UNIVERSIDAD DE LA REPÚBLICA

FACULTAD DE INGENIERÍA

Inteligencia Artificial aplicada al reconocimiento de crímenes

Proyecto de grado presentado a la Facultad de Ingeniería de la Universidad de la República en cumplimiento parcial de los requerimientos para la obtención del título de Ingeniería en Computación.

Autores
Rafael Castelo
Manuel Rodríguez
Diego Wins

Tutores
Omar Viera
Sandro Moscatelli

Tribunal
Libertad Tansini
Monica Martinez
Pablo Rebufello

Facultad de Ingeniería, Universidad de la República.
Montevideo, Uruguay.
Febrero de 2023.

Resumen

Este proyecto busca dar respuesta a la siguiente pregunta: ¿es posible detectar un acto criminal en tiempo real en base al lenguaje corporal de un individuo? Para responder a tal interrogante, se propone el desarrollo de un sistema informático capaz de detectar violencia a través de imágenes de video, apoyándose en la amplia variedad de *hardware* disponible dedicado a la recolección de imágenes de video en la vía pública en las ciudades actuales.

Para el reconocimiento de acciones violentas se recurre a los últimos avances a nivel de procesamiento de imágenes, así como también a el estudio de las Ciencias Sociales. De las Ciencias Sociales se busca obtener toda la información posible con respecto al estudio del comportamiento humano a través del lenguaje corporal, en un intento por obtener un marco teórico que pueda ser luego plasmado en código informático. En el segundo caso, se estudian las capacidades más avanzadas con las que cuenta la tecnología actual en cuanto al procesamiento e interpretación de imágenes en una secuencia de video, con el objetivo de obtener información sobre qué puede "ver" un programa informático en ellas.

Dado que en el ámbito de las Ciencias Sociales, la interpretación de las acciones o emociones de una persona a partir de su lenguaje corporal es un tema bastante debatido y lejos de estar cerrado, la solución a este problema se apoya fuertemente en el sector tecnológico. Se elabora un sistema basado en *Machine Learning*, que es entrenado utilizando un *dataset* de videos de hechos violentos ocurridos en la vía pública. Se desarrolla en primera instancia un sistema basado en un modelo de reconocimiento de imágenes llamado *Inception V3*, que obtiene resultados prometedores. Posteriormente, se ve la oportunidad de buscar una solución completamente propia, desarrollando el sistema completamente desde cero, para compararlo con el modelo basado en *Inception*. Este sistema se basa en la utilización de Redes Neuronales y, en particular, de Redes Neuronales Recurrentes de tipo *Long Short-term Memory*, que permiten procesar secuencias de datos a través del tiempo.

Se obtienen resultados positivos, con un *f1-score* de 0.87 para la solución basada en *Long Short-term Memory*, contra un valor de 0.80 de la solución basada en *Inception*. Estos resultados son considerados positivos para los recursos disponibles en el desarrollo de este proyecto.

Índice

Siglas

API *Application Programming Interface*. 40

ERD *Encoder Recurrent Decoder*. 19

GMCP *Generalized Minimum Clique Graphs*. 18
GMM *Gaussian Mixture Models*. 12
GMOF *Gaussian Model of Optical Flow*. 12, 13

HIK *Histogram Intersection Kernel*. 11
HOF *Histogram of Optical Flow*. 13
HOG *Histogram of Oriented Gradients*. 13, 17
HOHA *Hollywood Human Action*. 18

KDE *Kernel Density Estimation*. 18
KNN K-Nearest Neighbours. 18

LRCN *Long-term Recurrent Convolutional Networks*. 13
LSTM *Long Short-Term Memory*. 8, 11–13, 18–20, 42–44, 49, 50

METT *Micro Expressions Training Tool*. 30
MoSIFT *Motion Scale-Invariant Feature Transform*. 11, 13

OF *Optical Flow*. 12, 14
OHOF *Orientation Histogram of Optical Flow*. 13

RBF *Radial Basis Function*. 11
ReLU *Rectified Linear Units*. 44
ResNet-50 Residual Learning Net 50. 18
RGB *Red, Green and Blue*. 14
RLVS-2019 *Real Life Violence Situations Dataset 2019*. 12
RNC Redes Neuronales Convolucionales. 11–15, 19, 20, 35, 36, 38
RNR Redes Neuronales Recurrentes. 13, 15, 19, 20

STIP *Space Time Interest Points*. 11
STM *Selective Transfer Machine*. 16, 17
SVM *Support Vector Machines*. 11, 17–19

VGG16 Visual Geometry Group 16. 12, 18

ViF *Violent Flows*. 13

YOLOv3 *You Only Look Once, Version 3*. 14, 15

1. Introducción

1.1. Motivación

Las sociedades del mundo actual se encuentran en lucha constante por combatir los actos criminales de diversa índole que en ellas ocurren. Como ocurre en muchos otros aspectos de la vida moderna, la tecnología se ha filtrado en esta lucha contra el crimen [1] [2], ampliando y mejorando las capacidades de quienes llevan adelante esta tarea. Sin embargo, esto no implica que las soluciones puestas en práctica actualmente sean óptimas, o que las capacidades actuales de la tecnología estén completamente aprovechadas.

Las ciudades modernas cuentan con una enorme dotación de hardware instalado en la forma de cámaras de videovigilancia [3], que permite tener a disposición imágenes en tiempo real de lo que sucede a lo largo y ancho de las mismas, así como también la capacidad de almacenarlo para futuro estudio. Sin embargo, en los casos donde estos videos son "procesados" por humanos: son las personas quienes deben revisar las imágenes capturadas y analizarlas para obtener información útil a partir de ellas. Si un criminal comete un atraco en la vía pública, alguien debe estar supervisando las transmisiones de las cámaras para detectar este hecho en tiempo real. Es evidente que esto puede provocar un desacoplamiento entre la ingente cantidad de información que proveen estas cámaras y la capacidad de procesamiento de las mismas que pueden llevar a cabo las personas encargadas de ello: un operador sólo puede atender a una cámara a la vez; además, la capacidad de un humano de analizar video almacenado es lineal, dado que por cada hora de video debe dedicar una hora de su tiempo a analizarlo. Por lo tanto, es posible creer que hay un margen de mejora en el proceso de análisis de las imágenes de video mediante la aplicación de la tecnología.

Es deseable aplicar la tecnología también a la hora de procesar los datos crudos captados por la infraestructura existente, de manera que el volumen de los mismos sea más manejable y la información mejor aprovechada. Quizás la intervención humana no pueda ser evitada, pero al menos puede ser focalizada de manera que provea mayor valor en el combate al crimen. Es aquí donde se encuentra la motivación principal de este proyecto: la aplicación de tecnología al proceso de detección de crímenes haciendo uso de la infraestructura ya instalada en las ciudades actuales, en la forma de cámaras de videovigilancia en la vía pública.

Investigando las posibilidades para una solución, dados los avances de la tecnología y la Ciencia Social [4], se busca arribar a un producto que permite detectar actos criminales mediante imágenes de videovigilancia. Las Ciencias

Sociales llevan décadas estudiando el comportamiento humano, por lo que se busca ver cuáles son los últimos avances al respecto, para saber si es posible aplicarlos para comprender el comportamiento de los individuos registrados por las cámaras de seguridad. A su vez, desde el punto de vista técnico se busca explorar los últimos avances en procesamiento de imágenes mediante el uso de Inteligencia Artificial [5] [6] y todas sus subdisciplinas. Entendiendo los últimos avances tanto a nivel técnico como social, se busca aplicarlos de la manera más efectiva posible para facilitar la tarea humana en la lucha contra el crimen: en última instancia, se pretende que el ser humano se remita simplemente a validar advertencias de un sistema informático ante situaciones anómalas capturadas por las cámaras ubicadas en la vía pública.

1.2. Objetivos y resultados esperados

Este Informe de Proyecto de Grado busca dar cuenta de los conocimientos actuales de las Ciencias Sociales con respecto a la comprensión del comportamiento humano en base al lenguaje corporal, en un intento por reconocer conductas delictivas en la vía pública. Además, pretende arrojar luz sobre los últimos avances en cuanto a procesamiento de imágenes mediante Inteligencia Artificial, de manera que sea posible interpretar conductas humanas y clasificarlas en conductas normales o anormales desde el punto de vista delictivo.

Interesa encontrar qué elementos pueden aplicarse de estas 2 disciplinas (social y tecnológica) a la detección y posible predicción de crímenes. ¿Es posible detectar emociones que permitan anticipar comportamientos violentos a partir del lenguaje corporal? Si es así, ¿es posible crear software que utilice este conocimiento para catalogar secuencias de video? ¿Pueden aplicarse estos conocimientos en conjunto con las técnicas de lo que llamamos Inteligencia Artificial? Estos cuestionamientos son los disparadores de este proyecto, que busca respuestas tanto a nivel teórico como a nivel práctico.

Una vez se alcanza un nivel suficiente de conocimiento a nivel teórico sobre los campos ya mencionados, el foco pasa a un terreno más práctico: desarrollar un software que permita aplicar los conocimientos obtenidos en casos reales, para determinar qué tan lejos es posible llegar utilizando los recursos disponibles por parte del equipo. Sin tener a priori una idea de qué tan factible sea esto, el objetivo final es contar con un sistema informático capaz de emitir alertas cuando una acción delictiva sea ejecutada o, si fuera incluso posible, antes de que la misma se lleve a cabo. Se entiende que cuanto antes sea reportado un hecho delictivo, mayores serán las posibilidades de evitar daños o de capturar a los delincuentes.

Es de suma importancia contar con datos de calidad a la hora de llevar adelante

cualquier ejercicio práctico por parte del equipo. En este caso, estos datos deben venir en la forma de sets de datos que contengan distintas secuencias de videos con imágenes de la vía pública. Parte fundamental del proyecto es procurar un conjunto de datos de calidad o construirlo utilizando los recursos que se encuentren disponibles. En cuanto a esto, y siendo que este proyecto se lleva a cabo en la ciudad de Montevideo, Uruguay, donde se cuenta con un amplio abanico de cámaras de vigilancia en la vía pública, resulta interesante tratar de procurar datos similares a lo que éstas puede proveer, de manera que el proyecto desarrollado pueda ser de utilidad en el entorno local.

En las investigaciones llevadas a cabo se logra conseguir un set de datos gratuito con un gran conjunto de videos de las características requeridas. Los videos están segmentados de acuerdo al tipo de acción que en ellos se observa, siendo varias de estas categorías pertenecientes a tipos de delitos de los que aquí interesa evaluar. Más allá de eso, en diversas ocasiones los videos contienen demasiado "ruido" para el tipo de uso que este proyecto requiere, por lo que gran parte del trabajo de este proyecto consiste también en la depuración de estos datos, con el fin de adaptarlos al caso de uso requerido. La depuración de estos datos también permite mejorar los desarrollos llevados a cabo, dando resultados superiores a aquellos vistos con los videos "crudos".

Durante el transcurso del proyecto se desarrollan 2 soluciones diferentes para el problema planteado en este informe, consistiendo ambas en un software de reconocimiento de acciones delictivas utilizando como insumo imágenes de cámaras de videovigilancia como las que pueden encontrarse en la vía pública en cualquier gran ciudad. La primera de estas soluciones se basa en algunos desarrollos preexistentes, que fueron utilizados para cumplir con el fin antes mencionado. Principalmente, se apoya en la utilización del modelo de reconocimiento de imágenes *Inception V3* [7], que ha demostrado gran precisión sobre algunos *dataset* muy conocidos.

A pesar de contar ya con una solución que otorga resultados razonables, durante la búsqueda de una mejora se exploran otras técnicas, que derivan en la implementación de una segunda solución. Ésta ya no cuenta con el respaldo de *Inception V3*, sino que implementa un tipo de Red Neuronal conocido como *Long Short-Term Memory* (LSTM) [8], que permite establecer una vinculación temporal entre la información obtenida en cada fotograma de video. Esto, junto a la utilización de otros tipos de Redes Neuronales y otras técnicas explicadas en la sección 4 de este informe, da lugar a la nueva solución.

Los resultados obtenidos demuestran que la posibilidad de detectar crímenes mediante sistemas informáticos automatizados vía *Machine Learning* es real, con un razonable nivel de certeza, incluso bajo los acotados recursos con los que este

proyecto cuenta. Se estima que el margen de mejora es muy amplio, dado que existe la posibilidad de invertir una cantidad de recursos mucho mayor a lo que aquí se hace. Aún así, se sostiene que la supervisión humana sigue siendo necesaria.

1.3. Estructura del documento

En la Sección *"Revisión de antecedentes desde el punto de vista técnico"* se trata de dar un resumen de las posibilidades que ofrece la tecnología actual en cuanto a al procesamiento de imágenes y reconocimiento de acciones en ellas, en particular aquellas que implican violencia.

En la Sección *"Revisión de antecedentes desde el punto de vista de las Ciencias Sociales"* se hace un recuento de lo que la academia ha estudiado en cuanto al reconocimiento e interpretación del comportamiento humano a través del lenguaje corporal, con el objetivo de entenderlo mejor y buscar aplicaciones prácticas de estos conceptos para la resolución del problema central de este documento.

En la Sección *"Solución desarrollada"* se explica el desarrollo llevado a cabo, definiendo no sólo su estructura, sino que también explicando los conceptos teóricos claves detrás de cada parte del desarrollo y cada decisión tomada. Allí se pueden comparar las dos soluciones propuestas y los distintos enfoques de cada una de ellas.

La Sección *"Experimentación"* contiene los resultados experimentales obtenidos tras poner a prueba las dos soluciones desarrolladas. Se brinda información sobre los resultados individuales de cada una de ellas, así como también explicaciones para los números observados y comparaciones entre ambos resultados. Se habla de los problemas encontrados, de las soluciones a los mismos en aquellos casos en que esto fue posible, así como también de posibles ideas para solucionar problemas que no fueron completamente resueltos en el alcance de este proyecto.

En la Sección *"Conclusiones y trabajo futuro"* se presentan las conclusiones finales del equipo, repasando los objetivos iniciales y comparándolos con los logros obtenidos tras el desarrollo del proyecto. Se proponen también líneas futuras de investigación, sobre todo en aquellos aspectos en los que el equipo se vio más motivado pero a la vez limitado en cuanto a los recursos disponibles.

La Sección *"Bibliografía"* concentra las referencias bibliográficas a las que se hace referencia a lo largo del documento. Junto a ellas están algunos anexos, que contienen detalles relevantes del proyecto, pero que se omiten en el cuerpo principal del informe para mejorar su legibilidad y evitar abrumar al lector.

2. Revisión de antecedentes desde el punto de vista técnico

2.1. Metodología

Para la revisión de antecedentes se sigue un proceso tanto para la selección de material como para la clasificación y revisión de este. Comenzando por la selección del material, este proceso fue realizado en dos vías de estudio, una orientada hacia el área de *Social Science*, es decir, estudio del comportamiento humano, lenguaje corporal y reconocimiento de emociones. La otra vía de estudio fue orientada al aspecto técnico/tecnológico del trabajo, abarcando temas como modelos y *frameworks* de detección de violencia, procesamiento de video e imágenes, reconocimiento de patrones relevantes en el área de estudio (armas, comportamientos, etc), entre otros.

Se realizó la búsqueda de material en diferentes bases de datos, utilizando las palabras clave relacionadas con el tópico en cuestión: *real time crime prediction*, *violence detection*, *social engineering*, entre tantos otros términos vinculados a la problemática a tratar.

Se seleccionan los *papers* que a priori parecían relevantes, teniendo en cuenta su título, palabras clave y su *abstract*. De entre estos, muchos fueron descartados por diferentes motivos: ya sea por discrepancias con nuestro caso de uso, dificultad para replicar la experimentación bajo las condiciones actuales, falta de evidencia y/o experimentación, entre otras. Posteriormente se realizó una lectura exhaustiva y detallada de cada uno de los *papers* restantes, con el objetivo de comprender los métodos utilizados, los modelos más eficaces para cada situación, las condiciones de aplicación y el estado del arte en cuanto precisión a la hora de predecir y detectar crímenes en videos de vigilancia.

A continuación, en la sección de antecedentes se expondrán y analizarán los documentos más relevantes, sus principales características y aportes, así como también la manera en que éstos pueden orientar y servir a modo de sustento para el desarrollo de la solución al problema de estudio de este trabajo.

2.2. Antecedentes

En [9] el enfoque principal no es específicamente el de predecir o detectar violencia en general, sino detectar peleas en una secuencia de *frames* con una *accuracy* alrededor del 90 %. Para ello utiliza dos datasets: uno de peleas en partido de Hockey de la NHL (*National Hockey League* de los Estados Unidos) que es de su autoría; y otro de peleas ocurridas en películas de acción. El primero fue utilizado para la etapa de entrenamiento y contaba con 1000 clips de 50 *frames* cada uno, que estaban etiquetados con "Pelea" o "No pelea". El segundo dataset fue utilizado en la etapa de evaluación para comprobar si el modelo efectivamente transfería el reconocimiento de peleas a escenarios diferentes a un partido de hockey. Este último contaba con 200 clips obtenidos de películas de acción (donde 100 contenían una pelea).

Extrajo *features* de los videos a través de *Space Time Interest Points* (STIP) [10] y *Motion Scale-Invariant Feature Transform* (MoSIFT) [11], creando vectores descriptores de cada uno de los mismos para después aplicar la estrategia *Bag of Words* para videos. Esta estrategia consiste en representar a cada video como un histograma de "palabras" pertenecientes a un vocabulario. Este histograma es representado por un vector que luego puede ser procesado y evaluado por un clasificador. Para obtener el vocabulario de palabras, sobre el que se aplicará la estrategia *Bag of Words* explicada anteriormente, se empleara *K-means* [12] sobre una vasta colección de vectores descriptores obtenidos a través de STIP y MoSIFT, con diferentes valores para la cantidad de *clusters* (50, 100, 150, 200, 300, 500, 1000). Una vez se cuenta con el vocabulario, se puede etiquetar cada vector descriptor de un video evaluando la distancia a la palabra más cercana. Entonces, cada video seria un histograma diferente de las "palabras" del vocabulario obtenido anteriormente, representado por un vector. Por último, tan solo resta clasificar dicho vector. En el caso de este *paper* esto se hace a tráves de *Support Vector Machines* (SVM) [13] y con diferentes *kernels* (*Histogram Intersection Kernel* (HIK) [14], *Radial Basis Function* (RBF) y Chi Squared), cuya función es transformar los vectores de tal forma que puedan ser clasificados por SVM, es decir que el conjunto de vectores sea separable por un hiper-plano el cual determinara dos clases diferentes ("Pelea" y "No pelea").

Para videos de Hockey, STIP y MoSIFT tienen un comportamiento similar. Sin embargo, para videos de películas de acción MoSIFT obtiene mejores resultados. Se llega a la conclusión de que se puede obtener un detector de peleas medianamente flexible (si el descriptor lo es) con una *accuracy* cercana al 90 %.

En [15] los autores usan Redes Neuronales Convolucionales (RNC) [16], LSTM y *Feedforward* (una de las primeras variantes de Redes Neuronales [17]) para la

clasificación de videos en las categorías de “Violentos” y “No violentos”. Entre las RNC utilizadas destacan principalmente tres: MobileNet [18], InceptionV3 y Visual Geometry Group 16 (VGG16) [19]. Como entrada de datos se utiliza el set denominado *Real Life Violence Situations Dataset 2019* (RLVS-2019) [20].

