



Universidad Internacional de la Rioja (UNIR)

Escuela Superior de Ingeniería y Tecnología

Máster en Ingeniería Matemática y Computación

Optimización de un pipeline basado en BI-LSTM para la detección de violencia en video

Trabajo Fin de Estudios

presentado por: Cesar Antonio Madera Garcés

Dirigido por: Pablo Negre Rodriguez

Ciudad: Lima, Perú

Fecha: 23 de Marzo de 2025

Índice de Contenidos

Resumen	v
Abstract	vi
1. Introducción	1
1.1. Justificación	1
1.2. Enfoque de Trabajo	2
1.3. Estructura del Documento	3
2. Contexto	4
2.1. Tipos de Violencia	4
2.2. Soluciones Clásicas	6
2.3. Soluciones tecnológicas	7
2.4. Detección de violencia a través de IA	9
2.4.1. <i>Convolutional Neural Networks</i> (CNN)	9
2.4.2. VGG-19	11
2.4.3. InceptionV3	11
2.4.4. ResNet50	12
2.4.5. EfficientNet	13
2.4.6. YOLO	15
2.4.7. <i>Recurrent Neural Networks</i> (RNN's)	16
2.4.8. <i>Long Short-Term Memory</i> (LSTM)	16
2.4.9. Transformers	17
2.4.10. Algoritmos basados en la captura esquelética	20
2.5. <i>Transfer Learning</i>	20
2.5.1. <i>Pipelines</i> actuales para la clasificación de violencia	21
3. Objetivos	24
3.1. Objetivos	24
3.2. Contribuciones	24
4. Metodología	26
4.1. Descripción del <i>Dataset</i>	26

4.2. Pre-procesamiento de la base de datos	26
4.3. Métricas de Evaluación	28
4.4. Software y hardware utilizados	28
4.4.1. Evaluación de CNN's para la experimentación	28
5. Desarrollo del trabajo	32
5.1. Pipeline propuesto	32
5.1.1. Extracción de frames y preprocesamiento	32
5.1.2. Obtención de las características	33
5.1.3. Entrenamiento y prueba del clasificador	34
5.2. Experimentación	34
6. Conclusiones y Trabajo Futuro	41
Referencias	42
A. Apendices	48

Índice de Ilustraciones

2.1. Percepción de que la inseguridad es una prioridad (Bisca y cols., 2024)	5
2.2. Proceso simplificado de una convolución(Diego Calvo, 2019a).	10
2.3. Proceso simplificado de un pooling (Muhamad Yani, Budhi Irawan, Casi Setianingsih, 2019).	10
2.4. Arquitectura de una CNN convencional (Diego Calvo, 2019b).	11
2.5. Version simplificada de una VGG19 (IchiPro, 2020).	12
2.6. Reducción de dimensionalidad de InceptionV3 (IchiPro, 2020).	13
2.7. Versión simplificada de InceptionV3 (IchiPro, 2020).	13
2.8. Versión simplificada de ResNet50 (IchiPro, 2020).	14
2.9. Versión simplificada de EfficientNetB0 (Tashin Ahmed, 2020).	14
2.10. Representación de la arquitectura de YOLOv5(Jocher y cols., 2022).	15
2.11. Versión simplificada de RNN (Amidi, 2018).	16
2.12. Representación de la arquitectura de LSTM(DataScientest, 2024).	18
2.13. Representación de la arquitectura de un transformer(Services, 2024).	19
2.14. Comparativa entre un entrenamiento normal y por TL(Wenjin Taoa, Md Al-Aminb, Haodong Chena, Ming C. Leua, Zhaozheng Yinc, Ruwen Qinb, 2020).	21
4.1. Comparación entre frames con y sin violencia en el “Hockey Fight Detection Dataset”	27
5.1. Metodología propuesta. Creación propia.	32
5.2. Gráficos de pérdida y f1 score por época usando EfficientNetB0	35
5.3. Gráficos de pérdida y f1 score por época usando EfficinetnetV2-S	36
5.4. Gráficos de pérdida y f1 score por época usando MobilenetV3	36
5.5. Gráficos de pérdida y f1 score por época usando Resnet50	37

Índice de Tablas

4.1. Evaluación de diferentes modelos. Tabla brindada por Orhan Yalcin (Yalcin, 2020).	30
4.2. Datos obtenidos de la experimentación previa con dataset de peces	31
5.1. Tabla comparativa de las métricas obtenidas por configuración de celdas para EfficientNetB0.	37
5.2. Tabla comparativa de las métricas obtenidas por configuración de celdas para EfficientnetV2.	38
5.3. Tabla comparativa de las métricas obtenidas por configuración de celdas para MobileNetV3.	39
5.4. Tabla comparativa de las métricas obtenidas por configuración de celdas para Resnet50.	39
5.5. Tabla comparativa resumen final de la experimentación	40

Resumen

Nota: En este apartado se introducirá un breve resumen en español del trabajo realizado (extensión máxima: 150 palabras). Este resumen debe incluir el objetivo o propósito de la investigación, la metodología, los resultados y las conclusiones.

Palabras Clave: Se deben incluir de 3 a 5 palabras claves en español

Abstract

Nota: En este apartado se introducirá un breve resumen en español del trabajo realizado (extensión máxima: 150 palabras). Este resumen debe incluir el objetivo o propósito de la investigación, la metodología, los resultados y las conclusiones.

Palabras Clave: Se deben incluir de 3 a 5 palabras claves en inglés

1. Introducción

La violencia sigue siendo un problema global crítico, con millones de incidentes reportados anualmente. Según la Organización Mundial de la Salud (OMS), la violencia interpersonal ha permanecido como una de las 10 principales causas de muerte anual en las regiones de las Américas, con innumerables casos de agresión física y delitos violentos que no se reportan (Organization, 2024). América Latina, en particular, tiene algunas de las tasas de violencia más altas, con países como Perú, Chile, Brasil, Colombia y México enfrentando importantes desafíos en la prevención del crimen y la seguridad pública (Bisca y cols., 2024). En 2024, el Instituto Nacional de Estadística y Geografía (INEGI) reportó 21.9 millones de víctimas mayores de edad solo en México, y 31.3 millones de delitos (INEGI, 2024). La creciente disponibilidad de videovigilancia y medios digitales presenta una oportunidad para desarrollar sistemas automatizados capaces de detectar y mitigar incidentes violentos en tiempo real.

La inteligencia artificial (IA) ha emergido como una herramienta potente en el campo del análisis de video, ofreciendo soluciones prometedoras para la detección automatizada de violencia. Los modelos de aprendizaje profundo, particularmente las redes neuronales convolucionales (CNNs) y las redes de memoria a largo y corto plazo (LSTM), han demostrado capacidades excepcionales en el procesamiento de características espaciotemporales de los datos de video (Orozco, Buemi, y Berlles, 2021). Aprovechando estas tecnologías, los sistemas basados en IA pueden analizar flujos de video, reconocer acciones violentas y generar alertas con alta precisión. Sin embargo, desafíos como el desequilibrio de clases, la escasez de datos y los falsos positivos siguen siendo obstáculos críticos en las aplicaciones del mundo real (Kulkarni, Batarseh, y Chong, 2021). Esta investigación tiene como objetivo mejorar la robustez e interpretabilidad de los sistemas de detección de violencia impulsados por IA, contribuyendo a entornos más seguros en México y más allá.

1.1. Justificación

Las tasas crecientes de violencia en América Latina, particularmente en México como se expuso en la sección anterior, subrayan la urgente necesidad de sistemas avanzados de videovigilancia capaces de detectar incidentes en tiempo real. Los enfoques tradicionales de monitoreo, que dependen de la supervisión humana, a menudo son ineficientes debido a

la fatiga cognitiva y las limitaciones en la escalabilidad (Marois, Hodgetts, Chamberland, Williot, y Tremblay, 2021). La inteligencia artificial (IA), particularmente el aprendizaje profundo, ha demostrado un gran potencial para automatizar la detección de violencia mediante la integración de redes neuronales convolucionales (CNNs) y redes de memoria a largo y corto plazo (LSTM) (Negre, Alonso, Prieto, Dang, y Corchado, 2024; Negre, Alonso, Prieto, Garcia, y Corchado, 2024; Abdali y Al-Tuma, 2019; Sharma, Sudharsan, Naraharisetti, Trehan, y Jayavel, 2021). Mientras que las CNNs extraen características espaciales de los fotogramas de video, las LSTMs capturan dependencias temporales, lo que las convierte en una combinación poderosa para analizar escenas dinámicas. Sin embargo, el diseño óptimo de estos modelos sigue siendo un desafío abierto, ya que las variaciones en las arquitecturas de CNN y las configuraciones de LSTM afectan directamente la precisión de la detección, la eficiencia computacional y la aplicabilidad en el mundo real.

