, 2018.

[HHLH08] C. Hsu, T. Hung, C. Lin, and C. Hsu. Digital Watermarking. In *2008 IEEE 10th Workshop on Multimedia Signal Processing*, pages 170–174. IEEE, October 2008.

[HJS+17] P. He, X. Jiang, T. Sun, S. Wang, and Y. Dong. Frame-wise detection of relocated I-frames in double compressed H.264 videos based on convolutional neural network. *Journal of Visual Communication and Image Representation*, 48:149–158, October 2017.

[Hom16] Xataka Home. Códec H.265, 2016.

[HSJW13] P. He, T. Sun, X. Jiang, and S. Wang. Detection of double compression in mpeg-4 videos based on markov statistics. *IEEE*, 20(1):447–450, May 2013.

[HSJW15] P. He, T. Sun, X. Jiang, and S. Wang. Double compression detection in mpeg-4 videos based on block artifact measurement with variation of prediction footprint. *IEEE*, 1(2):787–793, August 2015.

[LH15] Y. Liu and T. Huang. Exposing video inter-frame forgery by Zernike opponent chromaticity moments and coarseness analysis. 23:223–238, August 2015.

[MMRR17] J. A. Michell Martín and G. A. Ruiz Robredo. Compresión de Video, 2017.

[OSLP19] R. A. Olivera Solís and Y. López Pérez. Estimation of objective video quality using HEVC / H.265, 2019.

[SM18] K. Sitara and B.M. Mehtre. Detection of Inter-Frame Forgeries in Digital Videos. *Forensic Science International*, 289(1):186–206, May 2018.

[WJSW14] Y. Wu, X. Jiang, T. Sun, and W. Wang. Exposing video inter-frame forgery based on velocity field consistency. pages 2674–2678, 05 2014.

[WJWS14] Y. Wu, X. Jiang, W. Wang, and T. Sun. Exposing video inter-frame forgery based on velocity field consistency. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2674–2678. IEEE, 2014.

[WLZM14] Q. Wang, Z. Li, Z. Zhang, and Q. Ma. Video Inter-Frame Forgery Identification Based on Consistency of Correlation Coefficients of Gray Values. 02:51–57, March 2014.

[YYG18] S. Weng Y. Yao, Y. Shi and B. Guan. Deep Learning for Detection of Object-Based Forgery in Advanced Video. *Symmetry*, 10(1):205–214, January 2018.