Para la fase de entrenamiento del modelo se utiliza la técnica *K-Fold* con un valor de K=10. Esto implica que se divide el dataset en 10 segmentos iguales, donde se usan 9 para entrenamiento y 1 para validación. Este proceso se repite 10 veces, hasta que todos los segmentos son usados para validación. Se utiliza 50 épocas para el entrenamiento de las distintas Redes Neuronales. Una época es considerada como el entrenamiento de una red neuronal utilizando el set de datos completo una única vez. Las épocas pueden estar subdivididas en *batches*, que son partes del set de datos. Cada sesión de entrenamiento utilizando un *batch* se denomina “iteración”. Las técnicas de *Temporal Fusion* [21] buscan mantener el vínculo entre *frames* de un video, pero sin utilizar tantos *frames* simultáneos que eleven mucho los requerimientos de procesamiento. Esto es importante porque en un video los *frames* tienen relación, no pueden ser analizados independientemente. Hay 3 técnicas de *Temporal Fusion*: *Late Fusion*, *Early Fusion* y *Slow Fusion*. Se utiliza RNC para extraer las *features* de los *frames* de los videos. Luego se utiliza LSTM para extraer *features* temporales a partir de las *features* obtenidas de las RNC. Finalmente se utiliza *feedforward* para realizar una clasificación de los patrones.

Los resultados obtenidos son positivos: una *Accuracy* de 0,91 y un *F1-Score* de 0,90 (más sobre este tipo de medidas en [22]). Fueron *Slow Fusion* y MobileNet las que permitieron alcanzar este resultado, que sobrepasa a los números alcanzados por estudios previos sobre el set de datos utilizado.

En [23] se propone un método para la detección de violencia a través de imágenes de cámaras de seguridad, orientado a la adaptabilidad al entorno de las cámaras de seguridad y a la velocidad de clasificación y capacidad de respuesta en tiempo real. Este algoritmo, en lugar de realizar la detección de violencia sobre todo el *frame*, detecta regiones candidatas a contener violencia a través del algoritmo *Gaussian Model of Optical Flow* (GMOF), a partir de lo cual clasifica la violencia únicamente sobre estas regiones.

Este algoritmo, a diferencia del *Gaussian Mixture Models* (GMM) [24] normal, detecta anomalías en el movimiento y no en los píxeles. Además, los *features* de movimiento son extraídos a través de *Optical Flow* (OF) [25] (de allí el nombre GMOF). Es decir, se extraen los *features* de movimiento a partir de OF y se “clusterizan” con GMM, obteniendo así áreas posibles de violencia, mientras se evita analizar todo el *frame* y se analizan solo las regiones candidatas. Esto evita el desperdicio de capacidad de cómputo y aumenta la velocidad de

clasificación del algoritmo.

Se entrenó con 40 clips de dos datasets diferentes (BEHAVE [26] y Crowd Violence Dataset [27]), que conforman un total de 20.000 *frames*. Para el dataset BEHAVE, el método descrito en este *paper* fue el que obtuvo mayor *accuracy* y *area under the curve* [28] con respecto a los métodos *Histogram of Oriented Gradients* (HOG) [29], *Histogram of Optical Flow* (HOF) [30], HNF (HOG y HOF combinados) [31], *Violent Flows* (ViF) [32] y MoSIFT. También obtuvo buenos resultados incluso en el dataset Caviar [33] que no fue utilizado en la etapa de entrenamiento del modelo. Además, presenta resultados aceptables tanto en cámaras 360º, como en diferentes distancias e iluminación, fondos difusos y multitudes. La clave de este método radica en la detección previa de regiones candidatas de violencia mediante la utilización de GMOF, junto con un descriptor *Orientation Histogram of Optical Flow* (OHOF) que es invariante a la rotación.

En el proyecto llevado a cabo en [34] proponen una arquitectura de entrenamiento *end-to-end*: no sólo se entrena un modelo para el reconocimiento de imágenes, sino que se lo entrena para que determine también la mejor representación de éstas. Llaman a esta arquitectura *Long-term Recurrent Convolutional Networks* (LRCN). El modelo no sólo reconoce videos, sino que también debe poder describirlos en lenguaje natural. Destacan que las LRCN permiten propagar el estado durante más tiempo, lo que hace que se puedan vincular *frames* de video en intervalos más grandes de tiempo. Usan un *framework* de *Deep Learning* llamado Caffe [35], que es muy eficiente por ser desarrollado en C++ (aunque con una interfaz en Python) y cuenta con implementaciones para Redes Neuronales Recurrentes (RNR) [36] y LSTM.

El modelo de LRCN involucra la utilización de un extractor de *features* visuales, como lo son las RNC, sumado a un modelo que pueda reconocer dinámicas temporales. El sistema toma los frames de video y los utiliza como entrada de la RNC, que extrae de éstos un conjunto de *features*. A continuación este resultado pasa a ser utilizado como entrada de la LSTM, que esta vez otorga como resultado una predicción de largo variable. El sistema fue probado utilizando el dataset UCF101 [37], que contiene videos categorizados en 101 tipos distintos de acciones humanas, donde en la mayoría de éstas se superan los resultados obtenidos por los modelos de referencia.

El propósito del *paper* [38] es poder reconocer violencia (peleas) a través de cámaras de seguridad en tiempo real. Parten de que, según estudios anteriores, existe una falta de sistemas de seguridad inteligentes, que puedan ser aplicados en el mundo real. Si bien algunos buscan predecir y/o detectar crímenes o situaciones violentas con un nivel de *accuracy* que les permita ser realmente útiles, la mayoría carecen de velocidad y operabilidad, siendo estos los factores claves para un

sistema de seguridad. Dado que su propósito es detectar actos violentos a través de cámaras de seguridad, los autores plantean que fue necesario crear su propio dataset para entrenar el modelo, ya que los más comunes (como *Hockey Fights* y escenas de peleas violentas) no tienen flujos que sean representativos de lo que captaría una cámara, por lo que su rendimiento no es óptimo.

El sistema utiliza arquitecturas del estado del arte para la obtención de *features* espaciales y temporales que son usados para catalogar acciones simples. Posteriormente, las mismas son clasificadas como violentas o no violentas a través de una regresión logística. Las acciones complejas están compuestas por acciones simples: caminar, empujar, etc. Luego, los patrones de comportamiento humano están determinados por acciones complejas. Con el fin de mejorar la velocidad del modelo, se toman las siguientes decisiones:

- Se utiliza una arquitectura *2-stream* [39]: este tipo de arquitecturas usualmente utiliza 2 RNC en el que el dato (en este caso un *frame*) entra por una red dedicada a reconocer los *features* espaciales y otra a los temporales. De esta forma, pueden trabajar con los dos tipos de informacion al mismo tiempo.
- Se integra un sistema de descomposición de acciones complejas usando un enfoque de razonamiento inductivo.
- El dataset creado no sólo tiene videos de cámaras, sino que también crean a partir de eso un dataset usando el espectro de colores *Red, Green and Blue* (RGB), y *Action Silhouettes* [40] que consiste en utilizar siluetas de los individuos y OF (el "action" refiere a la acción que está realizando el sujeto, mientras que la silueta intenta contenter al mismo con la menor cantidad de ruido ambiente posible).

Por último, para el dataset y el modelo en general se definen diferentes clases como: no hacer nada, caminar, correr, empujar, etc.

Buscando un balance entre *accuracy* y velocidad, obtienen una *accuracy* de 81,2 % y un *recall* de 81,7 % (la cual es una medida importante en escenarios de violencia). Utilizan *You Only Look Once, Version 3* (YOLOv3) [41], un sistema de detección de objetos de última generación y *Fast Dense Inverse Search* que es un método de extracción de features relacionados a OF en el que se intenta reducir el tiempo de cómputo manteniendo una *accuracy* competitiva. [42] que fueron muy útiles para estos resultados y para trabajar el dataset de los tan útiles *action silhouettes*. Además, se utiliza *LiteFlowNet* [43], una OF-RNC que ayuda para detectar la intensidad de las acciones.

En [44] se tiene por objetivo reconocer actos de violencia, usando como dataset *Hockey Fights*, que es una recopilación de peleas de Hockey, ya que es bastante común que ocurran actos violentos en el medio de un partido. Los autores introdujeron RNC de 3 dimensiones para su modelo. Esto no quiere decir que sea

una sola de 3 dimensiones o varias de éstas conectadas, sino que usan las tradicionales junto a la de 3 dimensiones. Es de especial interés el trabajo realizado en busca de la mejor forma de combinar las RNC convencionales con la de 3 dimensiones para encontrar la solución que mejores resultados otorga. Además de Hockey Fights, deciden usar UCF101 [37]. Se entrena con el este mismo y luego hacen un ajuste final con el de *Hockey Fights*.

Consiguen resultados muy positivos, pero también llegan a una conclusión interesante: a pesar de que a más *frames* hay más *accuracy*, el modelo empieza a tardar exponencialmente más tiempo en conseguir un resultado. Este balance entre calidad de los resultados y desempeño no es trivial y debe ser evaluado para cada caso particular. No obstante esto, consiguen más *accuracy* y velocidad que sus predecesores. Sin embargo, hay que ser cuidadosos con estos dataset, ya que hemos visto previamente en [38] que los mismos no son ideales para reconocer violencia tan genérica como es el caso de la calle. De todas formas, la estructura de su RNC es muy interesante.

En [45] apuntan a reconocer armas en tiempo real utilizando la libreria YOLOv3. El proceso es el siguiente: entra un video, se procesa por *frames*, a estos *frames* se les hace un pre-procesamiento y luego se pasa al reconocimiento de objetos. Para esto último el modelo define una *bounding box* que es donde está incluido el objeto. Una vez que tienen la *bounding box*, el modelo decide si es un arma o no. Para llevar a cabo esto se utilizan RNC con *feedforward* (es decir con prealimentación, sus nodos no forman ciclos, a diferencia de RNR).

El objetivo final consiste en poder dar una alerta en un sistema de seguridad, identificando el tipo de arma y el lugar donde ésta fue encontrada, para lo que entrenaron reconocimiento de varios modelos de armas en particular. A pesar de sus buenos resultados al momento de reconocer los distintos tipos de armas, la comparación con otras investigaciones desarrolladas no resulta fácil ya que los trabajos citados utilizan distintos sets de datos.

En [46] se utiliza *Local Binary Patterns*, que es una forma de dividir una imagen en secciones y trabajar con el promedio de una característica de la misma (en este caso el color). Esto logra trabajar con una imagen con menos ruido [47] para mejorar la performance de algoritmos de reconocimiento de expresiones faciales en cámaras de baja resolución. El concepto radica en que, en vez de trabajar directamente con los *frames*, se divide la imagen que contiene la cara del sujeto en secciones (la imagen pasa a tener varios cuadrados consecutivos) en el que cada sección se “pinta” con el color más dominante. De esta forma pasan a tener una representación más genérica de la imagen en lo que a resolución se refiere.

Entre los resultados obtenidos se destaca una *accuracy* cercana a 75 % en

resoluciones muy bajas, mientras que los otros algoritmos ni siquiera son útiles para el problema. Así mismo, segmentan los resultados según el tipo de emociones, a fin de ver en qué casos se hace más difícil el reconocimiento. En algunos casos logran resultados iguales o incluso mejores que otros algoritmos de detección, cuando se trata de resoluciones decentes de imagen. Parte de las pruebas fueron realizadas con *ADABoost* [48], que mejora la performance pero es más susceptible al sobreajuste.

Existen, sin embargo, algunos puntos negativos. En algunos casos los dataset utilizados contenían imágenes que no eran lo suficientemente claras para que el sistema pudiera aplicar la técnica de subdivisión y coloreado. Esto ocurre generalmente con las tomas desde arriba y aquellas en que hay un ángulo muy pronunciado con respecto al rostro del sujeto de estudio. Se menciona trabajo futuro a los efectos de mitigar esta situación.

La idea detrás de [49] parte de la base de que los clasificadores genéricos toman *features* tan comunes que luego dificultan la tarea de reconocer sujetos más específicos, lo que deteriora su rendimiento. Se busca entonces encontrar es un punto medio entre clasificadores específicos de personas y clasificadores genéricos, ajustando un clasificador ya entrenado.

De esta forma, crean *Selective Transfer Machine* (STM), que es básicamente un modelo que ajusta un clasificador (el clasificador no tiene por qué ser alguno en particular, puede ser cualquiera). La STM entonces busca re-ajustar los pesos relacionados al muestreo de entrenamiento según lo que es más relevante para cada sujeto de testeo. ¿Por qué es esto una buena idea? Debido a que STM es un modelo no supervisado, no usa instancias etiquetadas y por lo tanto es muy útil a la hora de cambiar el dominio. Se hacen comparaciones tanto con clasificadores específicos como genéricos, donde STM (aplicado a otro clasificador) termina siendo el que obtiene mejores resultados.

Cabe destacar que se basan en reconocer *Action Units*. Éstas son las acciones que puede realizar un humano y que son de interés para conocer su comportamiento. En particular, existen varias relacionadas con las expresiones, que son las que se buscan en el artículo.

A modo de conclusión de la experimentación, los autores llegan a que STM puede obtener mejores resultados que los clasificadores genéricos y acercarse a la performance de un clasificador específico sin serlo. De todas formas, esta es un área de muhco cuidado debido a que podríamos provocar un *overfit* [50] del modelo. Cabe aclarar que el concepto de *overfitting* refiere al caso en que un modelo estadístico se ajusta mucho a sus datos de entrenamiento. Cuando esto ocurre el algoritmo pierde su propósito, dado que no podrá tener un buen rendimiento cuando los datos de entrada difieran de aquellos usados en las fases de entrenamiento. Sin

embargo, cuando hay una baja cantidad de *Action Units* en el conjunto de test, el rendimiento de STM baja considerablemente, por lo que es importante tener en cuenta la calidad de los datos suministrados al sistema. Además, concluyen que puede ser aplicado a otros dominios como objetos y actividades.

En este trabajo del artículo [51] se estudia estado del arte respecto a la detección de crímenes en tiempo real en las cámaras de seguridad de los cajeros automáticos. Habla acerca de los *frameworks* y arquitecturas más utilizados a la hora de tomar decisiones. Se puede ver un patrón general a la hora de procesar los videos y tomar una decisión, el cual cuenta con los siguientes procesos: pre-procesamiento de imagen, detección de objetos en movimiento, reconocimiento facial, reconocimiento de los componentes faciales, detección de la forma y apariencia de los objetos y por último la toma de decisiones para determinar si se está ante presencia de un acto criminal o no. Para cada etapa, ilustró las técnicas utilizadas, nombrando sus ventajas, desventajas y escenarios de aplicación.

Con respecto a las etapas mencionadas, existen algunos puntos a tener en cuenta. Para la detección de objetos en movimiento la técnica más comunmente utilizada fue *Frame Differencing* [52], que consiste en como sugiere el nombre, evaluar la diferencia entre dos frames contiguos para detectar movimiento. Se suele aplicar mejoras para tratar el fondo de las imágenes, ya que estas suelen contar con ruido y son sensibles a los cambios de luz y de foco. Para la detección de la forma y apariencia de los objetos, el método más eficiente fue HOG [29], un descriptor que calcula las ocurrencias de la orientación del gradiente en porciones localizadas de la imagen. Finalmente, para la toma de decisiones se contrastó entre clasificadores entrenados y sin entrenar. Respecto a los clasificadores entrenados destacó SVM, con múltiples métodos para la reducciones de dimensionalidad en los vectores de la entrada. Por otra parte, las redes neuronales convolucionadas *Hourglass* [53] resultan ser útil en escenarios de tiempo real. Éstas son redes convolucionadas que se dividen en diferentes tipos de capas, con diferentes cometidos. Las capas convolucionales se dedican a extraer los features de la imagen, las capas *MaxPooling* remueven el ruido, las capas residuales se encargan de propagar la información hacia redes futuras, las capas *Bottleneck* introducen convoluciones mas simples (1x1 hasta 3x3) para liberar memoria y capacidad de procesamiento y por ultimo las capas de *upsampling* que aumentan el tamaño de la entrada a traves de *Element Wise Adition* (suma coordenada a coordenada). Por otra parte, cuando se cuenta con ruido en las entradas, el clasificador *Random Forest* es altamente robusto para este tipo de casos.

En [54] lo que se busca es predecir crímenes a través de análisis de video, Redes Neuronales *Fuzzy* [55] y mapeo de densidad (o *density mapping*, por su

nombre en inglés). Utilizando los *datasets* VSD2014 [56], *Hollywood Human Action* (HOHA) [57] y HMDB [58]. Primero, se obtienen los diferentes conceptos indicadores de crimen a partir de los *datasets* de entrenamiento y además se genera una matriz de co-ocurrencia entre dichos conceptos. Dado el conjunto de conceptos de tamaño *k* obtenido en la etapa anterior, a la hora de clasificar un video nuevo, el proceso es el siguiente: el video es dividido en *h* clips, en todos los clips se aplican los *k* detectores de conceptos, resultando en un vector de probabilidades *k * h*. Este vector es utilizado para representar un grafo *G*, donde los vértices son los conceptos y las aristas las probabilidades de co-ocurrencia con sus respectivos pesos. La idea es encontrar un subgrafo *G'*, tal que contenga un vértice de cada clip y que maximice la suma de los pesos de las aristas, en otras palabras, un subgrafo que maximice la probabilidad de co-ocurrencia de los eventos de diferentes clips, durante un mismo video. Para esta optimización, es que es utilizado el algoritmo *Generalized Minimum Clique Graphs* (GMCP) [59]. Una vez hecho esto, se cuenta con conceptos indicadores de crimen para cada video. Cabe destacar que este algoritmo obtuvo mejores resultados que SVM y K-Nearest Neighbours (KNN) [60] para dicha tarea. Posteriormente, se evalúan los conceptos obtenidos con un clasificador *Neuro-Fuzzy* [55] que arroja un indicador de incidencia de crimen, el cual refleja la probabilidad de que un crimen suceda en un video en particular. Por último, se simula un área geográfica, donde se asignan "cámaras" en diferentes ubicaciones, las cuales transmiten video. A partir de estos, en tiempo real, se calcula la incidencia de crimen de cada uno de los puntos y esta información es mapeada a través de *Kernel Density Estimation* (KDE) [61] para lograr visualizar áreas con gran inminencia de crimen fácilmente. Las dos primeras etapas (análisis de video e indicadores de incidencia) serán las más relevantes para el trabajo que queremos desarrollar, ya que tiene potencial para predecir el crimen en tiempo real a través del análisis de video. Además utiliza algoritmos efectivos y eficientes para la obtención de indicadores de crimen en videos de cámaras de vigilancia.

Los autores de [62] proponen un sistema denominado *E-Police* para detección en tiempo real y predicción de crímenes. Dicho sistema tiene 2 componentes fundamentales: un sistema de vigilancia y notificación automatizado que utiliza *Deep Learning* [17] y un sistema de predicción de crímenes basado en *Machine Learning* [63]. Se clasifica el comportamiento humano visto en los videos en base a la extracción de *features* de los *frames* de éstos. Se extraen las *features* de los *frames* usando modelos pre-entrenados que representan el estado del arte, como: VGG16, InceptionV3 y Residual Learning Net 50 (ResNet-50) [64]. Estas *features* de alto nivel se usan como entrada en una red LSTM para hacer las clasificaciones finales del comportamiento humano. Además, se recurre a la

utilización de RNR para el manejo de información temporal en los videos, ya que las RNC sólo pueden extraer *features* de los *frames*, pero no entienden la secuencia de tiempo de éstos, como sí lo hacen las RNR.

Para la predicción de crímenes se recopila información de diversas fuentes, como sitios de noticias. Se usan algoritmos de clasificación como: SVM, *Decision Tree* [65], *Random Forest* [66] y Regresión Logística. Este es un enfoque algo más simple, ya que utiliza información de crímenes pasados para entrenar modelos que así permitan estimar dónde y cuándo puede ocurrir un próximo crimen. No utiliza, sin embargo, información de videos para realizar dichas predicciones, como sí lo hace al momento de detectar en tiempo real los hechos delictivos.