Este estudio busca investigar sistemáticamente el balance entre diferentes extractores de características de CNN y el número de celdas LSTM para determinar la pipeline más efectiva para la detección de violencia. La elección de la CNN influye en la calidad de la extracción de características, mientras que el número de celdas LSTM impacta en la capacidad del modelo para capturar patrones temporales sin incurrir en costos computacionales excesivos. Al optimizar este balance, la investigación busca mejorar tanto el rendimiento como la eficiencia de los sistemas de detección de violencia impulsados por IA. Los resultados contribuirán no solo al avance académico del análisis espaciotemporal de video, sino también al despliegue práctico de soluciones de videovigilancia robustas y escalables, mejorando finalmente la seguridad pública en México y más allá.

A continuación se especifican tanto el enfoque de este trabajo y cual será la estructura del resto del documento.

1.2. Enfoque de Trabajo

Habiendo establecido la justificación para esta investigación, es evidente que la selección de técnicas de extracción de características y el número de celdas LSTM juegan un papel crucial en la optimización de los modelos de detección de violencia. Los enfoques existentes a menudo pasan por alto el balance entre estos dos factores, lo que puede limitar el rendimiento en aplicaciones del mundo real. Por lo tanto, este estudio tiene como objetivo evaluar y refinar el balance entre los extractores de características de CNN y las

configuraciones de celdas LSTM para desarrollar una pipeline más eficiente y precisa para la detección automatizada de violencia.

1.3. Estructura del Documento

Este documento se organiza en varios apartados que permiten comprender el desarrollo completo del trabajo. Primero, se expone el contexto del problema y su relevancia. Luego, se definen los objetivos de la investigación. La sección de metodología describe los datos utilizados, el preprocesamiento y el enfoque técnico adoptado. A continuación, en el desarrollo del trabajo, se detallan los experimentos realizados, las configuraciones evaluadas y los resultados obtenidos. Finalmente, se presentan las conclusiones y se proponen posibles líneas de trabajo futuro.

2. Contexto

El presente capítulo proporciona un análisis detallado sobre el problema de la violencia y las metodologías actuales para su detección en video, ambos fundamentales para el desarrollo de esta investigación. Se aborda la clasificación de los diferentes tipos de violencia y su impacto global, así como una revisión exhaustiva de los modelos más utilizados en la literatura para identificar eventos violentos en entornos audiovisuales.

La Sección 2.1, establece una categorización de diversas formas de violencia, la cual destaca su ocurrencia en diferentes contextos como espacios públicos, entornos domésticos y situaciones de conflicto. Además, se presentan estadísticas relevantes para ilustrar la magnitud del problema a nivel global, con un enfoque particular en América Latina y sus tendencias recientes.

Por otra parte la Sección 2.2, se listarán las soluciones que se están utilizando actualmente para la detección y toma de decisiones punitivas de violencia mencionadas en la anterior sección.

Por último, la Sección 2.4, se revisa sistemáticamente los enfoques más representativos en la literatura para la detección de violencia en video. Cada metodología ha sido clasificada en función de su fundamento teórico y técnico, distinguiendo entre modelos basados en extracción manual de características, redes neuronales convolucionales (CNNs) y enfoques híbridos. Finalmente, se discute la necesidad de evaluar diferentes configuraciones de estos modelos para optimizar su rendimiento en la identificación de incidentes violentos.

2.1. Tipos de Violencia

A continuación se explica cada uno de los tipos de violencia clasificados por la OMS y como estos han afectado a la población a lo largo de los años

La proliferación de la violencia en todas partes del mundo constituye un problema social creciente que afecta la convivencia y el sentido de seguridad entre las personas. Dependiendo de las características de quienes cometen el acto, la violencia puede clasificarse en las siguientes categorías (OMS, 2014):

- Autoinfligida (conducta suicida y autolesiones),
- Interpersonal (violencia doméstica, incluyendo a niños, parejas y personas mayores; así como violencia entre personas no relacionadas),

- Colectiva (social, política y económica).

La OMS clasifica los actos violencia según la naturaleza como: física, sexual, psicológica, privación y negligencia. Con base en datos de 2014, indicó que “los actos repetidos de violencia que van desde la intimidación, el acoso sexual y las amenazas hasta la humillación y el menosprecio de los trabajadores pueden convertirse en casos muy graves debido al efecto acumulativo. En Suecia, se estima que dicho comportamiento ha sido un factor en el 10 % al 15 % de los suicidios”. En el mismo documento, se menciona que en el año 2000, hubo alrededor de 199,000 homicidios de jóvenes en todo el mundo (9.2 por cada 100,000 habitantes). Es decir, en promedio, mueren diariamente 565 niños, adolescentes y adultos jóvenes de entre 10 y 29 años como resultado de la violencia interpersonal. Las tasas de homicidio varían considerablemente según la región, desde 0.9 por cada 100,000 habitantes en países de altos ingresos en Europa y algunas partes de Asia y el Pacífico hasta 17.6 en África y 36.4 por cada 100,000 en América Latina.

Por otro lado, el informe del Fondo Monetario Internacional revela un aumento exponencial en la sensación de inseguridad y una mayor aceptación de que los crímenes violentos son, unánimemente, el problema más importante desde 2020, como se muestra en la Figura 2.1 (Bisca y cols., 2024):

1. Perception that Crime and Insecurity Are the Main Problems Facing the Country (Percentage of respondents)

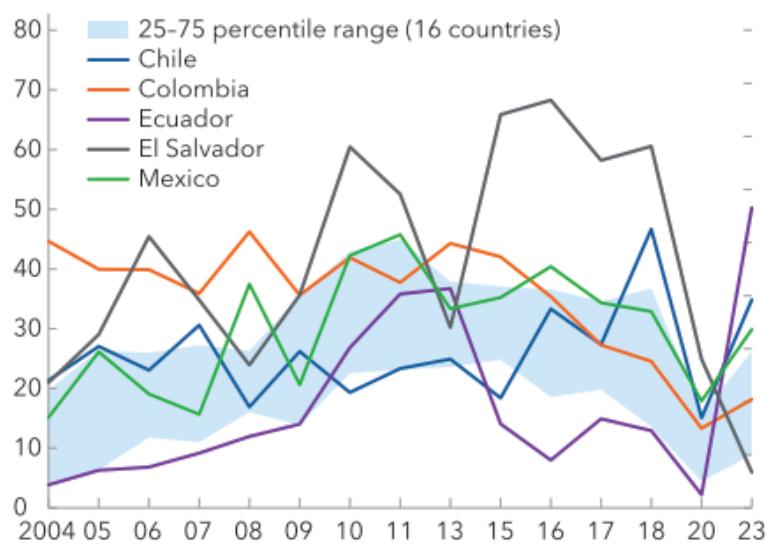


Figura 2.1: Percepción de que la inseguridad es una prioridad (Bisca y cols., 2024)

La Figura 2.1 muestra las prioridades con respecto al tiempo de la población en di-

ferentes países latinoamericanos. La percepción de la prioridad de la violencia venía en decadencia desde el 2004, mientras que en los últimos 5 años, esta percepción ha mostrado una tendencia exponencialmente incremental. En este sentido, en la actualidad, la violencia representa la prioridad más importante que cualquier gobierno debería abordar. Por esta razón, en este proyecto se aborda la solución de este problema a través de la detección de violencia interpersonal directa, a través del uso de inteligencia artificial.

2.2. Soluciones Clásicas

En la presente sección se presentan las soluciones que se han aplicado desde tiempos antiguos, las cuales mayormente se han enfocado en el uso de la policía o milicia para el apaciguamiento y reacción ante incidentes violentos.

Desde la antigüedad, los Estados han utilizado la organización militar como principal herramienta para contener la violencia interna y establecer el orden. En imperios como el romano o el chino, los soldados eran distribuidos estratégicamente en territorios clave no solo para la defensa externa, sino para prevenir revueltas, aplicar justicia y garantizar la autoridad del Estado(Gilliver, 2005). Estas asignaciones eran determinadas por el mando central según criterios geográficos, demográficos y políticos, pero siempre con una intención clara: imponer un monopolio legítimo de la fuerza, siguiendo lo que Max Weber definiría siglos después como una característica esencial del Estado moderno.

A partir del siglo XIX, con la consolidación de cuerpos policiales profesionales en Europa y América, surgieron modelos más especializados para combatir la violencia interna, especialmente en contextos urbanos crecientes. Las asignaciones de personal policial se realizaban mediante mapas manuales de criminalidad, análisis de denuncias y supervisión directa, y buscaban prevenir delitos, desarticular disturbios y pacificar zonas consideradas conflictivas(Emsley, 2007). Sin embargo, estas asignaciones eran altamente dependientes de la interpretación subjetiva de los mandos y muchas veces respondían a presiones políticas o sesgos institucionales, lo que afectaba su efectividad en la reducción sostenible de la violencia.

Hasta inicios de los años 1990, tanto en contextos militares como policiales, la distribución de fuerzas para combatir la violencia interna se realizaba sin apoyo computacional, lo que limitaba la capacidad para prever patrones delictivos o reasignar recursos eficientemente. La planificación se basaba en documentos físicos, experiencia de campo y órdenes

jerárquicas, lo cual implicaba lentitud en la respuesta y una escasa adaptación al cambio (Sheptycki, 1998). Esta rigidez estructural dificultaba la aplicación de estrategias preventivas de largo plazo y muchas veces resultaba en una gestión reactiva de la violencia, en lugar de su erradicación sistemática o su abordaje desde perspectivas más integrales.

2.3. Soluciones tecnológicas

Debido a la ineficiencia de las soluciones clásicas para este problema, se identificó la subjetividad de dichas soluciones y se comenzaron a utilizar soluciones más enfocadas en datos estadísticos para su implementación. En la presente sección se detallan las soluciones tecnológicas propuestas.

El primer sistema de vigilancia fue creado en la década de 1960, inicialmente en Londres, Reino Unido, como una estrategia para monitorear el tráfico. Sin embargo, en los años 1990, su propósito se expandió al control del crimen y la violencia urbana. Las cámaras de circuito cerrado de televisión (CCTV) fueron instaladas en zonas de alto riesgo, estaciones de tren, centros comerciales y calles céntricas, con el objetivo de disuadir actos violentos, registrar evidencia visual de delitos y mejorar la capacidad de respuesta de las fuerzas policiales. A diferencia de tecnologías modernas que dependen de IA para análisis automático, estos sistemas tradicionales requieren monitoreo humano en centros de control y revisión manual de grabaciones(Vilalta, Sanchez, Fondevila, y Ramirez, 2019).

Uno de los primeros sistemas analíticos propuestos fue CompStat, en Estados Unidos, el cual fue desarrollado por la policía de Nueva York en 1994, para mejorar la eficacia de la respuesta policial ante el crimen. Este sistema de gestión basado en datos permite a los departamentos de policía analizar de manera detallada las tendencias delictivas mediante la recopilación, el análisis y la distribución de estadísticas del crimen. CompStat no solo se utiliza para mejorar la respuesta ante incidentes de violencia, sino también para asignar recursos de manera más eficiente, identificar patrones de comportamiento delictivo y ajustar las estrategias de intervención en tiempo real. La recolección de datos, combinada con reuniones periódicas con líderes de la policía, permite una supervisión más directa y una rendición de cuentas más eficaz de las operaciones policiales(Police Executive Research Forum, 2003).

Tras su implementación en Nueva York, CompStat fue adoptado por diversas agencias de policía en los Estados Unidos y otras partes del mundo(Weisburd, Mastrofski, McNally,

Greenspan, y Willis, 2003). Su enfoque basado en datos ha permitido reducir la violencia en algunas de las ciudades más grandes del país, como Los Ángeles y Chicago. Además de su aplicación en crímenes violentos, el sistema ha demostrado ser efectivo en la mejora de la eficiencia operativa, permitiendo a las fuerzas de seguridad adaptar rápidamente sus tácticas ante la dinámica cambiante de los delitos. El éxito de CompStat ha llevado a su integración con otras soluciones de análisis de datos y monitoreo de crímenes, convirtiéndose en una herramienta clave en la estrategia de reducción del crimen a nivel mundial.

Otra solución más reactiva fue el Botón de pánico en México. El botón de pánico es una solución tecnológica diseñada para proporcionar seguridad inmediata en situaciones de violencia(de la Ciudad de México, 2008). Inicialmente desarrollado en México, este dispositivo permite a las víctimas de violencia, ya sea doméstica o en la vía pública, enviar una señal de alerta a las autoridades o familiares cercanos con solo presionar un botón. Este sistema puede estar integrado en un dispositivo móvil o en un dispositivo independiente que envíe un mensaje de emergencia con la ubicación geográfica exacta del usuario, lo que facilita una respuesta rápida por parte de las autoridades, como por ejemplo en un poste. Aunque se originó en México, ha sido implementado en varios países de América Latina como parte de las políticas públicas de seguridad.

El uso de botones de pánico ha sido altamente efectivo en mejorar la respuesta de las fuerzas de seguridad ante emergencias. Este dispositivo ha sido especialmente útil en situaciones de violencia doméstica, donde la víctima puede no tener la capacidad de realizar una llamada telefónica o salir de su hogar. La solución también ha sido integrada en algunas plataformas gubernamentales, y en algunos casos, se han vinculado directamente a sistemas de videovigilancia y patrullaje policial, lo que reduce significativamente el tiempo de respuesta ante situaciones de violencia.

Por otra parte, en El Salvador, el Sistema de Alerta Temprana fue desarrollado en 2012 para monitorear y alertar sobre incidentes violentos, especialmente relacionados con pandillas y crimen organizado. Este sistema se basa en la recolección de datos a través de diversas fuentes, como reportes ciudadanos, medios de comunicación, redes sociales y bases de datos oficiales. El sistema permite identificar posibles riesgos y patrones de violencia antes de que se conviertan en amenazas graves. Una de las características destacadas de este sistema es su capacidad para generar alertas en tiempo real, lo que permite a las autoridades tomar medidas preventivas antes de que ocurran actos violentos(de Justicia y Seguridad Pública, 2025).

Existen muchas más implementaciones tecnológicas como las detalladas anteriormente, lo cual en su mayoría se tratan de soluciones preventivas o reactivas con cierta medida de subjetividad o demora en la detección de actos violentos a nivel mundial. Para tratar de optimizar aquellos problemas, en los últimos años se ha decidido adicionar el uso de inteligencia artificial a las soluciones existentes.

2.4. Detección de violencia a través de IA

Para lograr comprender como es que funcionan las propuestas que han hecho uso de la inteligencia artificial en la actualidad es necesario primero entender como es que funciona esta relativamente nueva tecnología. Para ello, las siguientes secciones se encargan de explicar los principales modelos y algoritmos más utilizados.

2.4.1. *Convolutional Neural Networks (CNN)*

Las CNN son modelos basados en una arquitectura diseñada específicamente para el análisis de imágenes, en particular la clasificación. Estas realizan la tarea de extracción de características a través de convoluciones. Esto evita la pérdida de información y mejora tanto la eficiencia como la precisión.

Las convoluciones se basan en un procedimiento que extrae características mediante la aplicación de una pequeña matriz de transformación cuadrada sobre la imagen original, la cual retorna una imagen modificada. Estas matrices de transformación se denominan kernels. La Figura 2.2 ilustra una iteración de convolución.