En [67] se propone la utilización de un modelo *Encoder Recurrent Decoder* (ERD), utilizado para el reconocimiento y predicción de la pose de un cuerpo humano. El sistema consta de una RNR, así como de *nonlinear encoder and decoder networks*. De esta manera se busca reconocer y predecir el movimiento de un humano, partiendo de 2 dominios diferentes: *motion capture* (conocido también como "mocap") y secuencias de video. Utilizan lo que ellos denominan como un enfoque más "Lagrangiano", donde tratan de predecir el futuro en base a la trayectoria previa del objeto, en lugar de fijar la atención en un conjunto específico de píxeles (califican a este último enfoque como "Euleriano"). Afirman que gracias a esto se explota mejor la historia visual del objeto para la predicción, evitando además los problemas que tiene el otro enfoque cuando hay movimientos de cámara, por ejemplo.

Para *mocap*, el encoder y el decoder son *Multilayer Fully Connected Networks*. Para el etiquetado de poses en video el *encoder* es una RNC inicializada por un detector RNC por *frame* de las partes del cuerpo y el *decoder* es una *Fully Connected Network* [68]. La principal ventaja de las ERD es que aprenden simultáneamente la mejor representación para el reconocimiento y predicción, así como también las dinámicas propias de estudio, dado que entrenan conjuntamente el *encoding*, *decoding* y la RNR.

Para el desarrollo del sistema hacen uso del dataset H3.6M [69], enfocado en videos de poses. Éste es el *dataset* más grande de videos de poses actualmente disponible. Se generó utilizando actores profesionales a los cuales se registró realizando distintas actividades con un sistema de captura de movimiento. Los autores afirman haber demostrado que la utilización de ERD con *encoder* y *decoder* no lineales es superior a los sistemas multicapa basados en LSTM, que según ellos no producen resultados realistas más allá de cortos períodos de tiempo.

2.3. Conclusiones

Esta revisión de antecedentes ha marcado algunas tendencias en lo que se refiere a la temática de este informe. En primer lugar, la detección de crímenes mediante imágenes puede ser abordada utilizando un amplio abanico de técnicas. No existe un método notoriamente superior a los demás, dado que la investigación al respecto aún parece estar en fases tempranas de su desarrollo. Sin embargo, algunas tecnologías parecen ser un punto común entre varios de los trabajos citados: RNC, RNR y LSTM son los ejemplos más claros.

Por otro lado, cuando el objetivo es anticiparse a los actos criminales en lugar de simplemente detectarlos en tiempo real, la tendencia indica que la utilización de modelos entrenados en base a datos de criminología específicos de cada región es la solución más utilizada. Si bien algunos de los trabajos citados intentan anticipar *frames* de videos en base a los precedentes, las soluciones no han demostrado poder anticipar más que unos pocos *frames* en el futuro. No obstante esto, algunas de las ideas basadas en detección de armas de fuego u otros patrones conocidos de delincuencia sí suponen una apuesta interesante a fin de predecir un posible acto delictivo con la anticipación suficiente como para tener éxito.

3. Revisión de antecedentes desde el punto de vista de las Ciencias Sociales

3.1. Introducción

Buscar anticiparse a un hecho delictivo no es una tarea trivial, pues requiere de un alto grado de comprensión de la forma de actuar del ser humano. De hecho, aún no se ha cerrado el debate sobre la viabilidad de esta idea. Probar que esto sea factible implica la existencia de ciertas características comunes a todo ser humano que revelen, al menos parcialmente, sus intenciones a corto plazo. Éstas deberían ser independientes de todo contexto social en que la persona se haya desarrollado, puesto que de lo contrario no podrá afirmarse que son consecuencia de la propia condición humana. Es aquí donde las distintas corrientes de investigación difieren: algunas sostienen que tales características universales existen; otras sostienen que no es así, y que aquellas características calificadas como "universales" son en realidad dependientes del contexto.

Es de especial interés, por lo tanto, conocer qué dice la academia al respecto. La viabilidad o no del desarrollo de sistemas orientados a prevenir el crimen podría depender mucho de los hallazgos relativos al tema. Existen a día de hoy muchas situaciones en las que ya se utilizan marcos teóricos que determinan la universalidad de ciertas características de los seres humanos, que pueden determinar o no la inocencia de las personas y condicionar sus vidas. Por ejemplo, se conoce que el FBI aplica técnicas de detección de mentiras basadas en reconocimiento de expresiones faciales. Algunos cuestionan estos métodos, sosteniendo que no existe evidencia concluyente y que por lo tanto se están tomando decisiones determinantes para la vida de las personas sin un marco teórico suficiente para justificarlas.

El exponente máximo de este tipo de teorías basadas en patrones de comportamiento universales e inherentes al ser humano es el afamado Paul Ekman, quien investigó durante muchos años las expresiones faciales de las personas y su vínculo con las emociones que éstas experimentan. En base a esto desarrolló un marco teórico de inclinación "Darwiniana", donde vincula expresiones faciales a sentimientos y los adjudica a los rasgos evolutivos del ser humano. Por supuesto, las críticas a su obra no se hicieron esperar, dadas las implicancias que esta teoría supone a nivel social. Es por ello que durante esta sección se dará un repaso a las ideas sostenidas por uno y otro bando, a fin de presentar todos los puntos de vista. El debate aún está vigente, y la academia todavía no ha tomado una postura definitiva al respecto.

Más allá de las corrientes favorables o no a la asociación entre expresiones

y emociones, existen algunos estudios que resultan de particular interés para el enfoque de este documento y que es conveniente mencionar. Estos estudios abordan la situación desde el punto de vista social, no desde el punto de vista tecnológico, como ocurría en el capítulo precedente. Interesa evaluar la teoría social y no la tecnología.

En [70], se explora la posibilidad de que participantes en un experimento puedan reconocer emociones a partir de un dataset auditivo, uno visual y uno audiovisual. También le piden a los participantes que califiquen la intensidad de la expresión que escucharon/vieron. Esto se debe a que dentro de sus dataset hay algunas expresiones "fingidas" o más "exageradas" tanto como espontáneas, por lo que es de interés tener una comparación de cómo los usuarios perciben ambas. Dentro de estos datasets (usan más de uno) hay algunos enfocados al aspecto auditivo y otros al aspecto visual: una recopilación de clips pertenecientes a shows encontrados en YouTube y SoundCloud para el rubro auditivo; el *Geneva Multimodal Expression Corpus (GEMEP)* para la parte visual, que incluye datos de mayor riqueza para el análisis de la comunicación no verbal. Al llevar a cabo la experimentación, los autores llegan a resultados bastante interesantes. Los resultados en cuanto a estímulos visuales y espontáneos reflejan un *accuracy* de 74.17 %, mientras que en estímulos auditivos resulta en un 74.44 %.

Téngase en cuenta que estos son resultados obtenidos por humanos interpretando emociones, no por máquinas programadas para tal propósito. Esto es importante porque busca determinar qué tanta capacidad de interpretación tienen los individuos a la hora de observar expresiones de sus pares. Si los humanos no podemos interpretar emociones, sería difícil programar máquinas que lo hagan, dado que no tendríamos cómo corroborar sus resultados.

En el estudio [71] se explora el sesgo que se genera en las personas para clasificar una acción de un tercero, o qué expectativas se tiene de cómo se comportará el mismo a partir de sus expresiones faciales. El experimento reúne varios participantes y les presenta con un video (animación) con un agente que tiene una expresión facial que se mantiene durante todo el video y donde se pueden presentar los siguientes escenarios:

- El agente tiene una expresión facial en la que parecería estar enojado, acto siguiente da un golpe con su puño cerrado hacia la pantalla en dirección a el observador.
- El agente tiene una expresión facial en la que parecería estar enojado, acto siguiente acerca su puño cerrado a la pantalla, haciendo un saludo de puño hacia el observador.
- El agente tiene una expresión facial en la que parecería estar feliz, acto siguiente

da un golpe con su puño cerrado hacia la pantalla en dirección a el observador.

- El agente tiene una expresión facial en la que parecería estar feliz, acto siguiente acerca su puño cerrado a la pantalla, haciendo un saludo de puño hacia el observador.

También se trabajó con el largo de los videos, teniendo como posibilidad un video corto (termina 10 *frames* antes) un video de duración media (termina 5 *frames* antes) o largo. El resto de las variables permanecen igual en todos los escenarios: el agente, su ropa, la ubicación del agente, etc. Los participantes deben entonces, luego de visualizar el clip, calificar si lo que observaron fue un golpe por parte del agente, o un saludo de puño. Terminada la experimentación, los autores hallan que si bien la expresión facial permite dar una pista al usuario de qué tipo de acción puede estar por realizar el otro (una expresión que transmite enojo puede llevar a una pelea, por ejemplo) los participantes terminan guiándose más que nada por los movimientos que hacía el agente cuando estaba a punto de realizar la acción en cuestión. Sin embargo, cabe destacar que la expresión facial de todas formas influenciaba la decisión de respuesta de los participantes, principalmente en aquellos que les tocaba un video de mediana o corta duración. En conclusión, esto nos lleva a pensar que si bien las expresiones faciales influencian en cierta manera a la otra persona, no queda tan claro qué tanto tanto peso tienen en ello.

El artículo [72] investiga la relación entre expresiones faciales y emociones tomando como participantes a preescolares. Parten de estudios anteriores en los que se aplicaba alguna de las siguientes técnicas:

- Mostrarle al niño una expresión facial y que éste defina qué emoción se corresponde con la misma.
- Describir al niño una breve situación (como un cuento por ejemplo) y luego preguntarle a éste qué expresión facial representa mejor al protagonista de la misma.
- Seleccionar una emoción y que el niño haga una expresión facial acorde.

Este estudio realiza entonces una experimentación multimetódica, mezclando las técnicas previamente mencionadas. Su objetivo es explorar las relaciones entre las tareas de comprensión de las emociones dentro de los mismos niños. Luego de la experimentación se concluye que, a pesar de que conseguían buenos resultados en las pruebas, en algunos era difícil encontrar una correlación entre ellos. Concluye que esto se puede deber a el proceso de experimentación y la comodidad de los niños a la hora de participar, pero también plantea algunas dudas entre emociones y expresiones faciales.

Casos de éxito en los estudios mencionados podrían derivar en utilización de

sistemas informáticos dedicados a la interpretación emocional, como ocurre en [73], donde los autores terminan obteniendo un *accuracy* de más del 90 %. Y allí se plantea también el problema de qué *features* utilizar para medir esas emociones, como tratan los autores de [74], investigando 25 diferentes estudios y obteniendo finalmente 18 features, que creen son las más útiles para reconocer emociones según los estudios.

Son este tipo de estudios los que pueden arrojar luz sobre qué tan cerca está la ciencia de obtener resultados concluyentes acerca de la capacidad del ser humano de expresar e interpretar emociones a través de sus gestos. Esto puede abrir la puerta para utilizarlo en función de la prevención de actividades consideradas dañinas para la sociedad, dado que podría inferirse el estado emocional de una persona antes de que ésta tome una acción concreta. Sin embargo, para determinar la validez de estas investigaciones se debe saber qué teorías las respaldan, qué metodologías aplican y así comprender mejor la implicancia de sus resultados.

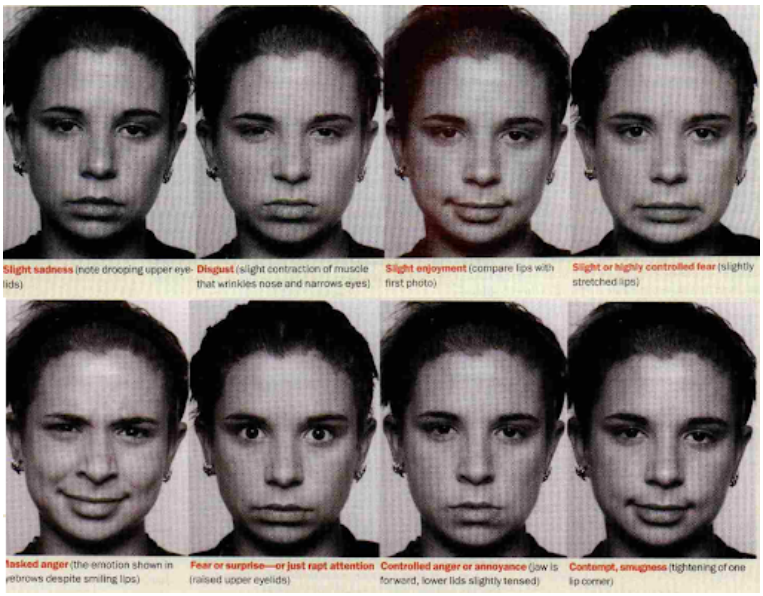
3.2. Paul Ekman y las expresiones como reflejo de las emociones

En [75] el psicólogo Paul Ekman explica que existen expresiones faciales involuntarias y de muy corta duración (entre 1/24 y 1/5 de segundo) que pueden llegar a reflejar las emociones que está sintiendo un individuo en dicho momento, llamadas microexpresiones. Estos tipos de manifestaciones por parte de los individuos son involuntarios y muy difíciles de controlar o reprimir. Es más, algunas de estas microexpresiones son incluso difíciles de reproducir voluntariamente. Según Ekman, solo el 10 % de las personas puede mover las esquinas internas de las cejas hacia arriba voluntariamente y solo el 15 % de los que intentan fingir tristeza logran que las cejas, los párpados y las arrugas de la frente se sitúen en las posiciones correctas.

Sostiene Ekman que existen 7 grandes emociones generales que son expresadas a través de las microexpresiones: asco, tristeza, alegría, enojo, desprecio, miedo y sorpresa. Además, es común ver combinaciones de varias de ellas, como miedo con sorpresa o asco con enojo, por nombrar algunas. Asimismo, el autor indica que estas emociones son universales y trascienden a cualquier raza, religión, lenguaje o grupo social. Entiende que las emociones *"nos preparan para manejar situaciones importantes sin siquiera tener que pensar en ellas, por lo tanto éstas son una ocurrencia inesperada, que los individuos no eligen sentir, sino que simplemente suceden"*. [76]:

A modo de ejemplo, a continuación se muestran un conjunto de las microexpresiones más comunes, tomadas de [77]

Si bien el principal propósito del estudio de las microexpresiones es el de



poder detectar si una persona está diciendo la verdad o no, en el caso de predicción de crimen puede ser útil a la hora de entender las emociones de los individuos y comprender mejor su comportamiento, lo cual puede llevar a anticiparse a algunos hechos antes de que estos sucedan. Ekman sostiene que la clave para detectar mentiras es prestar suma atención a estas microexpresiones y detectar las emociones presentes en un individuo, para a partir de allí analizar el escenario en un conjunto, evaluando la congruencia de estas emociones entre sí, sumado a lo que se está diciendo o haciendo y el rol que juega el contexto (ambiente, gestos y costumbres del individuo, etc).

Además, como es de esperarse, ni los mejores profesionales en la detección de mentiras logran acertar con una probabilidad del 100 %, siempre existen errores humanos y personas difíciles o imposibles de "leer". Tal es el caso de los psicópatas, que muchas veces no sienten culpa por estar mintiendo o incluso logran creerse su mentira, o los actores, que logran compenetrarse en un papel y tienen talento nato para fingir emociones. También está el caso de individuos que dicen la verdad pero que son traicionados por su temor a ser acusados, agregando nerviosismo, temor e incluso enojo en sus relatos sin que esto implique que están faltando la verdad, sino todo lo contrario. Por lo tanto, hay que ser cuidadosos a la hora de tomar en cuenta las microexpresiones para realizar un juicio sobre la veracidad de un discurso o sobre las intenciones de un individuo. Según Ekman, es fundamental contar con personas entrenadas en la lectura e interpretación de estas manifestaciones.

Para las personas es altamente dificultoso detectar microexpresiones en tiempo

real debido a su corta duración. Es por esto que sería razonable contar con la ayuda de la inteligencia artificial para detectar las microexpresiones y catalogar las emociones que estas representan. En los artículos [78], [79], [80] se trabaja en el reconocimiento automático de microexpresiones y las emociones asociadas a ellas, siendo capaces de llevar a cabo la tarea con un resultado aceptable. Además, en [81] se experimenta el reconocimiento de microexpresiones unido a la detección de mentiras a través de la inteligencia artificial. Para realizar el experimento tomaron video clips del programa *"The Moment of Truth"*, en el cual los invitados responden preguntas acerca de su vida personal y son descalificados al mentir. Los autores parten de la hipótesis de que la microexpresión del miedo esta fuertemente relacionada con la mentira, por ende analizan las microexpresiones de miedo encontradas en los clips. Se logro detectar la mentira con una efectividad superior al 80 % y, además, se encontró una relación entre la duración del miedo y las simetría de la cara con la veracidad en el discurso de un individuo. Un punto importante a aclarar sobre este estudio es que la veracidad de las preguntas en el show era ratificada únicamente a través del polígrafo [82]. Este método no es el mas apropiado según Ekman [83], ya que lo que hace no es detectar mentiras sino medir el estrés en el cuerpo a través de diferentes indicadores fisiológicos como: ritmo cardiaco, sudor en las manos, presión sanguínea, respiración, entre otros. Sin embargo, esto no asegura que una persona esté diciendo o no la verdad. Muchos inocentes sufren de alto estrés por miedo a ser juzgados injustamente a la hora de enfrentar el polígrafo y muchos mentirosos no creen en la efectividad de esta herramienta y no sufren las alteraciones fisiológicas típicas del estrés. Por ende, quizás las microexpresiones de miedo en las caras de los participantes estén altamente correlacionadas con el resultado del polígrafo, pero no se podría estar seguro acerca de la veracidad de las respuestas de los individuos.

3.3. Críticas a las teorías que vinculan expresiones a emociones

El artículo *"Emotional Expressions Reconsidered: Challenges to Inferring Emotion From Human Facial Movements"* [84] realiza un meta-estudio, recopilando toda la información disponible hasta el momento con respecto al análisis de expresiones faciales y su vínculo con las emociones que experimentan las personas. El objetivo es arrojar luz a un tema controvertido, ya que sostienen que muchas de las teorías desarrolladas sobre esta temática carecen del debido fundamento científico, pero repercuten de manera muy importante en la vida de las personas. No sólo se busca una mayor claridad, sino que también se pretende sentar las bases para mejorar futuras investigaciones al respecto, estableciendo

fundamentos más sólidos.

Es tan grande el impacto de estas teorías, sostienen los autores, que repercuten en un amplio abanico de facetas de la vida de las personas. Por ejemplo: grandes empresas del rubro tecnológico están invirtiendo ingentes sumas de dinero en sistemas de reconocimiento de emociones basados en la lectura de expresiones faciales; existen formas de comunicación basadas en *emojis*, donde ciertas expresiones faciales ya están asociadas a una emoción en concreto; posters que enseñan a niños en edad pre-escolar el significado de expresiones faciales; shows de televisión que traen a la cultura popular la idea de una correlación perfecta entre emociones y gestos; policía, servicios de inteligencia y hasta jueces que se basan en estas correlaciones entre emociones y expresiones para detener o sentenciar a sospechosos, cambiando rotundamente la vida de una persona; tratamientos para personas con autismo u otras afecciones mentales en los cuales se les enseña cómo percibir las emociones de los demás.

Todo el estudio se basa en el análisis de un grupo limitado de emociones: asco, tristeza, alegría, enojo, desprecio, miedo y sorpresa. Sin embargo, se sostiene que los resultados son extrapolables a cualquier otra posible emoción que pueda sentir una persona. Es así como identificaron tres problemas en la literatura científica respecto a la temática analizada: la capacidad de "lectura" de emociones es limitada, ya que no hay consistencia ni confiabilidad al momento de expresar o percibir emociones; no hay una correlación 1:1 entre expresiones faciales y emociones; no hay suficiente investigación ni documentación respecto a los efectos de la cultura y el contexto en que se producen las expresiones.