Por otro lado, existen otras capas importantes llamadas *poolings*, como se muestra en la Figura 2.3. Estas capas aplican una lógica a un cuadrante de la matriz resultante de las convoluciones. Esta lógica varía dependiendo de las necesidades del usuario. La Figura 2.3 muestra una comparación entre *Max Pooling* y *Average Pooling*, que obtienen respectivamente los valores máximos y promedios de los cuadrantes seleccionados para generar un nuevo resultado con menor dimensionalidad.

Las CNN están compuestas por combinaciones repetidas de capas convolucionales y de *pooling*, finalizando en un conjunto de capas densas de neuronas llamadas *Multi Layer Perceptron*(MLP), que realiza el procesamiento final de las características extraídas. La

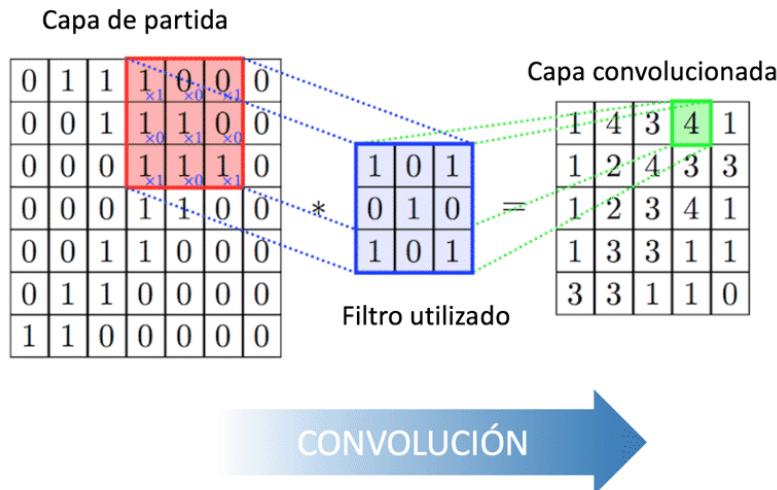


Figura 2.2: Proceso simplificado de una convolución(Diego Calvo, 2019a).

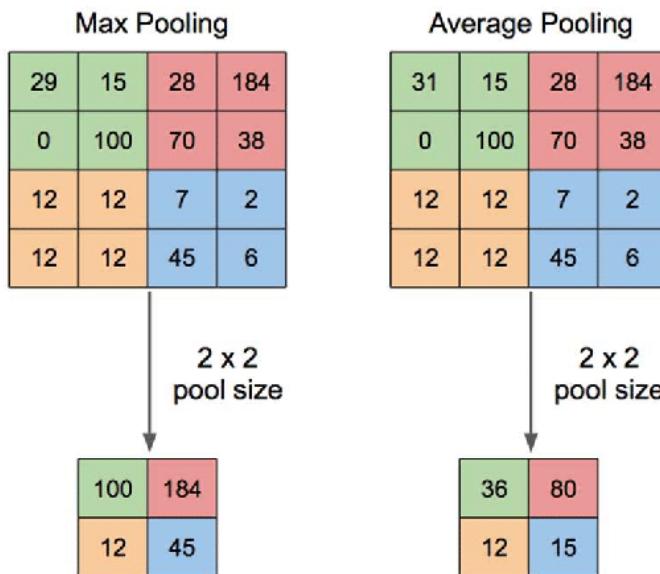


Figura 2.3: Proceso simplificado de un pooling (Muhamad Yani, Budhi Irawan, Casi Se-tianingsih, 2019).

Figura 2.4 muestra la estructura de una CNN. Como se mencionó anteriormente, la extracción de características se realiza dentro de la propia red neuronal, evitando la pérdida de información que se observa en los MLP y dejando únicamente la tarea de clasificación a estos últimos.

A continuación, se explican cada una de las arquitecturas que son utilizadas en el presente trabajo.

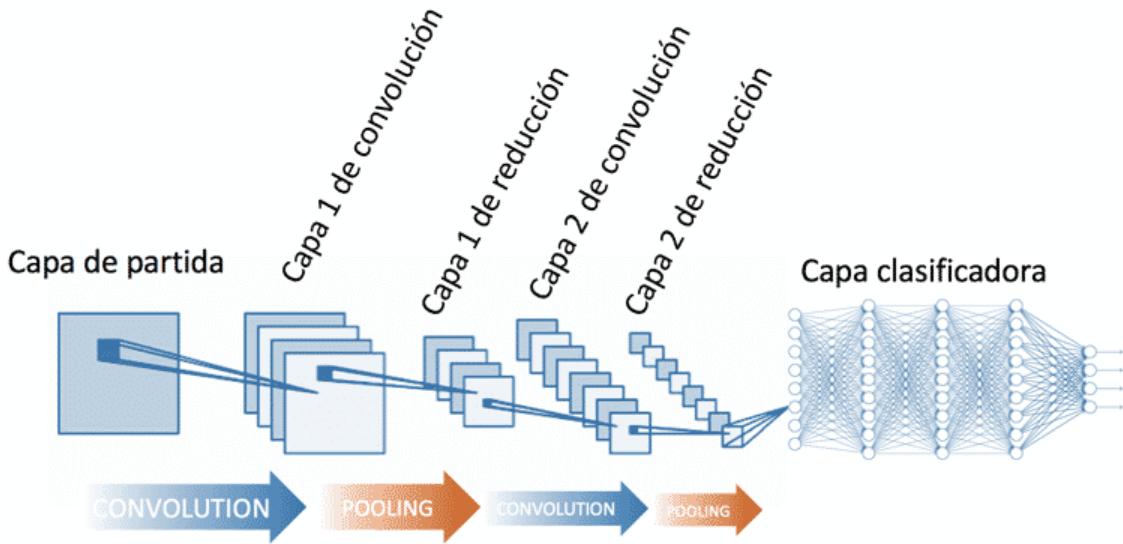


Figura 2.4: Arquitectura de una CNN convencional (Diego Calvo, 2019b).

2.4.2. VGG-19

Esta CNN tiene una profundidad de 19 capas y fue creada por Karen Simonyan y Andrew Zisserman (Simonyan y Zisserman, 2015) en la Universidad de Oxford en 2014, y publicada posteriormente en 2015. Su versión detallada se ilustra en la Figura 2.5. Este modelo fue utilizado para clasificar imágenes en el *dataset* ImageNet, logrando clasificar hasta 1000 objetos diferentes. Espera una imagen de entrada de 224x224 píxeles para su procesamiento. Debido a su profundidad, este modelo es bastante pesado, llegando a consumir hasta 550 MB de memoria con alrededor de 143 millones de parámetros. Cabe destacar que el 70 % de estos parámetros se encuentran entre la última capa convolucional y la primera capa de clasificación. Con todas estas características, logró un 90 % de precisión en ImageNet.

2.4.3. InceptionV3

Aunque VGG19 logró una alta precisión, consumía demasiados recursos. Por esta razón, Google desarrolló InceptionV3, una red que prometía resultados similares pero a un menor costo. Presentado en “*Going deeper with convolutions*” (Szegedy y cols., 2014), este modelo alcanza un 93.7 % de precisión en el mismo *dataset*.

Aunque esta red tiene 50 capas de profundidad (más que VGG19), tiene menos parámetros entrenables (23.8 millones). Su diseño propuso que realizar convoluciones unidimen-

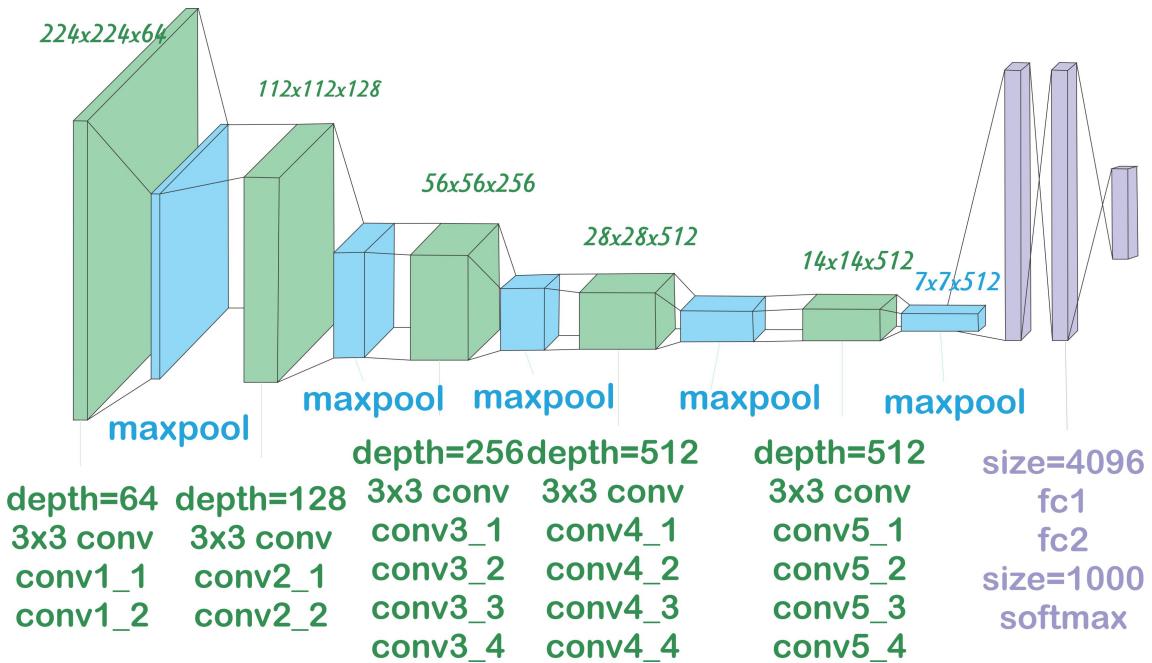


Figura 2.5: Versión simplificada de una VGG19 (IchiPro, 2020).

sionales en serie (como se muestra en la Figura 2.6) es equivalente a una convolución bidimensional, evitando el uso de filtros matriciales. Esto redujo la complejidad de los modelos CNN convencionales, haciéndolo más liviano en memoria y mejorando su capacidad de aprendizaje. En total, este modelo pesa 92 MB—aproximadamente seis veces menos que el anterior. Requiere imágenes de entrada de 299x299 píxeles, y su arquitectura se muestra en la Figura 2.7.

2.4.4. ResNet50

Creado por Microsoft en 2015, este modelo también cuenta con 50 capas de profundidad. Emplea una técnica llamada “residual learning” (He, Zhang, Ren, y Sun, 2015), que consiste en guardar una copia de la salida actual y sumarla al resultado obtenido de un conjunto de convoluciones (típicamente cada tres). La Figura 2.8 ilustra esta modificación y la arquitectura general del modelo. Este modelo también fue probado en el mismo *dataset*, obteniendo una precisión del 92.1 %, y el aprendizaje residual evitó un aumento en la dimensionalidad del modelo. Contiene aproximadamente 25.6 millones de parámetros y ocupa 98 MB de memoria.

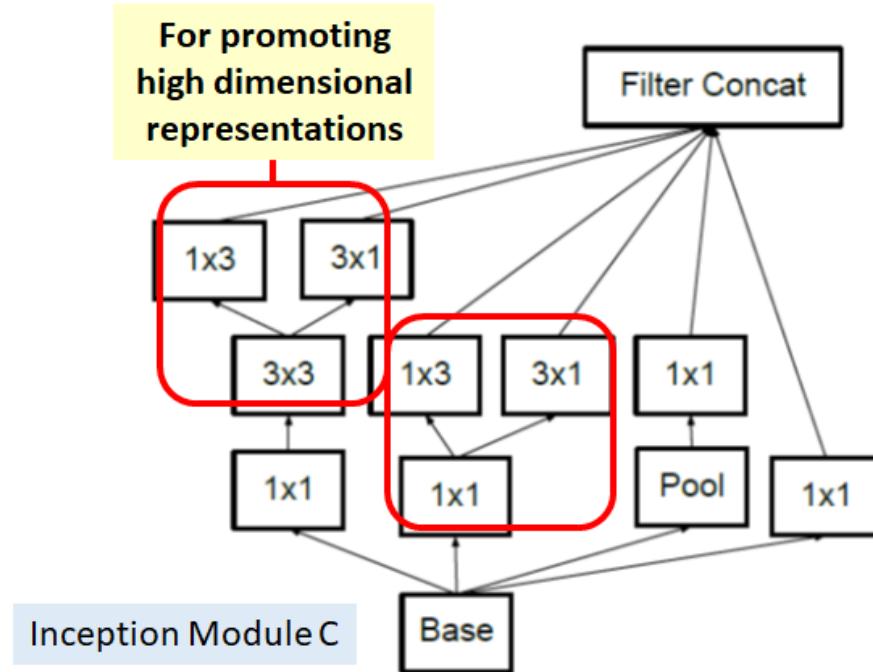


Figura 2.6: Reducción de dimensionalidad de InceptionV3 (IchiPro, 2020).

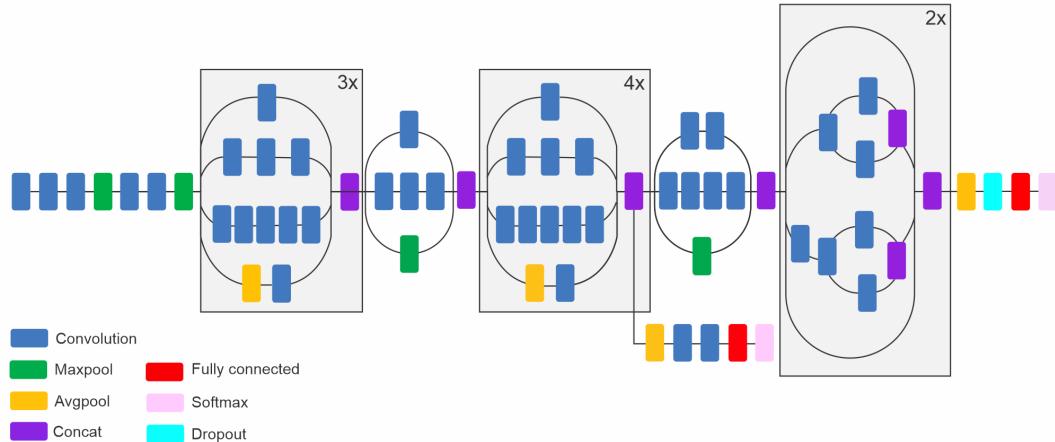


Figura 2.7: Versión simplificada de InceptionV3 (IchiPro, 2020).

2.4.5. EfficientNet

“EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks” (Tan y Le, 2020) en 2020, consiste en 8 implementaciones diferentes (B0 a B7). La versión más ligera (B0) contiene aproximadamente 5.5 millones de parámetros y logró un 93 % de precisión en el mismo *dataset*. Otras versiones continúan aumentando el número de parámetros

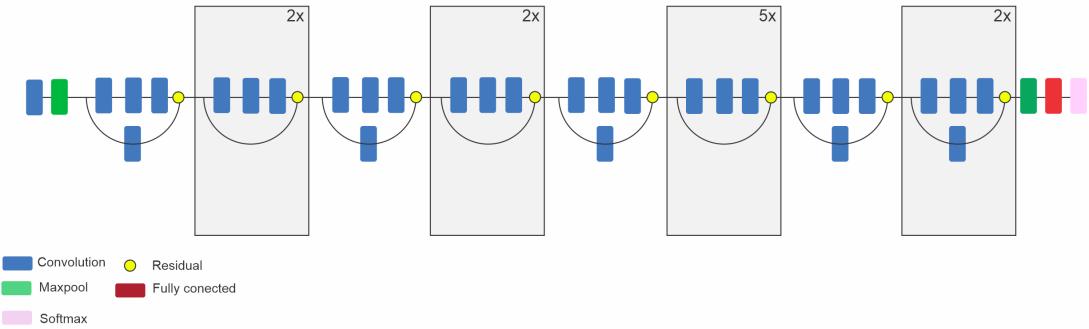


Figura 2.8: Versión simplificada de ResNet50 (IchiPro, 2020).

y la precisión resultante. Comparado con los modelos anteriores, EfficientNetB4 con 19.5 millones de parámetros alcanza un 96.4 % de precisión, superándolos. Este modelo optimiza su aprendizaje mediante un algoritmo de aproximación que genera parámetros para crear cada uno de los ocho modelos. Este algoritmo considera tres factores:

- Profundidad de las capas
- Ancho de las capas (capas múltiples)
- Resolución de la imagen

La Figura 2.9 muestra la estructura de EfficientNetB0.