Por la evidencia relevada en este informe, se plantea un fuerte debate en cuanto al grado de similitud entre las expresiones asociadas a las distintas emociones. Existe una corriente que afirma que cada emoción está vinculada a un "prototipo" de expresión: existen variaciones en las expresiones manifestadas para cierta emoción, pero son leves y todas se sostienen en un patrón (el prototipo) común que puede ser fácilmente identificado. Por otro lado, existe otra corriente de investigación que plantea que el grado de variabilidad entre expresiones asociadas a una emoción es mucho más amplio, haciendo que la correlación expresión-emoción sea difícil de establecer, lo cual atenta contra las aplicaciones prácticas de la teoría descritas en los párrafos precedentes. La variación entre expresiones es lo que mantiene vivo el debate científico al respecto en la actualidad.

Para que una expresión pueda ser asociada a una emoción subyacente, dicen los autores, debe cumplir con 4 criterios clave: confiabilidad, especificidad, generalizabilidad y validez. Omitir uno de estos criterios es motivo suficiente para abstenerse de afirmar que la expresión observada manifiesta una determinada

emoción. En palabras más sencillas, si uno de estos criterios no se cumple no podemos afirmar que "la persona está contenta porque sonríe", sino que simplemente debemos detenernos en "la persona sonríe". Sostiene el estudio que el incumplimiento de esta consigna es uno de los principales causantes de la confusión a nivel público respecto al tema de estudio.

En el documento se relevan una amplia gama de estudios relativos a la forma de expresar las emociones mediantes expresiones de diversos grupos de individuos. En primer lugar, adultos sanos en países del primer mundo, segmentados en tres grupos: condiciones de laboratorio, situaciones naturales del día a día y situaciones en las que a las personas se les pedía que posaran con una expresión relativa a una emoción (ejemplo: "ponga cara de felicidad"). De este conjunto de estudios concluyen que hay una falta de especificidad y confiabilidad. Luego continúan con estudios sobre adultos que viven en culturas remotas y de pequeña escala, que segmentan en observaciones naturales en el día a día y estudios donde las personas debían posar con una expresión concreta, como en el caso anterior. Encuentran aquí una falta de datos debido a que no existe una observación sistemática y controlada de la forma en que las personas utilizan sus expresiones faciales al experimentar distintas emociones. También recogen un grupo de estudios vinculados a niños sanos, donde se estudian los movimientos faciales que van desde fetos hasta niños jóvenes. Con respecto a estos también encuentra que no existe suficiente evidencia que permita cumplir con los criterios de confiabilidad y especificidad. Sostienen que se necesita información adicional del movimiento corporal y los sonidos vocales emitidos por el niño para poder inferir emociones. Además, encuentran que las mismas emociones son expresadas con distintos movimientos faciales y los mismos movimientos faciales se usan en distintas expresiones. Se plantea como pregunta abierta que la confiabilidad y especificidad pueda lograrse a través de un proceso de aprendizaje, pero la respuesta requerirá mayor estudio. En un último grupo se ubica a los estudios referentes a personas invidentes o con severos problemas de visión, que fueron estudiadas para comprobar si las expresiones faciales son aprendidas a través de la observación del contexto por parte de los humanos. Pero de nuevo los datos no permiten cumplir con la especificidad y confiabilidad necesarios para dar una respuesta contundente a favor de que las expresiones faciales reflejan emociones. Concluye el estudio que la evidencia en cuanto a la producción de expresiones faciales en base a emociones no es confiable ni específica, presentando una enorme variación entre expresiones, lo cual contradice las clásicas asociaciones entre expresión y emoción que se conocen en la cultura popular.

Estudiar cómo se expresan las emociones a través de movimientos faciales es sólo una cara de la moneda. Es necesario también conocer cómo hacen los

individuos para interpretar esas señales e inferir a través de ellas las emociones que los demás experimentan. Es decir, como en todo canal de comunicación, no sólo se requiere de una correcta emisión del mensaje, sino de su correcta recepción. Citan estudios de diversa naturaleza para este apartado: etiquetado de expresiones a partir de un conjunto de emociones, etiquetado de emociones libre y correlación inversa (asignación de emociones a una emoción dentro de un conjunto predefinido). En el primer caso los participantes etiquetan expresiones faciales a partir de una "bolsa de etiquetas" de emociones. En el segundo caso, los participantes etiquetan expresiones pero sin partir de un conjunto de emociones predefinido, sino que lo hacen en base a sus propias palabras. El tercer caso es el inverso del primero, donde los participantes tienen un conjunto de emociones a los que deben asignar aquellas expresiones que creen que pertenezcan a un determinado conjunto. Existe un cuarto y último tipo de experimento, algo diferente a los anteriores, en donde se invita a los participantes a interactuar con "humanos virtuales" (rostros generados por animaciones 3D) que están programados para representar ciertas expresiones, teniendo los participantes que interpretarlas.

Los resultados constatados por los autores indican que la confiabilidad con la cual las personas perciben emociones a partir de expresiones faciales depende de qué preguntas se les hace al respecto y cómo se les pide que registren esas percepciones. Se pone especial énfasis en que en la mayoría de los estudios se le pide a los participantes que evalúan expresiones de rostros que posaron de manera exagerada para acentuarlas, mientras que se les da un conjunto muy reducido de palabras (una palabra para cada emoción) de entre las cuales elegir para etiquetar dicha expresión. Si bien estos estudios muestran una evidencia moderada a alta en favor de la teoría dominante (teoría que asocia expresiones faciales a emociones), lo hace sin otorgar muchas posibilidades a que se observe evidencia contraria, dadas las limitaciones recién mencionadas. Tampoco aportan evidencia a favor del cumplimiento del criterio de especificidad. Por otro lado, los estudios de etiquetado libre y correlación inversa aportan, según remarcan los autores, escasa evidencia en favor de la teoría dominante en cuanto a la confiabilidad de las predicciones. Estos hallazgos también fueron confirmados cuando los autores revisaron los mismos tipos de estudios, pero realizados en pequeñas comunidades de sectores relativamente aislados. Más aún, estudios realizados en niños pequeños también validan estas críticas y permiten ver evidencias de que la capacidad de percibir emociones es aprendida y desarrollada mediante la interacción social.

Concluyen los autores de este extenso estudio que tanto en la producción de expresiones faciales en base a emociones como en la percepción de emociones a

través de expresiones faciales se observa la misma tendencia: no existe evidencia concluyente a favor de la teoría dominante. Sólo se observa una excepción en el caso en que se pide a las personas que etiqueten, a partir de un conjunto reducido de opciones, imágenes de rostros que posaron (de manera exagerada) con una expresión determinada.

En [85] se pone en tela de juicio qué tan científicos son los métodos de Ekman, y qué tan útiles pueden llegar a ser las microexpresiones a la hora de detectar las emociones de una persona. Primero hace notar que la efectividad en la detección de mentiras a través de microexpresiones faciales es cercana al azar [86]. Además, Barret sostiene que la metodología en la que Ekman lleva a cabo su estudio esta sesgada. Ekman le otorga a los participantes del estudio un conjunto emociones universales y los induce a que asocien cada cara con alguna de las expresiones provistas anteriormente. De esta forma estaría sesgando a los participantes a detectar emociones dentro de un rango acotado de las mismas. Cuando Barret replica el mismo estudio, mas sin indicar emociones de antemano, se detecta que la capacidad para detectar emociones por parte de los participantes había reducido considerablemente [87]. Por lo tanto sugiere que la teoría de las emociones universales, sobre la cual esta cimentado y desarrollado todo el método de Ekman, es débil y fue sujeta a una metodología defectuosa. Otra de las principales criticas que trae a colación el autor es que no todos los individuos muestran microexpresiones con la misma intensidad ante un mismo estimulo, e incluso, no todos muestran microexpresiones. En su estudio [88] examinaron expresiones faciales reales y falsas. Encontraron que el 100 % de los participantes mostraba inconsistencia interna, reaccionando de formas diferentes ante los mismos tipos de estímulos. También encontraron que solo el 21.95 % mostraba microexpresiones faciales. Tambien se puso a prueba el método de entrenamiento de Paul Ekman, el *Micro Expressions Training Tool* (METT), para la detección de mentiras. El estudio [89], consistía en dividir a un grupo en tres subgrupos, donde uno de ellos iba a ser entrenado con el METT, otro de los subgrupos con un método diseñado por los investigadores a cargo del estudio y el ultimo subgrupo no contaría con entrenamiento. A cada uno de los individuos se les mostrarían videos de personas diciendo la verdad o mintiendo acerca de algún tema y los participantes deberían dar su juicio acerca de si se trata de una verdad o una mentira. Los resultados arrojaron que el METT no es efectivo a la hora de detectar mentiras, ya que los participantes contaron con una eficacia del 46.30 %, siendo este resultado peor que el azar. Cabe aclarar que el método propuesto por los investigadores tampoco prosperó, ya que obtuvo 47.30 % de efectividad.

3.4. Conclusiones

El debate científico acerca del vínculo entre emociones y expresiones está aún abierto. Lo que en algunos lugares es tomado como teoría definitiva es cuestionado en otros, siendo que ni la misma academia ha logrado aún emitir un veredicto firme al respecto. Mientras tanto, las consecuencias de una u otra visión afectan el día a día de muchas personas, dependiendo de la "biblioteca" con la que son juzgadas.

Sin duda alguna la investigación debe avanzar. De confirmarse las teorías favorables, la sociedad podría beneficiarse de sistemas que ayuden a interpretar a las personas y anticiparse a acciones de consecuencias negativas que éstas puedan llevar a cabo. Mientras tanto, de rechazarse esas teorías habría que dejar de utilizarlas para juzgar a individuos que quizás están siendo penalizados por ser evaluados en base a una teoría falsa.

4. Solución desarrollada

Vistos los antecedentes desarrollados a lo largo de las secciones precedentes, se cree que la mejor solución posible al problema de la detección de crímenes pasa por un sistema informático capaz de aprender patrones ocurridos en imágenes de video de escenas que contengan efectivamente actos criminales. Utilizar mecanismos de detección de emociones para intentar anticipar estos hechos no parece la solución más efectiva, dado el intenso debate a nivel académico al respecto, así como también las limitaciones del hardware utilizado en la vía pública, que no siempre permite capturar imágenes con la calidad suficiente para esto. Por lo tanto, se opta por la idea de buscar un set de datos que contenga imágenes de este tipo, para de esta manera seguir los pasos de algunas de las investigaciones mencionadas en el Estado del Arte basadas en Redes Neuronales. Utilizando varios de los conceptos vistos durante esa sección, el equipo investiga aquellos que considera más prometedores y cómo puede apoyarse en ellos para desarrollar las ideas aquí mencionadas.

Durante esta sección se expandirá en los conceptos clave utilizados para resolver el problema central de este informe, por qué y cómo fueron utilizados. El aporte principal de este proyecto consiste en combinar ideas que fueron recabadas durante la fase de investigación en una forma que resultara útil para los objetivos trazados, así como también buscar y transformar los datos necesarios para utilizar como entrada del sistema; este aspecto incluye también el procesamiento de los videos para obtener cada cuadro de los mismos, optimizando este proceso para aportar la mayor cantidad de datos con el menor uso de recursos posibles.

4.1. Requerimientos funcionales y no funcionales

A continuación se presentan los requerimientos funcionales y no funcionales para la solución desarrollada. Los cuales indicaran una pauta a seguir a la hora de la construcción de la solución, así como pautas de medición sobre el cumplimiento de los mismos. Permitiendo dar una idea sobre el éxito de la solución implementada.

Requerimientos funcionales:

- La *precisión* del modelo debe ser superior al 70 %, de manera que presente una mejora sustancial con respecto a una decisión al azar a la hora de marcar una secuencia de video como violenta o no violenta.
- El *recall* deberá alcanzar al menos el 70 %, como el caso anterior, para garantizar que el modelo es capaz de reconocer una porción considerable de las secuencias de video pertenecientes a una determinada categoría.

- El sistema debe ser capaz de funcionar con imágenes de baja resolución, que son las típicamente encontradas en sistemas de vigilancia mediante cámaras de video.

Requerimientos no funcionales:

- La solución debe poder ser ejecutada en *hardware* relativamente accesible, sin requerir desembolsos considerables de dinero.
- Los set de dataos utilizados para el desarrollo del sistema deben ser de libre disposición, al menos para fines académicos, para permitir replicar el proceso de entrenamiento en el futuro.

4.2. Dataset utilizado

Para el desarrollo de la solución propuesta se utilizó el set de datos *Video Anomaly Dection* [90] [91], desarrollado por Waqas Sultani, Chen Chen y Mubarak Shah. Dicho set de datos contiene 128 horas de video distribuidas en 1900 videos diferentes. A su vez, los videos están etiquetados de acuerdo a las siguientes categorías: abuso, arresto, incendio, asalto, accidente vial, robo, explosión, peleas callejeras, disparos, vandalismo, entre otras. Además, existe una categoría de videos catalogados como "normales", donde se encuentras grabaciones de las cámaras de seguridad en circunstancias ordinarias del día a día de una ciudad.

Dicho *dataset* fue elegido debido a estar basado en videos de cámaras de seguridad de lugares públicos, tanto en el exteriores como en interiores. Es la opción que más se asemeja al tipo de escenario objetivo de este proyecto, que son las cámaras de seguridad instaladas en la vía pública por los organismos públicos. Además, y como detalle no menor, cumple con un punto fundamental para este proyecto: ser de uso libre para propósitos de investigación. Disponer de esta cantidad y calidad de datos es indispensable para desarrollar un sistema basado en Inteligencia Artificial, dado que el resultado de los modelos depende fuertemente de los datos utilizados durante el entrenamiento.

Durante la fase de implementación se buscó "jugar" con el *dataset* para obtener el mejor resultado para el modelo desarrollado. Dado que este proyecto pretende implementar un sistema capaz de detectar violencia, la segmentación de los videos según el tipo de acto violento cometido no era relevante. A su vez, algunos de los tipos de delito contenidos en el *dataset* no resultan útiles para tal objetivo. Un ejemplo claro es el caso de los videos de arrestos, dado que en tal caso los oficiales de policía ya están al tanto de que un posible delito ha sido cometido y han actuado,

por lo cual carece de sentido utilizarlos. Se redujo el *dataset* para entrenar el modelo a las siguientes categorías: abuso, asalto, robo, peleas callejeras y videos normales. Cada categoría incluye unos 50 videos, de los cuales se seleccionan 20; tal medida permitió optimizar los recursos con los que el equipo contó para el desarrollo de este proyecto, aunque sin generar un impacto significativo en el resultado final, según las pruebas realizadas.

Durante la fase de experimentación también se realizaron pruebas modificando los videos. En varias ocasiones se encontró que el *dataset* contiene videos de hechos violentos de larga duración, en los cuales el acto de violencia en sí tiene una duración muy breve, por lo que el equipo consideró que esto podía añadir "ruido" a los datos. Por ejemplo, en algunos casos un video de más de 1 minuto de duración contenía un hecho violento que ocurre en un espacio de 10 segundos, por lo que hay un espacio de tiempo muy grande en el que no ocurre nada. Se consideró que todo ese "tiempo muerto" podía confundir al modelo, dado que no difería de lo que se observa en los videos de circunstancias normales (no violentos) del *dataset*. Dada esta situación, también se experimentó realizando una edición de los videos violentos, cortándolos y conservando únicamente la porción de los mismos en la que el hecho violento en cuestión ocurría.

4.3. Procesamiento de los videos

Una vez se cuenta con los videos organizados y divididos de forma adecuada para el entrenamiento y la evaluación del modelo a desarrollar, se requiere de un conjunto de pasos extra, para que puedan ser usados como *input* de Inception V3 [92]. Todo este procesamiento es realizado con la librería de Python OpenCV [93], por ser la que otorgaba mayor facilidad para la tarea, mejor calidad de documentación y acceso a una comunidad [94] suficientemente grande de desarrolladores que dan soporte a la librería.

Primero, cada video es representado por un subconjunto de todos los *frames* de este, a razon de 1 *frame* por segundo de video, con el objetivo de reducir el tiempo y los recursos necesarios para el entrenamiento y a su vez evitar información redundante. Cada frame es redimensionado a un tamaño de 244x244 píxeles, dado que es el tamaño de *frame* requerido para los vectores de entrada de Inception. Cada frame es representado por un vector de dimensiones (244, 244, 3), donde cada posición del vector representa un píxel. El primer valor indica la posición en el eje X del píxel, el segundo valor indica la posición en el eje Y y el ultimo valor, que es un array de 3 dimensiones, denota el color en formato RGB [95] siguiendo la forma [R, G, B].

Por lo tanto, cada video es representado como (n, 244, 244, 3) siendo n la

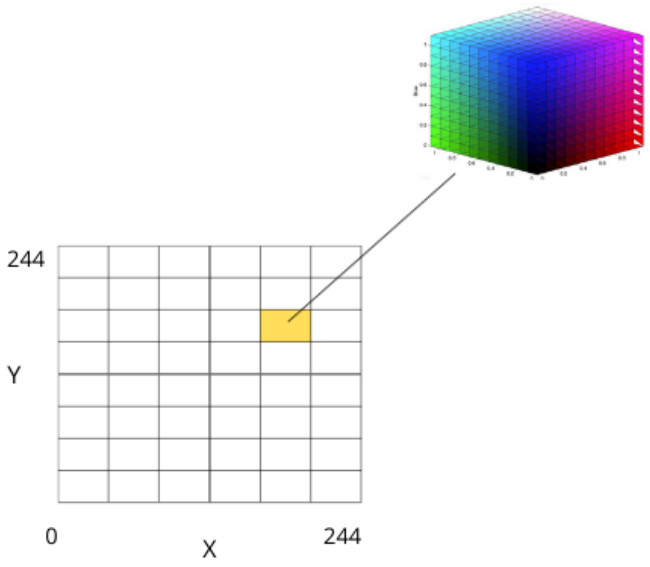


Figura 1. Representación de un *frame*.

cantidad de *frames*. En otras palabras, el video puede verse representado como un vector de matrices de dimensión 244x244x3, donde cada entrada de una matriz representa el color de un píxel de un *frame* en representación RGB.

4.4. Modelo Inception

En el mundo de la Inteligencia Artificial, en particular en el mundo del procesamiento de imagen, varios modelos seguían la practica de encadenar capas de Redes Neuronales, una detrás de la otra en busca de mejorar la performance del modelo. Cuando uno piensa en los modelos de procesamiento de imagen, lo mas común suele ser utilizar RNC y aplicar filtros a las imágenes. Los filtros ayudan a reducir la cantidad de información y de ruido en cada imagen, mientras que las RNC se usan para clasificación de imágenes desde los años 90, debido a su alta precisión [96].

Las RNC y el procesamiento de imagen llevan una estrecha relación, ya que si se usa una red neuronal totalmente conectada, uno de los modelos mas clásicos de *Machine Learning* que permiten aceptar varias entradas (lo que sería adecuado

en este caso, ya que una imagen tiene mucha información) se pierde capacidad de escalado en el sistema, dado el alto coste computacional de dichas soluciones.

Teniendo en cuenta la representación de imágenes RGB mencionada en la sección precedente, considérese como ejemplo el caso de una imagen de tamaño 244x244 píxeles previamente mencionada. Esto significa que, usando una Red Neuronal totalmente conectada, se necesitarán 178.608 (244 píxeles de ancho, 244 píxeles de alto y 3 matrices para la representación RGB) neurones para poder procesarla. Y esto podría ser mucho peor para las resoluciones de imagen manejadas en la actualidad. Por otro lado, cuanto mayor es la cantidad de parámetros de entrada (producto de imágenes con mayor resolución), mayores son los riesgos de caer en *overfitting*.