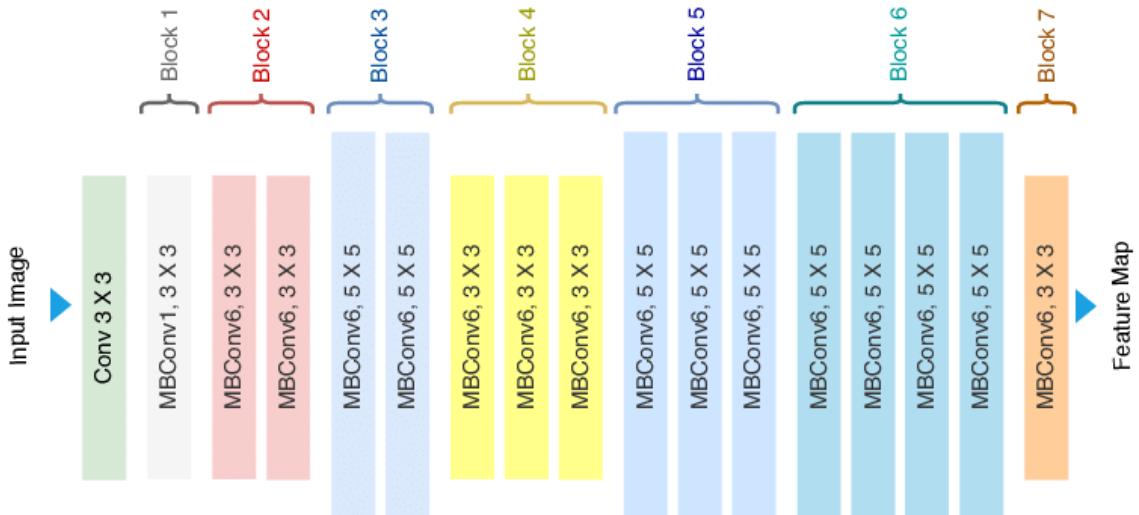


Figura 2.9: Versión simplificada de EfficientNetB0 (Tashin Ahmed, 2020).

2.4.6. YOLO

YOLO (*you only look once*) es una arquitectura que permite la “predicción simultánea de múltiples *bounding boxes* y probabilidades de clase para ellas” (Redmon, Divvala, Girshick, y Farhadi, 2015). Este modelo está basado en una CNN convencional, que, a diferencia de implementaciones anteriores (R-CNN y FR-CNN), realiza la predicción de la caja de enlace internamente, reduciendo la latencia y habilitando su uso en tiempo real.

Este cálculo se basa en generar cuadrículas o particiones de la imagen, dentro de las cuales se inicializa un número predeterminado de *bounding boxes* predefinidas (ambos son hiperparámetros de la arquitectura). Esto es replicable usando cualquier arquitectura CNN como base (por ejemplo, uno de los modelos mencionados previamente), solo ajustando la salida y los hiperparámetros en consecuencia. Las versiones más recientes logran una mejor detección, módulos de atención y otras variaciones que la hacen cada vez más robusta. La Figura 2.10 muestra un ejemplo de la arquitectura del modelo YOLOv5.

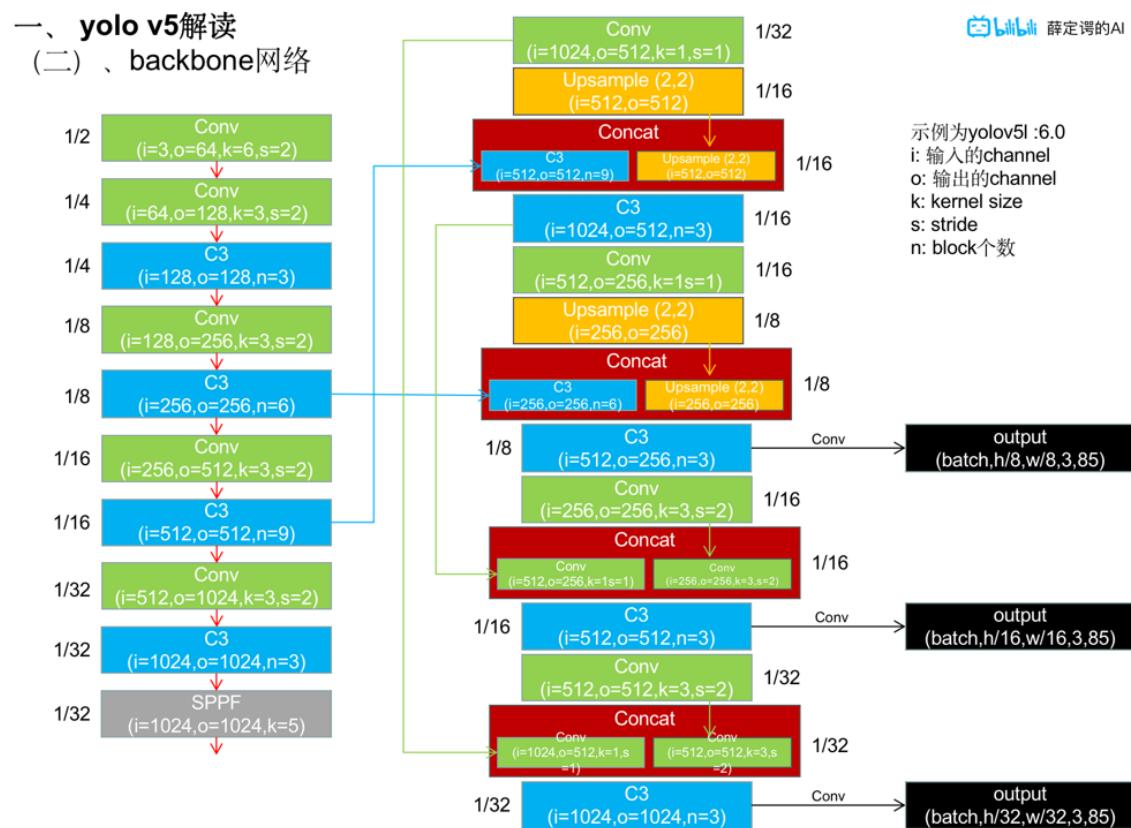


Figura 2.10: Representación de la arquitectura de YOLOv5(Jocher y cols., 2022).

2.4.7. Recurrent Neural Networks (RNN's)

Las redes neuronales recurrentes (RNN, por sus siglas en inglés) son un tipo de arquitectura de red neuronal diseñada para procesar secuencias de datos, como texto, señales de audio o series temporales. A diferencia de las redes neuronales tradicionales (feedforward), las RNNs cuentan con conexiones cíclicas que permiten que la información persista en el tiempo, lo cual es fundamental para modelar dependencias temporales o contextuales. En su forma básica, una RNN toma una entrada secuencial y actualiza un estado oculto interno a cada paso, el cual captura información de entradas previas. Este estado oculto actúa como una "memoria" dinámica que se transmite a través del tiempo, lo que hace posible que la red utilice información pasada para influir en las decisiones presentes(Elman, 1990). La arquitectura simplificada se ilustra en la Imagen 2.11:

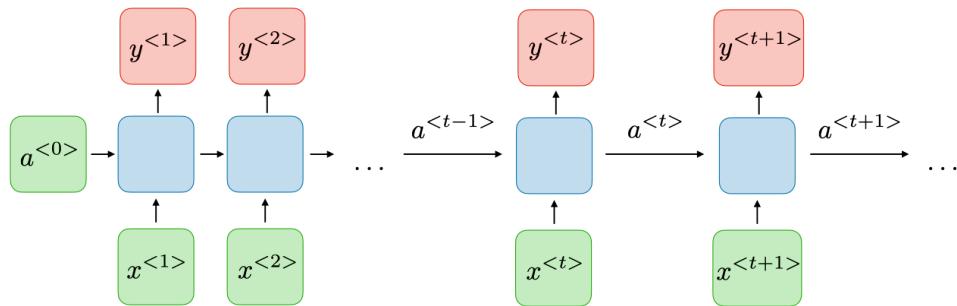


Figura 2.11: Versión simplificada de RNN (Amidi, 2018).

Sin embargo, las RNN tradicionales enfrentan limitaciones prácticas debido al problema del desvanecimiento o explosión del gradiente, lo cual dificulta el aprendizaje de relaciones a largo plazo dentro de la secuencia. A pesar de estas limitaciones, las RNNs han sido fundamentales en el desarrollo de modelos secuenciales y han servido de base para arquitecturas más complejas como LSTM (Long Short-Term Memory) y GRU (Gated Recurrent Units), que fueron diseñadas para mitigar estos problemas. La formulación moderna de las RNNs fue introducida por Elman en 1990(Elman, 1990), quien propuso una red simple que puede aprender representaciones temporales a partir de datos secuenciales mediante el uso de retroalimentación interna en la arquitectura.

2.4.8. Long Short-Term Memory (LSTM)

Las redes *Long Short-Term Memory* (LSTM) fueron desarrolladas como una extensión de las redes neuronales recurrentes (RNN) con el propósito de superar la dificultad

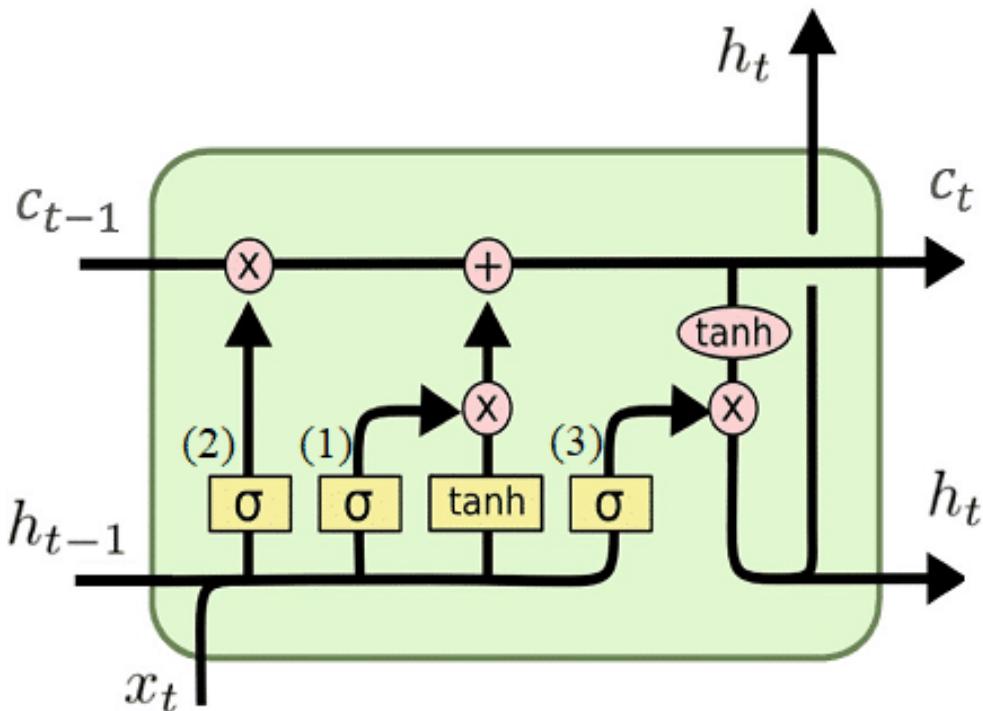
de aprender dependencias a largo plazo en secuencias (Hochreiter y Schmidhuber, 1997). Las RNN tradicionales tienden a sufrir del problema del desvanecimiento o explosión del gradiente durante el proceso de entrenamiento, lo que limita su capacidad para capturar relaciones temporales distantes. Las LSTM abordan esta limitación mediante la incorporación de una memoria interna controlada por compuertas, lo que permite conservar información relevante a lo largo del tiempo.

La arquitectura de las LSTM se basa en una celda de memoria capaz de mantener su estado a lo largo de múltiples pasos temporales. Esta celda está regulada por tres compuertas fundamentales: la compuerta de entrada, que controla qué nueva información se almacena en la celda; la compuerta de olvido, que decide qué información debe eliminarse del estado anterior; y la compuerta de salida, que determina qué parte del contenido de la celda se utiliza como salida. Estas compuertas permiten que la red aprenda a retener o descartar información de manera adaptativa, mejorando significativamente el aprendizaje de secuencias largas.

Gracias a esta estructura, las LSTM han demostrado un rendimiento notable en una amplia gama de tareas secuenciales donde el contexto temporal es esencial. Aplicaciones como el modelado del lenguaje natural, la traducción automática, el reconocimiento de voz y el análisis de series temporales se han beneficiado enormemente de su capacidad para capturar dependencias a largo plazo. En consecuencia, las LSTM se han convertido en una arquitectura fundamental dentro del campo del aprendizaje profundo secuencial. La arquitectura de este modelo se puede ver en la imagen 2.12:

2.4.9. Transformers

Los transformers son una arquitectura de redes neuronales introducida por Vaswani et al. en 2018(Amidi, 2018), originalmente diseñada para tareas de procesamiento de lenguaje natural. Su principal innovación es el mecanismo de autoatención (self-attention), que permite al modelo identificar qué partes de una secuencia son más relevantes entre sí, sin depender del orden secuencial como ocurre en RNNs. Este mecanismo proyecta las entradas en vectores de consulta (Q), clave (K) y valor (V), y calcula sus relaciones mediante un producto punto escalado, permitiendo capturar dependencias contextuales a largo plazo. Además, se introducen codificaciones posicionales para preservar el orden de los datos, y se utiliza atención multi-cabeza (multi-head attention) para que el modelo pueda aprender diversas relaciones de atención de manera paralela. Esta estructura permite un entrena-



LSTM (Long-Short Term Memory)

Figura 2.12: Representación de la arquitectura de LSTM(DataScientest, 2024).

miento más eficiente y una mejor capacidad de generalización. Esto se puede evidenciar en la Imagen 2.13:

Aunque su uso inicial fue en lenguaje natural, los transformers fueron adaptados para clasificación de imágenes mediante el modelo Vision Transformer (ViT)(Dosovitskiy y cols., 2020). En lugar de procesar secuencias de palabras, las imágenes se dividen en parches (por ejemplo, de 16x16 píxeles), que luego son linealmente proyectados a vectores y tratados como si fueran “palabras” en una secuencia. A estos vectores se les suman codificaciones posicionales, y se procesan con la arquitectura transformer para capturar relaciones espaciales entre los distintos parches. Al finalizar las capas de atención, se utiliza un token de clase (similar al [CLS] en NLP) que se conecta a una capa densa para realizar la predicción de clase. Esta aproximación ha demostrado resultados competitivos con arquitecturas CNN tradicionales, y ha abierto nuevas líneas de investigación para aplicar transformers a tareas visuales de manera generalizada.