Por las razones antes mencionadas, se decide utilizar RNC en procesamiento de imagen, cuyo enfoque es algo distinto. Las RNC tienen lo que se conoce como una Capa de Convolución, una Capa de *Pooling* y, por último, la Capa Totalmente Conectada. La Capa de Convolución es la primera en reducir la entrada. La idea es que al llegar a la Capa Totalmente Conectada habrán menos parámetros de entrada sin necesariamente haber perdido información útil para el reconocimiento de la imagen. La Capa de Convolución toma uno o varios filtros, siendo cada filtro representado por una matriz y aplicado mediante producto matricial entre el filtro y la representación matricial de un *frame*. Dado el tamaño de los filtros y el tamaño del *step* (que simplemente indica a qué sectores de la matriz original aplicar el filtro) se reduce el tamaño de la entrada inicial. La Figura 2 muestra un ejemplo de aplicación de 3 filtros (rojo, verde y azul) a una imagen.

Por otro lado, la Capa de *Pooling* toma la matriz que sale de la Capa de Convolución y le aplica el máximo o el promedio a todos los sectores de la matriz. En la Figura 3 se muestra el caso de una capa de *Pooling* de dimensión 2x2 y *step* 2.

Finalmente, se introduce una Capa Totalmente Conectada, que tomará la información "resumida" que viene de las capas precedentes. Está claro que si se tiene una imagen con una cantidad de píxeles muy alta, se puede aplicar el proceso de poner una Capa Convolucional y luego una de Capa de *Pooling* varias veces antes de llegar a la capa totalmente conectada y así reducir el tamaño de entrada de la misma.

Habiendo explicado por qué utilizar RNC para procesamiento de imagen, es importante mencionar también la motivación detrás de utilizar Inception. Según lo visto anteriormente, teniendo imágenes con densidad de píxeles grande, se debe aplicar varias veces la Capa de Convolución y luego la Capa de *Pooling* hasta llegar a la Red Totalmente Conectada. Esto hace que la red crezca en profundidad, por lo tanto también crece en complejidad (agregamos cada vez mas parámetros) y por lo

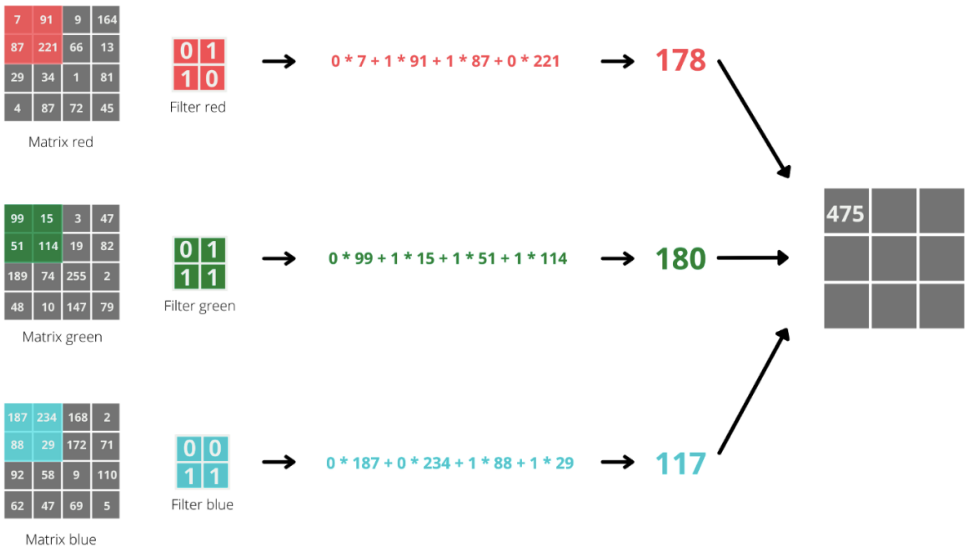


Figura 2. Filtros aplicados a una imagen, dentro de una capa convolucional *Using Convolutional Neural Network for Image Classification* [97]

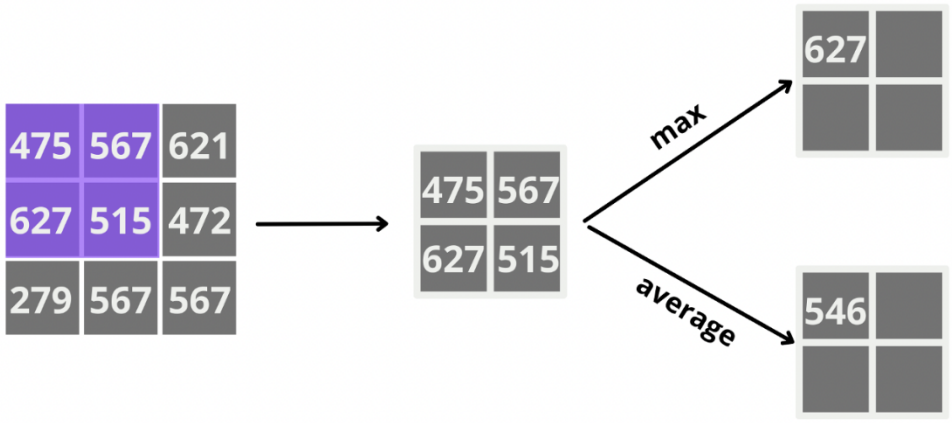


Figura 3. Capa de pooling reduciendo el tamaño de la matriz de el artículo. [97]

tanto se incrementa el riesgo de *overfitting* del modelo. Además, peligra también la eficiencia de éste, ya que agregar capas aumenta considerablemente la potencia computacional necesaria para completar la tarea [7].

El modelo Inception busca cambiar crecimiento en profunidad por un crecimiento más "horizontal", es decir, generar varias capas en paralelo que trabajen al mismo tiempo. Es esto último lo que permite generar un mejor compromiso entre las capcidades del modelo y el coste computacional asociado al mismo. Inception provee así una mejora en rendimiento en comparación con los modelos de convolución tradicionales y un uso más eficiente de los recursos computacionales, entre otras virtudes [7].

Inception introduce capas de convolución 1x1, que aplican el producto entre todos los valores de cada píxel de la imagen. Para el caso de la representación RGB de un *frame* mencionado anteriormente, donde se tienen 3 matrices y algún filtro extra aplicado, el resultado es una matriz de la misma dimensión que las otras pero que en cada celda tendrá el valor: `rojo` * `azul` * `verde` * `filtro`. De esta forma se reduce la cantidad de "canales". Cada canal es una matriz con diferente información. En el caso RGB, son 3 los canales, ya que se tienen tantos canales como filtros se hayan aplicado en esta capa de convolución. La capa de convolución 1x1 tiene usualmente configurada una cantidad reducida de filtros, por lo que sus salidas suelen tener menos cantidad de canales que la entrada. Su objetivo es entonces reducir la cantidad de canales y aprender patrones a través de estos.

Inception introduce también capas de 3x3 y 5x5, que aprenden patrones espaciales a través de todos los componentes dimensionales (alto, ancho y profundidad de la entrada). Estas dimensiones de filtros se dan luego de una investigación que buscaba encontrar el tamaño óptimo de los mismos en RNC para conseguir la performance óptima. Además, al tener estos tamaños ya definidos, se reduce la complejidad (menos parámetros). Por lo tanto, Inception incluye entonces lo siguiente:

- Capa de Entrada
- Capa Convolucional 1x1
- Capa Convolucional 3x3
- Capa Convolucional 5x5
- Capa *Max Pooling* (*pooling* que toma el máximo del sector)
- Capa de Concatenación

A excepción de la capa de entrada y la de concatenación, las otras 4 son independientes entre sí, todas tienen la misma entrada. Luego en la de concatenación se juntan todas las salidas, como lo muestra la Figura 4.

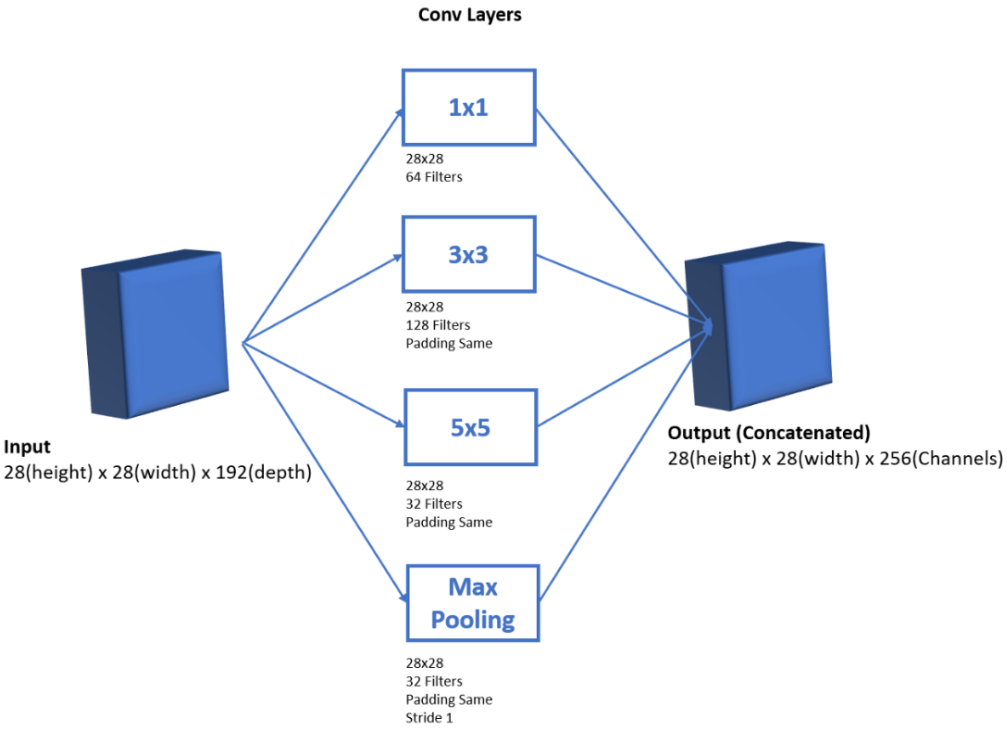


Figura 4. Módulo de Inception ilustrado. [98]

El modelo Inception esta formado por varios de estos módulos conectados secuencialmente [7]. Se han implementado varias versiones de Inception que mejoran el modelo paulatinamente. En particular, la versión 1 introdujo las Capas Convolucionales 1x1 antes de pasar a la de 3x3 y la de 5x5 para reducir el costo computacional. Para el desarrollo de la solución se utiliza Inception en su versión 3, dado que es la que la versión soportada por Keras, librería utilizada para el desarrollo de este proyecto [99].

4.5. Desarrollo del modelo

4.5.1. Modelo utilizando Inception V3

El modelo desarrollado toma como base un modelo anterior [100], el cual es aplicado sobre un dataset diferente. El modelo desarrollado en este proyecto se compone de varias capas, cada una construida sobre la anterior, que procesan los datos datos en forma de *pipeline*: cada capa recibe una entrada, la modifica de acuerdo a ciertos parámetros y emite una resultado que servirá de entrada para la

próxima capa. Estas capas son añadidas sobre un modelo base de *Machine Learning*, que es Inception, el cual ha sido objeto de análisis en secciones precedentes. Cada capa del modelo utiliza una librería conocida como Keras [101]. Keras es una librería de Software basada en TensorFlow [102], pero enfocada en proveer una *Application Programming Interface* (API) más amigable para el usuario final que esta última. Por su parte, TensorFlow es una plataforma de código abierto enfocada a *Machine Learning*, compuesta de decenas de herramientas y librerías.

La decisión de utilizar estas librerías se basa en algunos puntos clave. En primer lugar, el equipo cuenta con experiencia en el uso de estas herramientas, producto de trabajos académicos previamente realizados. Esto facilita el avance y permite poner un mayor énfasis en el desarrollo del proyecto en sí mismo, en lugar de tener que destinar tiempo al aprendizaje de las tecnologías a utilizar. En segundo lugar, estas herramientas cuentan con un excelente soporte para Python, lenguaje de programación elegido para el desarrollo de este proyecto. Gracias a esto se cuenta con buena documentación disponible, así como soporte por parte de la comunidad para los problemas que puedan surgir. Por último, pero no menos importante, Keras y TensorFlow son tecnologías de referencia en el desarrollo de sistemas basados en *Machine Learning*. Por su naturaleza de código abierto cuenta con gran participación de colaboradores individuales, pero también de grandes empresas de la industria del software, por lo que resulta difícil encontrar herramientas superiores.

La primer capa del modelo utiliza la función `AveragePooling2D` [103] de Keras. Esta capa opera recibiendo un tensor como entrada y retornando un tensor de tamaño menor al original, haciendo una operación que podría entenderse como una suerte de promedio. Más específicamente, toma el promedio de las entradas del tensor comprendidas dentro de una ventana de tamaño $m * n$ (conocida como `pool_size`) y lo retorna como una única entrada del tensor de salida. Esta "ventana" se mueve a través de cada dimensión del tensor a intervalos controlados por una variable denominada `strides`. El objetivo de esta capa consiste en reducir el tamaño de los tensores para facilitar su procesamiento por parte del sistema, así como también promediar valores para combatir el sobreajuste que pueda generar el modelo.

La siguiente capa del modelo utiliza la función `Flatten` [105] de Keras. Su función es la de convertir vectores multidimensionales (o matrices) en unidimensionales, con el objetivo de hacer más fácil (o "barato" computacionalmente hablando) su posterior procesamiento. Así, por ejemplo, si la capa anterior retornase una matriz de 3 filas y 5 columnas, `Flatten` la convertiría en un vector unidimensional de largo 15. En el caso concreto de este

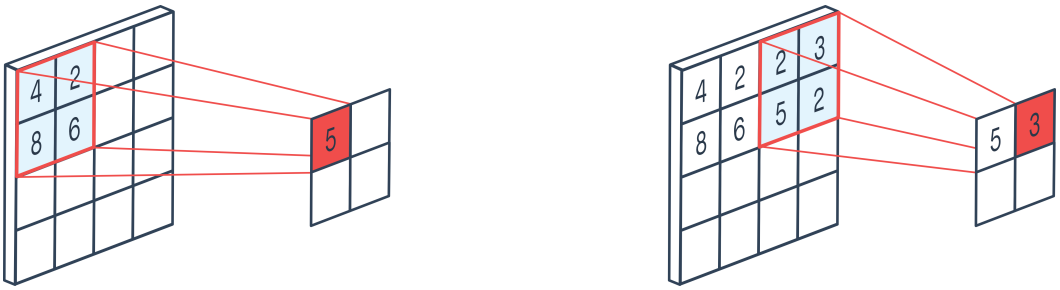


Figura 5. AveragePooling2D con `pool_size=(2,2)` y `strides=(2,2)`, ilustrado en el artículo *2D Average Pooling* [104].

proyecto, es utilizada para "aplanar" la matriz resultante de la capa `AveragePooling2D`, para ser utilizada como entrada en la capa `Dense` que ocurre a continuación.

La tercera capa del modelo utiliza la función `Dense` [106] de Keras, que implementa una Red Neuronal densamente conectada. En este caso, la salida de la Red Neuronal es un vector unidimensional de tamaño 512. Utiliza la función de activación `ReLU` [107], que puede definirse como $f(x) = max(x, 0)$, donde se devuelve la entrada si esta es positiva, pero 0 si es negativa. Cada función de activación tiene sus ventajas y desventajas, pero `ReLU` es una de las comunmente utilizadas.

A continuación se añade una capa que utiliza `Dropout` [108] de Keras, que cambia valores de la entrada por 0 con una probabilidad p que recibe como parámetro. A su vez, aquellos valores de la entrada que no son llevados a 0 son ajustados de acuerdo a la función $1/(1-p)$, de manera que la suma de todas las entradas se mantenga sin cambios. El objetivo es prevenir el sobreajuste del modelo al introducir cierta aleatoriedad durante el entrenamiento.

Como capa final hay otra instancia de `Dense`, esta vez utilizando la función de activación `softmax` [107]. La particularidad de `softmax` es que convierte los valores de un vector en una distribución de probabilidad, donde la suma de todas las entradas del vector da como resultado 1. Además, esta capa da como salida un vector de dimensión 2: una de estas 2 entradas representa el caso en que ocurre violencia, mientras que la otra representa el caso en que no ocurre violencia. Estos 2 elementos (la función `softmax` y la forma de la salida de la función) combinados entregan como resultado un vector donde sus 2 entradas expresan la probabilidad de que haya violencia o no haya violencia. Esta función es aplicada sobre los *frames* de video, por lo que se concluirá que un *frame* muestra un hecho violento cuando el valor (probabilidad) de la entrada del vector correspondiente a violencia supere a la entrada correspondiente a no violencia.

A lo largo de cada capa (empezando por la primera, que utiliza Inception) el objetivo es reducir la cantidad de datos a procesar, conservando la mayor cantidad y calidad de información posible. De esta manera se busca reducir la cantidad de potencia computacional requerida para cumplir con el objetivo deseado: etiquetar los videos como violentos o no violentos. Luego de que los datos son procesados y reducidos, las capas `Dense` implementan una Red Neuronal que, tras ajustar sus pesos en la fase de entrenamiento, será la encargada de dar un veredicto sobre cada *frame* de video, clasificándolo en violento o no violento.

4.5.2. Modelo utilizando LSTM

Durante el desarrollo del modelo presentado en la sección anterior, y a medida que se realizan pruebas sobre el mismo, surgen ideas sobre posibles mejoras. En base a lo estudiado durante la elaboración del Estado del Arte, el equipo tenía conocimiento de proyectos que se valieron de la utilización de LSTM para la resolución de problemas similares al que aquí se plantea. Es por tal motivo que se plantea la idea de incorporar esto al sistema desarrollado, para así realizar pruebas y entender si el hecho de tener la capacidad de vincular temporalmente los fotogramas de video puede dar mejores resultados que analizar cada uno individualmente.

Sin embargo, el hecho de incorporar LSTM a la solución cambia completamente la arquitectura de la misma, por lo cual no es tan sencillo como añadir una nueva capa al modelo. Es así que el trabajo requiere volver a comenzar desde cero el desarrollo del sistema, por lo que más que una evolución de lo presentado en la sección precedente, esta sección introduce un sistema totalmente nuevo. A su vez, el sistema anterior sirve como punto de referencia, dado que utiliza componentes desarrollados previamente por terceros y probados por una amplia comunidad.

Para sacar partido al potencial de las LSTM se debe cambiar el procesamiento de los videos realizado, de manera que se puedan procesar los fotogramas y entregarlos a la red en el formato neceasrio para que ésta pueda procesarlos. Luego se rediseña también toda la Red Neuronal en sí misma, utilizando no sólo la capa correspondiente a la LSTM, sino también varias más que serán explicadas a continuación, como aquellas de *pooling* y *dropout*.

Para sacar partido de la funcionalidad de LSTM, la entrada consta de una secuencia de *frames*, donde cada secuencia pertenece a un video y está conformada por 5 *frames*. Además, se extrae un *frame* por cada segundo de video para evitar caer en redundancias. Es decir, cada entrada es una secuencia de video de 5 segundos. Cada secuencia debe ser etiquetada como violenta o no violenta y las etiquetas para la etapa de entrenamiento se deducen a partir del video al cual

pertenecen dichas secuencias. O sea que, si a partir de un video violento se extraen 5 secuencias, las 5 serán etiquetadas como violentas; de igual forma sucede para los videos no violentos.

Se opta por la implementación de una red LSTM, por su capacidad para retener y asociar información a través del tiempo, aspecto fundamental para detectar patrones en un video.

Sobre la arquitectura del modelo, ésta aplica varias capas de procesamiento de imagen para extraer los *features* relevantes de cada *frame*. Una vez hecho esto, la secuencia preprocesada es enviada como parámetro de entrada a la capa de LSTM. Finalmente, sobre la salida de la red LSTM se aplica una red *fully connected*, seguido de una capa de *dropout* para evitar el sobreajuste y una capa de activación *softmax*, con una salida de dimensión 2. La primera dimensión indica la probabilidad de que la secuencia sea no violenta y la segunda dimensión la probabilidad de que la secuencia sea violenta. La Figura 6 muestra en detalle la arquitectura de la solución.

Se procede a explicar con mayor profundidad cada una de las capas. Para comenzar, a cada *frame* de la secuencia se le aplicarán una serie de transformaciones a través de las diferentes capas mencionadas. La primer capa aplica la función SeparableConv2D [109], la cual se trata de una convolución efectuada de manera más eficiente que la convolución ordinaria, utilizando menos parámetros e implicando menor cantidad de multiplicaciones. Esta se compone en dos etapas, una convolución *depthwise*, que puede ser conocida como el *filtering*, donde se aplica un filtro a cada canal de la imagen (en el caso del presente trabajo, un filtro por color) para luego aplicar una convolución *pointwise* que es una convolución de 1x1 sobre los 3 canales de color. Luego es seguido por una capa de BatchNormalization [110], con el objetivo de mantener todos los valores dentro de una misma escala y evitar problemas o inconvenientes durante el entrenamiento. Posteriormente una capa de MaxPooling2D [111] es agregada, con el fin de reducir el tamaño de la entrada nuevamente. *MaxPooling* funciona de igual manera que *AveragePooling* (explicado anteriormente) pero con la función de Máximo como selector, en lugar de la función de Promedio.

Se repite el proceso mencionado previamente agregando nuevamente una convolución (SeparableConv2D [109]) y una capa de MaxPooling [111], pero esta vez, se adiciona una capa de *Dropout* entre ambas, con valor de 0.2 para evitar el *overfitting*. Acto seguido, se aplica la capa de Flatten [105] (explicado anteriormente) para entregarle a la capa de LSTM una *input* con una entrada de 1 dimension. Posteriormente, existen dos capas de LSTM bidireccionales [112] de 128 y 64 unidades respectivamente y en secuencia, es decir, el *output* de una red hace de *input* de la siguiente. Una LSTM de tipo bidireccional permite entrenar

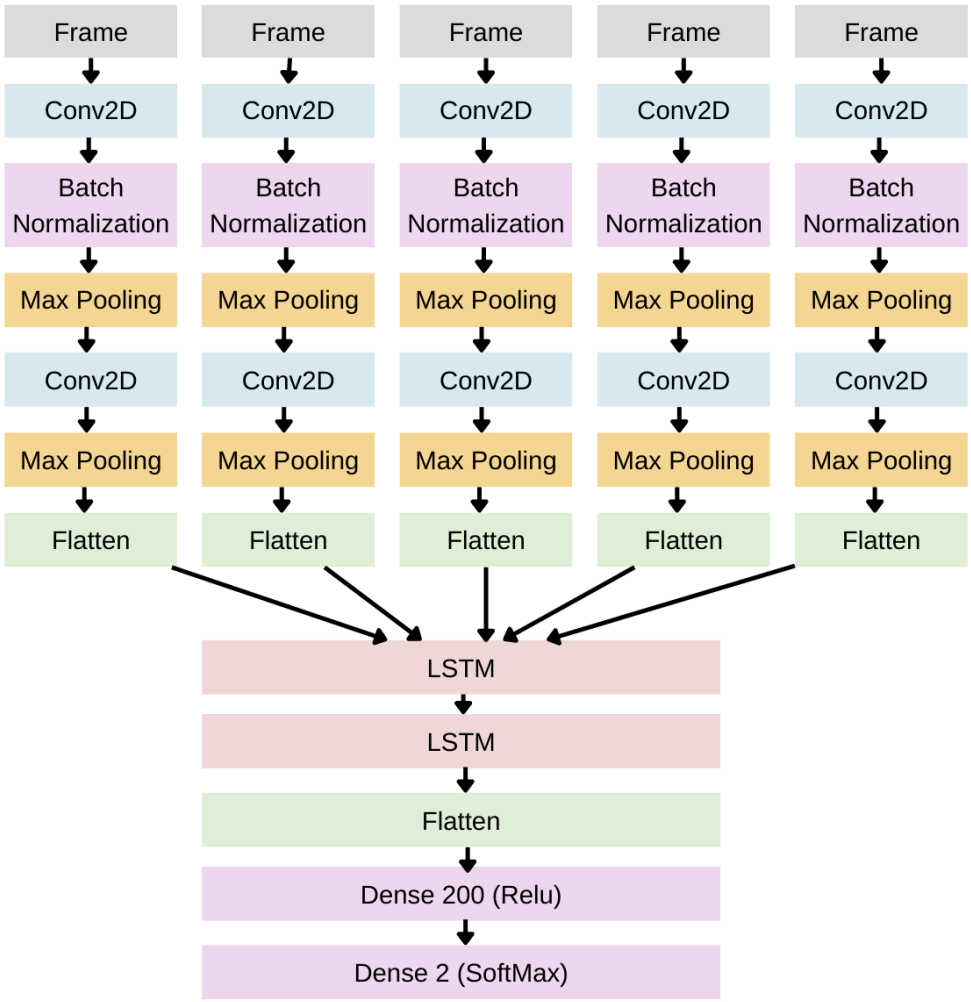


Figura 6. Arquitectura de la red LSTM en detalle

una LSTM tanto sobre el *input* tal y como está, así como sobre una copia invertida del *input*, permitiendo agregar contexto extra a la red y acelerar el aprendizaje. Por último, la salida de las LSTM es procesada por una capa de *flatten*, para después atravesar una red *"fully connected"* de 200 neurones con activación *Rectified Linear Units* (ReLU), seguido de una nueva capa de *Dropout* con el fin de evitar el sobreajuste.

Finalmente, se cuenta con una capa *"fully connected"* con una salida de tamaño 2 y con una función de activación *SoftMax*. Aquí cada dimensión del vector de salida está mapeado a una etiqueta (No Violencia o Violencia) y asociado a una

probabilidad. Siendo la dimensión de mayor probabilidad, la que indica el valor a etiquetar.

5. Experimentación

Los modelos desarrollados fueron puestos a prueba en diversas ocasiones y con distintas configuraciones, con el fin de evaluar su funcionamiento y realizar los ajustes pertinentes en busca de una mejora de performance. Tanto para la fase de entrenamiento como para la de evaluación de resultados se utiliza la plataforma Google Cloud, en la cual se contrata una instancia *e2-highmem-8 (Efficient Instance, 8 vCPUs, 64 GB RAM)*, sin GPU y con un disco de estado sólido de 100 GB. Se utiliza la versión 2.1 de TensorFlow [113]. Allí también es almacenado el *dataset* utilizado, aprovechando los servicios de almacenamiento de la plataforma. De esta manera todo el procesamiento de los videos previo a servir de entrada para el modelo fue también realizado en la instancia contratada. El modelo resultante de la fase de entrenamiento es guardado y utilizado para evaluar los resultados en esta misma instancia.

Son probadas distintas configuraciones de los modelos aquí presentados, variando no solo los parámetros de éstos, sino también la composición del *dataset* utilizado. Las primeras evaluaciones son hechas sobre pequeñas porciones del *dataset*, buscando optimizar los tiempos de entrenamiento para poder realizar más pruebas en el tiempo disponible. Aquellas configuraciones que muestran un mayor potencial son luego probadas con porciones mayores del set de datos, a fin de evaluar si los resultados muestran mejoras al contar con mayor información durante la fase de entrenamiento. Además, se busca optimizar los videos de situaciones violentas utilizados para el entrenamiento: el equipo ha notado que en muchas ocasiones éstos contienen un pequeño hecho de violencia de pocos segundos de duración, acompañados de decenas de segundos de acciones cotidianas y no violentas. Sin embargo, todos los *frames* de ese video eran etiquetados como violentos en el *dataset*, lo que podría inducir a error al modelo producto de la alta cantidad de cuadros no violentos etiquetados como violentos. Por tal motivo, los videos son editados para dejar únicamente aquellos segmentos en los que realmente ocurre violencia, a los efectos de mejorar los resultados obtenidos. Las pruebas experimentales demuestran que este enfoque fue acertado, mejorando ligeramente las distintas métricas con las que es evaluado el modelo.

5.1. Información necesaria para la comprensión de los resultados

Para comprender completamente los resultados que son presentados a continuación, es necesario conocer previamente algunos conceptos. En particular, interesa definir las métricas utilizadas para la evaluación de la performance de los modelos.

- *Precision*: se define como la proporción de resultados relevantes de entre el total de resultados obtenidos. En el caso particular de este proyecto, donde interesa clasificar imágenes de video entre hechos violentos y no violentos, la *precision* para videos violentos dará como resultado qué porcentaje de los *frames* de videos marcados como violentos son efectivamente violentos. Lo mismo ocurrirá para el caso de los videos no violentos. [114]
- *Recall*: refiere a la proporción de resultados relevantes que fueron obtenidos; es decir, indica qué porción de los resultados que deberían haber sido etiquetados dentro de una categoría realmente lo fueron. Para el caso de este proyecto, el *recall* correspondiente a hechos violentos (no violentos) dará como resultado el porcentaje de cuadros de video que fueron marcados como violentos (no violentos) del total de cuadros que efectivamente son violentos (no violentos). [115]
- *F1-score*: es definido como la media armónica de *precision* y *recall*, que busca resumir en una sola métrica la información que proveen las anteriores dos. [116]

Por otra parte, las matrices de datos que son presentadas incluyen una columna nombrada como *support*, que da cuenta de la cantidad de *frames* de video procesados para la categoría perteneciente a la fila en concreto. Además, se incluye información de los promedios por columna, así como también de promedios ponderados de acuerdo a la mencionada columna *support*. Y por último, la fila correspondiente a *accuracy* indica con que frecuencia la categoría que el modelo predice coincide con la categoría etiquetada en el *dataset*. [117]

5.2. Modelo basado en Inception

A continuación se encuentran los resultados finales obtenidos utilizando el primero de los modelos desarrollados, basado en Inception V3.

	precision	recall	f1-score	support
NonViolence	1.00	1.00	1.00	5155
Violence	1.00	0.99	1.00	2698
accuracy			1.00	7853
macro avg	1.00	1.00	1.00	7853
weighted avg	1.00	1.00	1.00	7853

Figura 7. Resultados de entrenamiento utilizando Inception V3.

La Figura 7 muestra los resultados obtenidos luego de la fase de

entrenamiento, que presentan un escenario en el cual el modelo es capaz de acertar prácticamente cada una de sus predicciones, lo cual parece poco real. Una situación como esta suele indicar un sobreajuste del modelo sobre el set de datos utilizado para entrenar. De ser este el caso, cuando el modelo sea ejecutado tomando como entrada un *dataset* diferente y más complejo que el que ha utilizado para entrenar, los resultados deberían mostrar una clara desmejora.

	precision	recall	f1-score	support
NonViolence	0.94	0.79	0.86	12764
Violence	0.54	0.82	0.65	3771
accuracy			0.80	16535
macro avg	0.74	0.81	0.75	16535
weighted avg	0.85	0.80	0.81	16535

Figura 8. Resultados de test utilizando Inception V3.

En la Figura 8 se encuentran las métricas referentes a los resultados obtenidos con el conjunto de test. En comparación a los datos analizados previamente, puede apreciarse una notoria pérdida de performance en la predicción de videos de acciones violentas. El *recall* obtenido en situaciones de violencia indica que el modelo es capaz de reconocer como violentos al 82 % de los *frames* de videos en los que efectivamente ocurren hechos de violencia. Sin embargo, la *precision* para este caso marca que el modelo sólo acierta en un 54 % de las ocasiones sus predicciones de *frames* de video violentos. Esto indica que existen muchos cuadros de video no violentos que están siendo etiquetados como violentos. Mirando los datos de los hechos no violentos se confirma la tendencia: hay un 94 % de acierto al etiquetar *frames* no violentos, pero únicamente el 79 % de aquellos que no son violentos son reconocidos como tales. El modelo está fallando al etiquetar muchos cuadros de video donde no ocurre violencia como violentos, provocando falsos positivos.

A los efectos del objetivo buscado en este proyecto, el problema podría no ser tan grande como a priori podría parecer: un sistema de detección de hechos violentos probablemente contará con supervisión humana, por lo que es preferible contar con falsos positivos que puedan ser descartados por los supervisores antes que obviar hechos violentos y no notificarlos (caso de falsos negativos). Sin embargo, tener un número muy elevado de falsos positivos puede causar demasiado “ruido” en el sistema, sobrecargando a los operadores del mismo y reduciendo su efectividad, por lo cual interesa buscar mejoras de rendimiento.

La evidencia prueba que el modelo tiene un sobreajuste sobre los datos de

entrenamiento, lo que provoca una merma de resultados cuando los datos utilizados como entrada difieren de aquellos usados para entrenar. Una causa de este problema podría estar en los hiper-parámetros configurados sobre el sistema, que en caso de ser demasiado agresivos podrían inducir al sobreajuste. Sin embargo, el *learning rate* final utilizado es de apenas 0.0001, cuando el valor por defecto es de 0.01, lo cual quiere decir que está cien veces por debajo. El resto de hiper-parámetros también fue probado en varias configuraciones, siendo los valores finales: *momentum* = 0.9, *decay* = 0.0001.

No fue posible evitar el sobreajuste utilizando configuraciones más conservadoras de hiper-parámetros, por lo que el equipo pasa al siguiente sospechoso: el *dataset*. Un set de datos demasiado pequeño puede provocar que el modelo converja rápidamente a una solución sobreajustada al set de entrenamiento, mientras que los resultados sobre un conjunto de test independiente sean malos. Sin embargo, la combinación de datos y *hardware* disponibles por parte del equipo hace imposible mejorar los valores para este modelo: los tiempos de entrenamiento se tornan sumamente extensos (días de entrenamiento por cada intento) y la cantidad de datos sigue siendo insuficiente como para obtener mejoras significativas.

Esta situación provoca la necesidad de experimentar con otro tipo de soluciones que permitan buscar una mejora de resultados, soluciones que se enfocasen más en la estructura del modelo en sí misma, en lugar de los datos de entrada o los hiper-parámetros.

5.3. Modelo basado en LSTM

El modelo basado en LSTM produce avances en cuanto a los problemas vistos al utilizar aquel basado en Inception. Como muestra la Figura 9, el modelo basado en LSTM también sobreajusta para el *dataset* de entrenamiento.

	precision	recall	f1-score	support
NonViolence	0.98	1.00	0.99	157
Violence	1.00	0.97	0.99	120
accuracy			0.99	277
macro avg	0.99	0.99	0.99	277
weighted avg	0.99	0.99	0.99	277

Figura 9. Resultados de entrenamiento utilizando LSTM.

Sin embargo, y como muestra la Figura 10, al ejecutar el modelo sobre los datos

de test, los resultados son superiores: la *precision* es más estable entre los casos de videos violentos y no violentos, mientras que el *recall* mejora en promedio con respecto al modelo basado en Inception.

	precision	recall	f1-score	support
NonViolence	0.87	0.93	0.90	458
Violence	0.87	0.76	0.81	266
accuracy			0.87	724
macro avg	0.87	0.85	0.86	724
weighted avg	0.87	0.87	0.87	724

Figura 10. Resultados de test utilizando LSTM.

Nuevamente el equipo se ve ante un problema derivado de los limitados recursos disponibles: pese a utilizar configuraciones conservadoras para intentar frenar el sobreajuste durante el entrenamiento, la escasez de datos y capacidad de cómputo disponible no permiten terminar de solucionar este problema. A pesar de esto, los resultados sobre el conjunto de test (independiente del de entrenamiento) se presentan más balanceados y con mayor precisión que en el caso anterior.

Cabe destacar que la cantidad de cuadros de video (determinados por la columna *support*) son notablemente menores en este modelo basado en LSTM, debido a que en este caso se combinan varios de éstos para extraer *features* comunes vinculadas temporalmente, como fue explicado en secciones precedentes. Estos resultados son evaluados como sumamente positivos por el equipo, dado que se trata de un modelo desarrollado enteramente para este proyecto, a diferencia del anterior modelo de presentado, que basaba gran parte de su potencial en la utilización de Inception como base para su desarrollo.

El equipo cree que el hecho de utilizar LSTM ha contribuido a la mejora de los resultados. Esto se debe a que las LSTM permiten lograr una vinculación temporal entre los cuadros de video, lo cual permite atenuar variaciones entre la detección o no de violencia de un cuadro al siguiente. Por ejemplo, es más difícil que un cuadro de video se catalogue como no violento si se encuentra entre otros 2 cuadros que son catalogados como violentos, dado que el cuadro del medio verá afectado su etiquetado por su predecesor.

6. Conclusiones y trabajo futuro

6.1. Conclusiones finales

Tras culminar el trabajo de investigación y desarrollo, habiendo creado dos modelos de detección de crímenes y experimentado con los mismos, el equipo obtiene algunos resultados interesantes. Los objetivos planteados al inicio, como se describe en secciones anteriores, consisten en desarrollar un sistema informático capaz de reconocer crímenes. El mismo busca funcionar con mayor eficacia en secuencias captadas por cámaras de seguridad ubicadas en la vía pública, y por eso el sistema fue optimizado para funcionar sobre los datos que se obtienen del *hardware* actualmente instalado.
Se puede ver esto como tres grandes tareas:

- Conseguir un *dataset* de actos delictivos en la vía pública para el desarrollo de la solución.
- Desarrollar un programa capaz de procesar los datos de entrada en forma de video y transformarlos en datos útiles para el sistema de detección de crímenes.
- Desarrollar un sistema informático para que reconozca estos actos criminales, minimizando el margen de error.

Se pudieron cumplir con éxito estas 3 tareas, ya que se pudo desarrollar un modelo que es capaz de tomar una secuencia de video y detectar si en ella ocurren actos criminales, comunicando a cada momento si lo observado es violento o no (con una etiqueta de valor booleano a la esquina del video que cambia a medida que pasa el tiempo), partiendo únicamente de videos preexistentes y etiquetados que le fueron dados a modo de *input* para entrenamiento. Así se demuestra que es posible, con el *hardware* adecuado, alimentar el modelo con transmisiones de video en tiempo real (o lo más cercano posible a esto) para formar parte de un sistema más complejo de alertas ante hechos violentos (que además cuenta con buenos valores en las métricas utilizadas en estas pruebas). Esto puede suponer un soporte fundamental para las autoridades públicas al momento de enfrentarse a este tipo de situaciones.

Por otro lado, queda en evidencia según lo visto en el Estado del Arte que existen otras técnicas complementarias a la ensayada en este proyecto. Si bien aquí se optó por poner a prueba aquella que se cree más efectiva, el complemento de distintos recursos aumentará sin dudas la eficacia en la tarea de combatir la criminalidad. Sin embargo, de las investigaciones llevadas a cabo en el proyecto han surgido serias dudas en cuanto a utilizar aquellas técnicas basadas en reconocimiento de emociones a partir de lenguaje corporal/facial. Ha quedado de manifiesto que este tipo de teorías tienen aún demasiados detractores en el ámbito

académico y científico, por lo que se entiende que es necesario que la ciencia laude este tema antes de aplicarlo en la vida cotidiana, dada la influencia que puede tener en toma de decisiones importantes.

Además, está el hecho del trabajo llevado a cabo por el equipo para desarrollar y evaluar 2 soluciones diferentes. La primera de ellas contaba con componentes basados en soluciones de terceros que permitían combinar soluciones ya probadas. Sin embargo, la segunda requirió bastante más investigación, dado que fue desarrollada de una manera mucho más "artesanal", utilizando los conocimientos adquiridos durante este trabajo de investigación. Los resultados fueron positivos para ambos casos, aunque la segunda solución demostró leves mejoras. Pero más allá de que los números muestren mejoras, esta solución se entiende por parte del equipo como una mejora a nivel conceptual. Dado que un video es una secuencia de fotogramas que tienen relación entre sí, es decir, que existen dependencias entre fotogramas a nivel semántico, parece correcto tener un sistema que busque estas relaciones. El primer modelo (basado en Inception) consideraba cada cuadro como una entidad completamente aislada, con lo cual se pierde información del contexto en que ocurre cada imagen.

Este trabajo dedicado a encontrar una mejor solución al problema también se extendió hacia el *dataset*, que como fue explicado en secciones precedentes, es fundamental para el entrenamiento de los modelos. No fue suficiente el hecho de contar con un juego de datos adecuado, sino que se trabajó en ajustarlo a las necesidades propias, buscando reflejar de la mejor manera posible los acontecimientos violentos o no violentos. Con esto se ha pretendido aportar a la comunidad no solo algunas ideas a nivel técnico, sino también desde el punto de vista de la calidad de datos para este tipo de sistemas.

6.2. Trabajo futuro

Se identifican varias líneas de trabajo futuro que podría ser interesante explorar, las cuales se detallan a continuación.

Para comenzar, en el armado de este modelo de reconocimiento de crimen, el *hardware* utilizado no fue el mejor. El servicio de Google Cloud utilizado no cuenta con tarjeta gráfica para acelerar el entrenamiento y testeo del mismo ni los modelos de procesadores óptimos para esto (ej: Intel Xeon, AMD Threadripper, etc.). El hecho de poder contar con un hardware más capaz daría la posibilidad de entrenar un modelo con una cantidad de datos mayor, favoreciendo los resultados finales. Dado que el modelo ha mostrado resultados promisorios en un hardware limitado y con un conjunto de datos acotado, expandir la investigación sería deseable.

La posibilidad de contar un *hardware* más potente dotaría al sistema de la

posibilidad de ser ejecutado sobre una transmisión de video en tiempo real. Esto es algo importante, ya que durante las pruebas realizadas no se contó con capacidad suficiente para lograr esto, teniendo el equipo que limitarse a procesar videos almacenados. Poder realizar esto y trabajar sobre las optimizaciones necesarias para lograr funcionar sobre videos en vivo es una de las líneas de trabajo más interesantes a futuro.

Otra gran limitante en la experimentación de este proyecto está en el dataset utilizado para el entrenamiento. Es muy difícil conseguir videos con crímenes tan explícitos y definidos, con poco ruido, que sean en la vía pública, con buena resolución y que además tengan momentos con visibles diferencias con un video ordinario del día a día. Como mejora se plantea entonces conseguir estos datos de algún ente, ya sea privado o estatal, como sería el caso del Ministerio del Interior. El equipo estuvo en tratativas con integrantes del Ministerio, pero lamentablemente no fue posible obtener resultados satisfactorios en el plazo de desarrollo de este proyecto. Sin embargo, con un plazo mayor de tiempo esta es una línea de investigación que puede ayudar en la mejora del modelo desarrollado. De tener dicho *dataset*, el modelo tendría videos bastante más similares entre sí (ya que sería todo la misma ciudad), cosa que además podría ayudarlo a adaptarse mejor a los rasgos criminales propios de una cierta región o ciudad, como puede ser Montevideo en este caso. Estudiar el balance entre un modelo más generalizable y uno más adaptado a entornos concretos sería útil para buscar posibles optimizaciones.

Si bien el modelo ha logrado "aprender" a reconocer situaciones violentas a partir del entrenamiento con un dataset determinado, luego de lo visto en el Estado del Arte, surge la idea de dotarlo de la capacidad de reconocer también otras señales que lo ayuden a la hora de determinar si se está por cometer un crimen o no. El primer ejemplo que viene a la mente es el del reconocimiento de armas, que podría ser muy útil para la predicción y detección de los mismos. Sin embargo, pueden surgir otras opciones que complementen este enfoque, dando lugar a distintas opciones a la hora de emitir una alerta en el sistema.

7. Bibliografía

Referencias

[1] Stephen J Fay. "Tough on crime, tough on civil liberties: some negative aspects of Britain's wholesale adoption of CCTV surveillance during the 1990s". En: *International Review of Law, Computers & Technology* 12.2 (1998), págs. 315-347.

[2] Brandon C Welsh y David P Farrington. "Public area CCTV and crime prevention: an updated systematic review and meta-analysis". En: *Justice Quarterly* 26.4 (2009), págs. 716-745.

[3] Clive Norris, Mike McCahill y David Wood. "The growth of CCTV: a global perspective on the international diffusion of video surveillance in publicly accessible space". En: *Surveillance & Society* 2.2/3 (2004).

[4] L. Greenfeld y . Robert A. Nisbet. "social science, Encyclopedia Britannica". En: (2021). URL: https://www.britannica.com/topic/social-science.

[5] Pamela McCorduck. *Machines who think : a personal inquiry into the history and prospects of artificial intelligence / Pamela McCorduck.* eng. San Francisco: W.H. Freeman, 1979. ISBN: 0716710722.

[6] Daniel Crevier. *AI: The Tumultuous History of the Search for Artificial Intelligence*. Ene. de 1993, págs. 1-386.

[7] Christian Szegedy y col. "Going deeper with convolutions". En: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, págs. 1-9. DOI: 10.1109/CVPR.2015.7298594.

[8] Sepp Hochreiter y Jürgen Schmidhuber. "Long Short-Term Memory". En: *Neural Computation* 9.8 (nov. de 1997), págs. 1735-1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[9] Enrique Bermejo Nievas y col. "Violence Detection in Video Using Computer Vision Techniques". En: (2011). Ed. por Pedro Real y col., págs. 332-339.

[10] Laptev y Lindeberg. "Space-time interest points". En: *Proceedings Ninth IEEE International Conference on Computer Vision*. 2003, 432-439 vol.1. DOI: 10.1109/ICCV.2003.1238378.

[11] Ming-yu Chen y Alexander Hauptmann. “MoSIFT: Recognizing Human Actions in Surveillance Videos”. En: *CMU-CS-09-161* (ene. de 2009).

[12] Aristidis Likas, Nikos Vlassis y Jakob J. Verbeek. “The global k-means clustering algorithm”. En: *Pattern Recognition* 36.2 (2003). Biometrics, págs. 451-461. ISSN: 0031-3203. DOI: https://doi.org/10.1016/S0031-3203(02)00060-2. URL: https://www.sciencedirect.com/science/article/pii/S0031320302000602.

[13] Corinna Cortes y Vladimir Vapnik. “Support-vector networks”. En: *Chem. Biol. Drug Des.* 297 (ene. de 2009), págs. 273-297. DOI: 10.1007/%2FBF00994018.

[14] Annalisa Barla, Francesca Odone y Alessandro Verri. “Histogram intersection kernel for image classification”. En: vol. 3. Oct. de 2003, págs. III-513. ISBN: 0-7803-7750-8. DOI: 10.1109/ICIP.2003.1247294.

[15] Jean Phelipe de Oliveira Lima y Carlos Maurício Seródio Figueiredo. “A Temporal Fusion Approach for Video Classification with Convolutional and LSTM Neural Networks Applied to Violence Detection”. En: *Inteligencia Artificial* 24.67 (abr. de 2021), págs. 40-50. DOI: 10.4114/intartif.vol24iss67pp40-50. URL: https://journal.iberamia.org/index.php/intartif/article/view/573.

[16] Jiuxiang Gu y col. “Recent advances in convolutional neural networks”. En: *Pattern Recognition* 77 (2018), págs. 354-377. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2017.10.013. URL: https://www.sciencedirect.com/science/article/pii/S0031320317304120.

[17] Jürgen Schmidhuber. “Deep learning in neural networks: An overview”. En: *Neural Networks* 61 (2015), págs. 85-117. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2014.09.003. URL: https://www.sciencedirect.com/science/article/pii/S0893608014002135.

[18] Andrew G. Howard y col. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”. En: *arXiv preprint arXiv:1704.04861* (2017).

[19] Karen Simonyan y Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. En: *arXiv 1409.1556* (sep. de 2014).

[20] Mohamed Mostafa Soliman y col. “Violence Recognition from Videos using Deep Learning Techniques”. En: *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*. 2019, págs. 80-85. DOI: 10.1109/ICICIS46948.2019.9014714.

[21] Bh. SravyaPranati y col. “Large-Scale Video Classification with Convolutional Neural Networks”. En: *Information and Communication Technology for Intelligent Systems*. Ed. por Tomonobu Senjyu y col. Singapore: Springer Singapore, 2021, págs. 689-695. ISBN: 978-981-15-7062-9.

[22] Tom Fawcett. “An introduction to ROC analysis”. En: *Pattern Recognition Letters* 27.8 (2006). ROC Analysis in Pattern Recognition, págs. 861-874. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2005.10.010. URL: https://www.sciencedirect.com/science/article/pii/S016786550500303X.

[23] Tao Zhang y col. “A new method for violence detection in surveillance scenes”. En: *Multimedia Tools and Applications* 75 (mayo de 2015). DOI: 10.1007/s11042-015-2648-8.

[24] Douglas Reynolds. “Gaussian Mixture Models”. En: *Encyclopedia of Biometrics*. Ed. por Stan Z. Li y Anil Jain. Boston, MA: Springer US, 2009, págs. 659-663. ISBN: 978-0-387-73003-5. DOI: 10.1007/978-0-387-73003-5_196. URL: https://doi.org/10.1007/978-0-387-73003-5_196.

[25] D. Fleet e Y. Weiss. “Optical Flow Estimation”. En: *Handbook of Mathematical Models in Computer Vision*. Ed. por Nikos Paragios, Yunmei Chen y Olivier Faugeras. Boston, MA: Springer US, 2006, págs. 237-257. ISBN: 978-0-387-28831-4. DOI: 10.1007/0-387-28831-7_15. URL: https://doi.org/10.1007/0-387-28831-7_15.

[26] S. Blunsden y Robert Fisher. “The BEHAVE video dataset: ground truthed video for multi-person”. En: *Ann. BMVA* 4 (abr. de 2009).

[27] Tal Hassner, Yossi Itcher y Orit Kliper-Gross. *Crowd Violence Dataset*. URL: https://www.openu.ac.il/home/hassner/data/violentflows/ (visitado 01-12-2021).

[28] Francisco Melo. “Area under the ROC Curve”. En: *Encyclopedia of Systems Biology*. Ed. por Werner Dubitzky y col. New York, NY: Springer New York, 2013, págs. 38-39. ISBN: 978-1-4419-9863-7. DOI: 10.1007/978-1-4419-9863-7_209. URL: https://doi.org/10.1007/978-1-4419-9863-7_209.

[29] N. Dalal y B. Triggs. “Histograms of oriented gradients for human detection”. En: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. 2005, 886-893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[30] Janez Pers y col. “Histograms of optical flow for efficient representation of body motion”. En: *Pattern Recognition Letters* 31 (ago. de 2010), págs. 1369-1376. DOI: 10.1016/j.patrec.2010.03.024.

[31] Fillipe de Souza y col. “Violence Detection in Video Using Spatio-Temporal Features”. En: ago. de 2010, págs. 224-230. DOI: 10.1109/SIBGRAPI.2010.38.

[32] Tal Hassner, Yossi Itcher y Orit Kliper-Gross. “Violent flows: Real-time detection of violent crowd behavior”. En: *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. 2012, págs. 1-6. DOI: 10.1109/CVPRW.2012.6239348.

[33] *CAVIAR Dataset*. https://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/. Accedido: 14 de Noviembre, año 2021.

[34] Jeff Donahue y col. “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description”. En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4 (2017), págs. 677-691. DOI: 10.1109/TPAMI.2016.2599174.

[35] *Caffe Deep Learning Framework*. https://caffe.berkeleyvision.org/. Accedido: 14 de Noviembre, año 2021.

[36] Iqbal Sarker. “Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions”. En: *SN Computer Science* 2 (nov. de 2021). DOI: 10.1007/s42979-021-00815-1.

[37] Khurram Soomro, Amir Zamir y Mubarak Shah. “UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild”. En: *CoRR* (dic. de 2012).

[38] Dylan Josh Coming Lopez y Cheng-Chang Lien. “Real-Time Human Violent Activity Recognition Using Complex Action Decomposition”. En: (2020), págs. 360-364. DOI: 10.1109/ICS51289.2020.00078.

[39] Karen Simonyan y Andrew Zisserman. *Two-Stream Convolutional Networks for Action Recognition in Videos*. 2014. arXiv: 1406.2199 [cs.CV].

[40] Yiğithan Dedeoğlu y col. "Silhouette-Based Method for Object Classification and Human Action Recognition in Video". En: vol. 3979. Mayo de 2006, págs. 64-77. ISBN: 978-3-540-34202-1. DOI: 10.1007/11754336_7.

[41] Joseph Redmon y Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: 1804.02767 [cs.CV].

[42] Till Kroeger y col. "Fast Optical Flow using Dense Inverse Search". En: *CoRR* abs/1603.03590 (2016). arXiv: 1603.03590. URL: http://arxiv.org/abs/1603.03590.

[43] Tak-Wai Hui, Xiaoou Tang y Chen Change Loy. *LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation*. 2018. arXiv: 1805.07036 [cs.CV].

[44] Shen Jianjie y Zou Weijun. "Violence Detection Based on Three-Dimensional Convolutional Neural Network with Inception-ResNet". En: (2020), págs. 145-150. DOI: 10.1109/TOCS50858.2020.9339755.

[45] Sanam Narejo y col. "Weapon Detection Using YOLO V3 for Smart Surveillance System". Inglés. En: *Mathematical Problems in Engineering* 2021 (2021). Publisher Copyright: © 2021 Sanam Narejo et al. ISSN: 1024-123X. DOI: 10.1155/2021/9975700.

[46] Caifeng Shan, Shaogang Gong y Peter Mcowan. "Facial expression recognition based on Local Binary Patterns: A comprehensive study". En: *Image and Vision Computing* 27 (mayo de 2009), págs. 803-816. DOI: 10.1016/j.imavis.2008.08.005.

[47] Abdenour Hadid. "The Local Binary Pattern Approach and its Applications to Face Analysis". En: *2008 First Workshops on Image Processing Theory, Tools and Applications*. 2008, págs. 1-9. DOI: 10.1109/IPTA.2008.4743795.

[48] Tu Chengsheng, Liu Huacheng y Xu Bing. "AdaBoost typical Algorithm and its application research". En: *MATEC Web of Conferences* 139 (ene. de 2017), pág. 00222. DOI: 10.1051/matecconf/201713900222.

[49] Wen-Sheng Chu, Fernando De la Torre y Jeffrey F. Cohn. "Selective Transfer Machine for Personalized Facial Action Unit Detection". En:

2013 IEEE Conference on Computer Vision and Pattern Recognition (2013), págs. 3515-3522.

[50] *What is OVerfitting? By IBM*. https://www.ibm.com/topics/overfitting. Accedido: 15 de Diciembre, año 2021.

[51] Tasriva Sikandar, Kamarul Hawari bin Ghazali y Mohammad Fazle Alam Rabbi. "ATM crime detection using image processing integrated video surveillance: a systematic review". En: *Multimedia Systems* 25 (2018), págs. 229-251.

[52] A M Husein y col. "Motion detect application with frame difference method on a surveillance camera". En: *Journal of Physics: Conference Series* 1230.1 (jul. de 2019), pág. 012017. DOI: 10.1088/1742-6596/1230/1/012017. URL: https://doi.org/10.1088/1742-6596/1230/1/012017.

[53] Nushaine Ferdinand. *Using Hourglass Networks To Understand Human Poses*. URL: https://towardsdatascience.com/using-hourglass-networks-to-understand-human-poses-1e40e349fa15 (visitado 02-12-2021).

[54] Mohammed Nurudeen y col. "Crime prediction and mapping based on real time video analysis". En: *Journal of Ambient Intelligence and Smart Environments* 10 (mar. de 2018), págs. 221-239. DOI: 10.3233/AIS-180476.

[55] Jyh-Shing Jang. "ANFIS Adaptive-Network-based Fuzzy Inference System". En: *Systems, Man and Cybernetics, IEEE Transactions on* 23 (jun. de 1993), págs. 665-685. DOI: 10.1109/21.256541.

[56] Bowen Zhang y col. "Mic-TJU at MediaEval Violent Scenes Detection (VSD) 2014". En: vol. 1263. Oct. de 2014.

[57] *Hollywod Human Action Dataset*. https://www.di.ens.fr/~laptev/actions/. (Visitado 04-12-2021).

[58] H. Kuehne y col. "HMDB: a large video database for human motion recognition". En: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2011.

[59] Amir Roshan Zamir, Afshin Dehghan y Mubarak Shah. "GMCP-Tracker: Global Multi-object Tracking Using Generalized Minimum Clique Graphs". En: *Computer Vision – ECCV 2012*. Ed. por Andrew Fitzgibbon

y col. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, págs. 343-356. ISBN: 978-3-642-33709-3.

[60] Padraig Cunningham y Sarah Delany. "k-Nearest neighbour classifiers". En: *Mult Classif Syst* 54 (abr. de 2007). DOI: 10.1145/3459665.

[61] Stanislaw Weglarczyk. "Kernel density estimation and its application". En: *ITM Web of Conferences* 23 (ene. de 2018), pág. 00037. DOI: 10.1051/itmconf/20182300037.

[62] Charuni Rajapakshe y col. "Using CNNs RNNs and Machine Learning Algorithms for Real-time Crime Prediction". En: *2019 International Conference on Advancements in Computing (ICAC)*. 2019, págs. 310-316. DOI: 10.1109/ICAC49085.2019.9103425.

[63] Tom M. Mitchell. *Machine Learning*. New York: McGraw-Hill, 1997. ISBN: 978-0-07-042807-2.

[64] Kaiming He y col. "Deep Residual Learning for Image Recognition". En: jun. de 2016, págs. 770-778. DOI: 10.1109/CVPR.2016.90.

[65] Xindong Wu y col. "Top 10 algorithms in data mining". En: *Knowledge and Information Systems* 14 (dic. de 2007). DOI: 10.1007/s10115-007-0114-2.

[66] Tin Kam Ho. "Random decision forests". En: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. 1995, 278-282 vol.1. DOI: 10.1109/ICDAR.1995.598994.

[67] Katerina Fragkiadaki y col. "Recurrent Network Models for Human Dynamics". En: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, págs. 4346-4354. DOI: 10.1109/ICCV.2015.494.

[68] Bharath Ramsundar y Reza Bosagh Zadeh. *TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning*. 1st. O'Reilly Media, Inc., 2018. Cap. 4. ISBN: 1491980451.

[69] Catalin Ionescu y col. "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (jul. de 2014), págs. 1325-1339.

[70] Disa A. Sauter y Agneta H. Fischer. "Can perceivers recognise emotions from spontaneous expressions?" En: *Cognition and Emotion* 32.3 (2018). PMID: 28447544, págs. 504-515. DOI: 10.1080/02699931.2017.1320978. eprint:

https://doi.org/10.1080/02699931.2017.1320978. URL: https://doi.org/10.1080/02699931.2017.1320978.

[71] Leon O. H. Kroczek y col. "Angry facial expressions bias towards aversive actions". En: *PLoS ONE* 16.9 (sep. de 2021), e0256912. DOI: 10.1371/journal.pone.0256912.

[72] Jennifer Bisson. "It's Written All Over Their Faces: Preschoolers' Emotion Understanding". En: *Social Development* 28 (jul. de 2018). DOI: 10.1111/sode.12322.

[73] Aya Hassouneh, A.M. Mutawa y M. Murugappan. "Development of a Real-Time Emotion Recognition System Using Facial Expressions and EEG based on machine learning and deep neural network methods". En: *Informatics in Medicine Unlocked* 20 (2020), pág. 100372. ISSN: 2352-9148. DOI: https://doi.org/10.1016/j.imu.2020.100372. URL: https://www.sciencedirect.com/science/article/pii/S235291482030201X.

[74] Devashi Choudhary y Jainendra Shukla. "Feature Extraction and Feature Selection for Emotion Recognition using Facial Expression". En: *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)*. 2020, págs. 125-133. DOI: 10.1109/BigMM50055.2020.00027.

[75] P. Ekman. *Telling Lies: Clues to Deceit in the Marketplace, Politics, and Marriage*. W.W. Norton, 2001. ISBN: 9780393321883. URL: https://books.google.com.pe/books?id=7I%5C_wDDfrwCgC.

[76] *What are emotions?* https://www.paulekman.com/universal-emotions/. (Visitado 28-12-2021).

[77] Bob Frost. "The Art of Reading Faces and How It's Being Used to Fight Terrorism." En: *Biography* 7.5 (2003), pág. 72. ISSN: 10927891. URL: http://proxy.timbo.org.uy/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=9480967&lang=es&site=eds-live.

[78] Yifei Guo y col. "A Magnitude and Angle Combined Optical Flow Feature for Microexpression Spotting". En: *IEEE MultiMedia* 28.2 (2021), págs. 29-39. DOI: 10.1109/MMUL.2021.3058017.

[79] Lu Shang y col. "Application of microexpressions analysis based on deep learning in the service industry". En: *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*. 2020, págs. 109-112. DOI: 10.1109/ITCA52113.2020.00030.

[80] Monu Verma, Santosh Kumar Vipparthi y Girdhari Singh. "AffectiveNet: Affective-Motion Feature Learning for Microexpression Recognition". En: *IEEE MultiMedia* 28.1 (2021), págs. 17-27. DOI: 10.1109/MMUL.2020.3021659.

[81] Xunbing Shen y col. "Catching a Liar Through Facial Expression of Fear". En: *Frontiers in Psychology* 12 (2021), pág. 2211. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2021.675097. URL: https://www.frontiersin.org/article/10.3389/fpsyg.2021.675097.

[82] *The Moment of Truth*. https://en.wikipedia.org/wiki/The_Moment_of_Truth_(American_game_show). (Visitado 08-01-2022).

[83] *Do lie detectors actually work?* https://www.paulekman.com/blog/do-lie-detectors-actually-work/. (Visitado 08-01-2022).

[84] Lisa Feldman Barrett y col. "Emotional Expressions Reconsidered: Challenges to Inferring Emotion From Human Facial Movements". En: *Psychological Science in the Public Interest* 20.1 (2019). PMID: 31313636, págs. 1-68. DOI: 10.1177/1529100619832930. eprint: https://doi.org/10.1177/1529100619832930. URL: https://doi.org/10.1177/1529100619832930.

[85] *MICRO-EXPRESSIONS: FACT OR FICTION?* https://www.tsi-mag.com/micro-expressions-fact-or-fiction/. (Visitado 12-01-2022).

[86] Bella DePaulo y col. "Cues to Deception". En: *Psychological bulletin* 129 (feb. de 2003), págs. 74-118. DOI: 10.1037/0033-2909.129.1.74.

[87] *MICRO-EXPRESSIONS: FACT OR FICTION?* https://www.nytimes.com/2014/03/02/opinion/sunday/what-faces-cant-tell-us.html. (Visitado 12-01-2022).

[88] Stephen Porter y Leanne ten Brinke. "Reading Between the Lies: Identifying Concealed and Falsified Emotions in Universal Facial Expressions". En: *Psychological Science* 19.5 (2008). PMID: 18466413, págs. 508-514. DOI: 10.1111/j.1467-9280.2008.02116.x. eprint: https://doi.org/10.1111/j.1467-9280.2008.02116.x. URL: https://doi.org/10.1111/j.1467-9280.2008.02116.x.

[89] Sarah Jordan y col. “A test of the micro-expressions training tool: Does it improve lie detection?” En: *Journal of Investigative Psychology and Offender Profiling* 16.3 (2019), págs. 222-235. DOI: https://doi.org/10.1002/jip.1532. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jip.1532. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jip.1532.

[90] Waqas Sultani, Chen Chen y Mubarak Shah. “Real-world anomaly detection in surveillance videos”. En: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, págs. 6479-6488.

[91] *Video Anomaly Dection Dataset*. https://webpages.charlotte.edu/cchen62/dataset.html. Accedido: 9 de Julio, año 2022.

[92] *Inception V3*. https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=es-419. Accedido: 23 de Junio, año 2022.

[93] *Open CV*. https://pypi.org/project/opencv-python/. Accedido: 23 de Junio, año 2022.

[94] *Open CV Github*. https://github.com/opencv/opencv-python. Accedido: 23 de Junio, año 2022.

[95] *RGB standard*. https://www.w3.org/Graphics/Color/sRGB.html. Accedido: 23 de Junio, año 2022.

[96] Farhana Sultana, Abu Sufian y Paramartha Dutta. “Advancements in Image Classification using Convolutional Neural Network”. En: *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*. IEEE, nov. de 2018. DOI: 10.1109/icrcicn.2018.8718718. URL: https://doi.org/10.1109%2Ficrcicn.2018.8718718.

[97] *Using Convolutional Neural Network for Image Classification*. https://towardsdatascience.com/using-convolutional-neural-network-for-image-classification-5997bfd0ede4. Accedido: 27 de Junio, año 2022.

[98] *Deep Learning: Understanding The Inception Module*. https://towardsdatascience.com/deep-learning-understand-the-inception-module-56146866e652. Accedido: 1 de Julio, año 2022.

[99] *Inception V3 library - Keras*. https://keras.io/api/applications/inceptionv3/. Accedido: 29 de Junio, año 2022.

[100] *Real Life Violence Detection: Using Inception V3*. https://github.com/NANDINI-star/Real-life-violence-detection/. Accedido: 29 de Agosto, año 2022.

[101] *Keras*. https://keras.io/. Accedido: 27 de Junio, año 2022.

[102] *TensorFlow*. https://www.tensorflow.org/. Accedido: 27 de Junio, año 2022.

[103] *Keras AveragePooling2D*. https://keras.io/api/layers/pooling_layers/average_pooling2d/. Accedido: 28 de Junio, año 2022.

[104] *2D Average pooling*. https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/blocks/2d-average-pooling. Accedido: 28 de Junio, año 2022.

[105] *Keras Flatten*. https://keras.io/api/layers/reshaping_layers/flatten/. Accedido: 29 de Junio, año 2022.

[106] *Keras Dense*. https://keras.io/api/layers/core_layers/dense/. Accedido: 4 de Julio, año 2022.

[107] *Keras Activation Functions*. https://keras.io/api/layers/activations/. Accedido: 4 de Julio, año 2022.

[108] *Keras Dropout*. https://keras.io/api/layers/regularization_layers/dropout/. Accedido: 4 de Julio, año 2022.

[109] *Separable Conv 2D - Keras*. https://www.tensorflow.org/api_docs/python/tf/keras/layers/SeparableConv2D. Accedido: 29 de Agosto, año 2022.

[110] *Batch Normalization - Keras*. https://keras.io/api/layers/normalization_layers/batch_normalization/. Accedido: 29 de Agosto, año 2022.

[111] *Max Pooling 2D - Keras*. https://keras.io/api/layers/pooling_layers/max_pooling2d/. Accedido: 29 de Agosto, año 2022.

[112] *Bidirectional Layer - Keras*. https://keras.io/api/layers/recurrent_layers/bidirectional/. Accedido: 29 de Agosto, año 2022.

[113] *TensorFlow*. https://www.tensorflow.org/. Accedido: 29 de Agosto, año 2022.

[114] *Precision metric - Keras*. https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Precision. Accedido: 29 de Agosto, año 2022.

[115] *Recall metric - Keras*. https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Recall. Accedido: 29 de Agosto, año 2022.

[116] *F1Score metric - Keras*. https://www.tensorflow.org/addons/api_docs/python/tfa/metrics/F1Score. Accedido: 29 de Agosto, año 2022.

[117] *Accuracy metric - Keras*. https://keras.io/api/metrics/accuracy_metrics/. Accedido: 29 de Agosto, año 2022.

8. Anexos

8.1. Procesamiento de los videos para la entrada al modelo

A continuación se explica el código que realiza el pre-procesamiento de los videos para formar la entrada al modelo de Inteligencia Artifical.

Para comenzar, se definen las etiquetas para los videos. Se utiliza "Violence" como nombre de la columna, conteniendo la misma valores booleanos. Estas etiquetas indican la ocurrencia de actos violentos en cada video. En el siguiente paso, según el valor en la columna "CATEGORY" en cada fila (es decir la categoría de cada video), se asigna el valor booleano correspondiente. "CATEGORY" puede contener hasta 5 valores diferentes según el dataset que fue utilizado (Burglary, Assault, Fighting, Abuse y Normal Videos) en los que para cualquiera de ellos, exceptuando "Normal Videos", se les asigna el valor "True" a los que contienen un crimen y "False" en caso contrario.

```
Video_Labels = list(map(lambda x: os.path.split(os.path.split(x)[0])[1], Video_Path))
Video_Path_Series = pd.Series(Video_Path,name="MP4").astype(str)
Video_Labels_Series = pd.Series(Video_Labels,name="CATEGORY")
Video_Violence_Series = pd.Series(np.where(Video_Labels_Series == 'Normal Videos', False
    , True), name='VIOLENCE')

Main_MP4_Data = pd.concat([Video_Path_Series, Video_Labels_Series, Video_Violence_Series],
    axis=1)
Main_MP4_Data
```

Realizado esto se hace un *drop* de la columna "CATEGORY", ya que no será necesaria para el modelo, y basta (de hecho es deseable) con tener solamente el video y la etiqueta de si es un crimen o no. Acto seguido se utiliza la librería de Python OpenCV [93] para extraer y redimensionar los *frames* de cada video, como se menciona en la sección 4 del informe.

```
import cv2
violence_frame_list = []
violence_label = []

for index,row in violence_df.iterrows():
  Video_File_Path = row['MP4']
  Video_Name = Video_File_Path.split("/")[-1]

  Video_Caption = cv2.VideoCapture(Video_File_Path)
  Frame_Rate = Video_Caption.get(5)
  current_video_frames = []

```

```
while Video_Caption.isOpened():

  Current_Frame_ID = Video_Caption.get(1)
  ret,frame = Video_Caption.read()

  if ret != True:
    break

  if Current_Frame_ID % math.floor(Frame_Rate) == 0:
    Frame_Resize = cv2.resize(frame,(224,224))
    current_video_frames.append(Frame_Resize)
 violence_frame_list.append(current_video_frames)
 violence_label.append(row['VIOLENCE'])


Video_Caption.release()
```

Se realiza este proceso tanto para los videos de crímenes como para los videos normales. Teniendo todos los *frames* del video y el *framerate* (cantidad de *frames* en un segundo), definido en la etapa anterior, se pueden agrupar los *frames* del video según el segundo en el que ocurren, tomando la misma cantidad de *frames* seguidos que de *framerate*.

```
frames_per_second = 5
seconds_of_violence = []
for violence_video in violence_frame_list:
  seconds_of_video = [violence_video[x:x+frames_per_second] for x in range(0, len(
    violence_video), frames_per_second)]
  result = [a for a in seconds_of_video if len(a) == 5]
  if(len(seconds_of_violence) == 0):
    seconds_of_violence = result
  elif (len(result) > 0):
    seconds_of_violence = np.concatenate((seconds_of_violence, result), axis = 0)
```

Realizando el mismo procedimiento tanto para los videos criminales como no criminales, se procede a juntarlos a todos en un mismo conjunto:

```
labels = np.concatenate(([ 'Violence']*len(seconds_of_violence), ['NonViolence']*len(
    seconds_of_no_violence)), axis=0)
```

Por último, se mezclan las filas del *dataset* (videos con crímenes con videos ordinarios) para evitar mala performance en el modelo. Se hace *one-hot-encoding* de las etiquetas, para que sea mas fácil de digerir para el modelo. El último paso es partir el *dataset* en conjunto de entrenamiento y conjunto de evaluación.

```
#Mezclar violencia y no violencia
def unison_shuffled_copies (a, b):
  assert len(a) == len(b)
  p = np.random.permutation(len(a))
  return a[p], b[p]
[train_X, train_Y] = unison_shuffled_copies (seconds, labels)

#Se realiza one-hot encoding rn las etiquetas
lb = LabelBinarizer ()
train_Y = lb. fit_transform (train_Y)
train_Y = to_categorical (train_Y)

# Particion de los datos en splits de entrenamiento y testing (75 % y 25 %
    respectivamente)
(trainX, testX, trainY, testY) = train_test_split (train_X, train_Y, test_size =0.25,
    stratify =train_Y, random_state=42)
```

8.2. *Etiquetado de los videos con modelo Inception*

```
import cv2
import keras
import numpy as np

violence_model = keras.models.load_model('inception_v3_large .h5') # Cargar el modelo
out_file = "violence_recognizer_2 .avi" #Setear el archivo de salida .

def recognize_video (video_path):
  player = get_video_stream (video_path) #Acceder al stream del video.
  assert player.isOpened() # Asegurar que es un stream y que es accesible
  Frame_Rate = player.get(5)
  x_shape = int(player.get(cv2.CAP_PROP_FRAME_WIDTH))
  y_shape = int(player.get(cv2.CAP_PROP_FRAME_HEIGHT))
  four_cc = cv2.VideoWriter_fourcc(*"MJPG")
  out = cv2.VideoWriter( out_file , four_cc , 20, \
              (x_shape, y_shape))
  ret, frame = player.read() # S lee el primer frame
  video_frames_results = []
  while ret: # Leemos hasta que se terminen los frames
    Current_Frame_ID = player.get(1)
    Frame_Resize = cv2. resize (frame,(224,224)) #Redimensionar el frame.
    results = score_frame(np. asarray ([Frame_Resize])) # Evaluamos el frame con nuestro
    modelo
    video_frames_results .append( results )
    frame = plot_score ( results , frame) # Ploteamos los resultados .
    out.write(frame) # Escribimos el frame en el archivo de salida
```

```
26
27      ret, frame = player.read() # leemos el proximo frame.
28
29    print( video_frames_results [np.argmax( video_frames_results )])
30
31 def get_video_stream (video_path):
32   stream = cv2.VideoCapture(video_path) #obtiene el stream.
33    return stream
34
35 def score_frame(frame):
36    Model_Test_Prediction = violence_model.predict(frame)
37    print ( Model_Test_Prediction )
38    return Model_Test_Prediction
39
40 def plot_score ( results , frame):
41   bgr = (0, 255, 0)
42   score = results [0]
43   if score[1] > 0.5: # Si el score del modelo es superior a 0.5 lo catalogamos como
        violencia y de los contrario como no violencia
44      label = 'Violence'
45   else:
46      label = 'Non-Violence'
47    label_font = cv2.FONT_HERSHEY_SIMPLEX
48    label_size = 0.5
49   textX = (frame.shape[1]) / 2
50   textY = (frame.shape[0]) / 2
51   cv2.putText(frame, "Score: " + label , (20, 20), label_font , 0.9, bgr, 2)
52    return frame
53
54 recognize_video ("Assault002_x264.mp4")
```

Se explica brevemente el código, para una mejor comprensión del mismo. La función *recognize_video*, recibe como parámetro la ruta de un vídeo a evaluar y etiquetar, procediendo a obtener el *stream* del mismo y recolectar información adicional como sus dimensiones y el *framerate*, que serán utilizados mas adelante.

Una vez hecho esto, lee *frame* a *frame* a través del *stream* y evalúa cada uno de ellos con el modelo cargado en memoria (en la linea 5). El *score* otorgado por el modelo es una probabilidad que indica la presencia de violencia en cada *frame*. Una etiqueta de *Violence* o de *No Violence* es ploteada sobre una esquina del *frame* original del vídeo y escrito en el vídeo de salida. La etiqueta dependerá de si el valor del *score* es mayor a 0.5 (violencia) o menor (no violencia).

De esta forma, se obtiene un vídeo de salida, con cada uno de los *frames* del vídeo original con su respectiva etiqueta asignada por el modelo de inteligencia artificial diseñado en este trabajo.

8.3. *Etiquetado de los videos con modelo LSTM*

```
import cv2
import time
import keras
import numpy as np
import math

violence_model = keras.models.load_model('LSTM time distributed.h5') # Cargar el modelo
    .
out_file = "violence_recognizer_2.avi" # Nombre asignado al output.

video_path = "full-dataset/Burglary/Burglary091_x264.mp4"

def recognize_video(video_path):
    player = get_video_stream(video_path) # Conseguir el video stream.
    frames_of_the_video = process_video(player)
    prediction = violence_model.predict([frames_of_the_video], batch_size=10)
    print(prediction)


def process_video(player):
    assert player.isOpened() # Asegurarse que es un stream.
    Frame_Rate = player.get(5)
    # El codigo siguiente crea un nuevo objeto para
    # escribir el output de nuestro video
    x_shape = int(player.get(cv2.CAP_PROP_FRAME_WIDTH))
    y_shape = int(player.get(cv2.CAP_PROP_FRAME_HEIGHT))
    four_cc = cv2.VideoWriter_fourcc(*"MJPG")
    out = cv2.VideoWriter(out_file, four_cc, 20, \
                          (x_shape, y_shape))
    ret, frame = player.read()
    video_frames = []
    video_frames_raw = []
    video_frames_results = []
    while ret: # Recorrer todos los frames
        video_frames_raw.append(frame)
        Current_Frame_ID = player.get(1)
        if Current_Frame_ID % math.floor(Frame_Rate) == 0:
            Frame_Resize = cv2.resize(frame,(224,224))
            video_frames.append(Frame_Resize)
        ret, frame = player.read()
    frames_per_second = 5
    seconds_of_violence = []

    for violence_video in [video_frames]:
```

```
44          seconds_of_video = [violence_video[x:x+frames_per_second] for x in range(0, len(
    violence_video), frames_per_second)]
45          result = [a for a in seconds_of_video if len(a) == 5]
46          if(len(seconds_of_violence) == 0):
47              seconds_of_violence = result
48          elif (len(result) > 0):
49              seconds_of_violence = np.concatenate((seconds_of_violence, result), axis =
    0)
50      video_frames_results = violence_model.predict([seconds_of_violence], batch_size
    =10)
51
52      player = get_video_stream(video_path)
53      ret, frame = player.read()
54      while ret:
55          Current_Frame_ID = player.get(1)
56          bucket_index = math.trunc((Current_Frame_ID / Frame_Rate) / 5)
57          if(bucket_index < len(video_frames_results)):
58              scored_frame = plot_score(video_frames_results[bucket_index], frame)
59              out.write(scored_frame)
60          ret, frame = player.read()
61
62      return seconds_of_violence
63  def get_video_stream(video_path):
64      stream = cv2.VideoCapture(video_path)
65      return stream
66
67  def score_frame(frame):
68      Model_Test_Prediction = violence_model.predict(frame)
69      print(Model_Test_Prediction)
70      return Model_Test_Prediction
71
72  def plot_score(results, frame):
73      bgr = (0, 255, 0)
74      if results[1] > 0.5: # Si el score es mas grande que 0.5, se define el video como
    violento.
75          label = 'Violence'
76      else:
77          label = 'Non-Violence'
78      label_font = cv2.FONT_HERSHEY_SIMPLEX
79      label_size = 0.5
80      cv2.putText(frame, "Score: " + label, (20, 20), label_font, 0.9, bgr, 2)
81      return frame
82
83  recognize_video(video_path) # Correr el codigo en el video.
```

De igual forma que en el código de etiquetado anterior, la función *recognize_video*, recibe como parámetro la ruta de un vídeo a evaluar y etiquetar.

Comienza obteniendo el *stream* y leyendo un *frame* por segundo del video. Una vez hecho esto, crea segmento con un tamaño de 5 *frames*, que forman una secuencia. De esta forma, el video esta listo para ser entrada del modelo. Posteriormente, se evalúa cada secuencia y su valor es ploteado en todos los frames que pertenecen a esta. La etiqueta de violencia/no violencia nuevamente dependerá del valor de la evaluación del modelo, donde un resultado mayor a 0.5 indica *"Violencia"* y uno menor *"No violencia"*.

De esta forma, se obtiene un vídeo de salida, donde cada segmento del video esta etiquetado como violento o no violento.