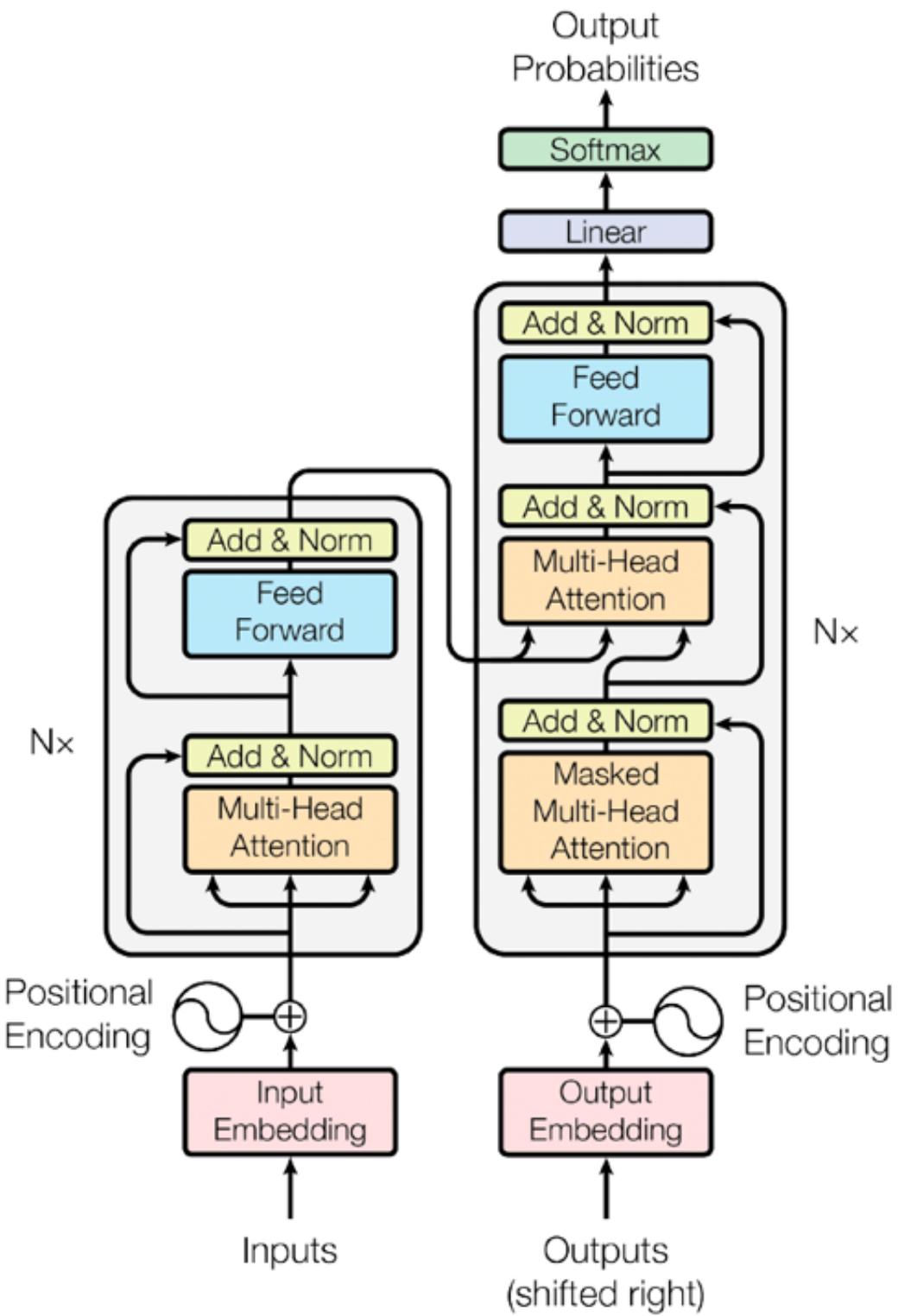


Figura 2.13: Representación de la arquitectura de un transformer(Services, 2024).

2.4.10. Algoritmos basados en la captura esquelética

Los algoritmos basados en esqueletos (*skeleton-based*) para el reconocimiento de acciones humanas utilizan datos tridimensionales de las articulaciones del cuerpo humano para identificar y clasificar actividades(Presti y Cascia, 2016). Estos datos se obtienen mediante sensores de profundidad, como Microsoft Kinect, o a través de técnicas de estimación de pose en imágenes RGB. Una vez capturados, los datos esqueléticos se procesan para modelar la estructura y el movimiento del cuerpo humano, permitiendo a los algoritmos aprender patrones temporales y espaciales asociados a diferentes acciones. Este enfoque ofrece ventajas significativas, como la reducción de la influencia de factores externos como la iluminación o el fondo, y proporciona una representación más abstracta y robusta del movimiento humano.

El reconocimiento de acciones basado en esqueletos ha evolucionado con el tiempo, incorporando diversas arquitecturas de aprendizaje profundo para mejorar su precisión y eficiencia(Ren, Liu, Ding, y Liu, 2024). Entre las técnicas más destacadas se encuentran las redes neuronales recurrentes (RNNs), que capturan la dinámica temporal de las secuencias de movimiento; las redes neuronales convolucionales (CNNs), que extraen características espaciales; y las redes de convolución en grafos (GCNs), que modelan las relaciones estructurales entre las articulaciones del cuerpo. Estas arquitecturas permiten a los sistemas aprender representaciones complejas de las acciones humanas, mejorando su capacidad para reconocer actividades en diversos contextos y condiciones.

2.5. *Transfer Learning*

Según Muhamad Yani (Yani, Irawan, y Setiningsih, 2019), *Transfer Learning (TL)* se define como “el proceso de transferir el conocimiento de un entrenamiento previo para ser utilizado en un nuevo modelo con el fin de reducir el tiempo de aprendizaje”. Este proceso puede ser observado en la Figura 2.14.

TL difiere del proceso de entrenamiento convencional de una red en que no es necesario entrenarla con un gran conjunto de datos. A diferencia del proceso tradicional, las capas iniciales del modelo (en nuestro caso, las capas convolucionales) están congeladas. Estas capas contienen todo el conocimiento pre-aprendido del conjunto de datos con el cual el modelo fue originalmente entrenado. Una vez obtenidas esas capas, se agregan nuevas capas

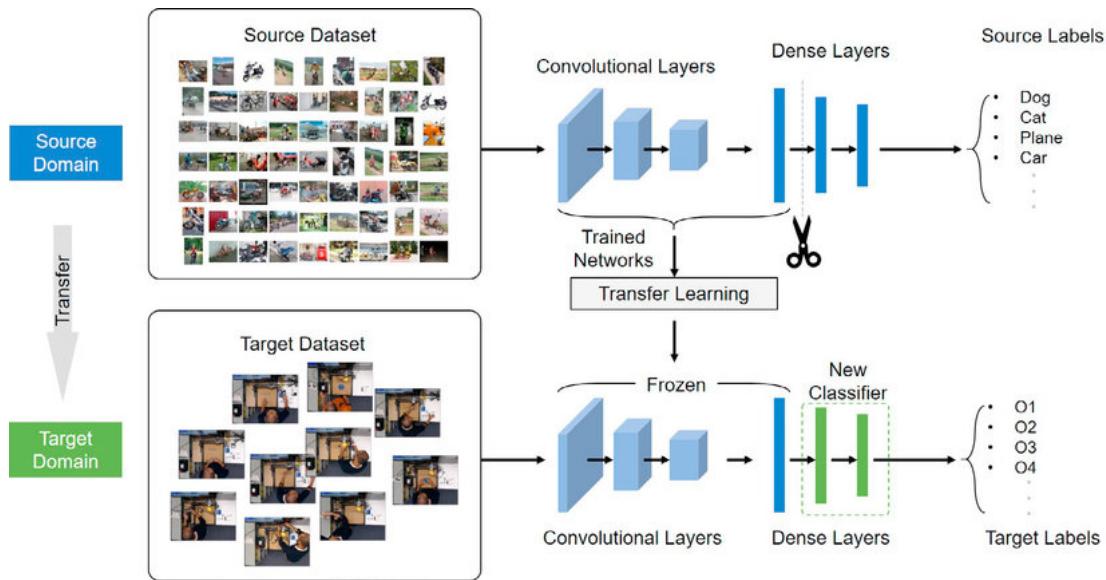


Figura 2.14: Comparativa entre un entrenamiento normal y por TL(Wenjin Taoa, Md Al-Aminb, Haodong Chena, Ming C. Leua, Zhaozheng Yinc, Ruwen Qinb, 2020).

densas (similares a las de un MLP) que sirven como clasificadores para nuestro propósito específico. Gracias a esto, solo las capas finales necesitan ser entrenadas, lo que requiere menos datos y generalmente resulta en predicciones precisas. Este tipo de entrenamiento se llama “*fine-tuning*”, lo que permite que la red ajuste el aprendizaje previo para predecir en función de un conjunto de datos diferente al que fue entrenada originalmente, ahorrando tiempo y mejorando la precisión.

2.5.1. *Pipelines* actuales para la clasificación de violencia

La literatura actual muestra un creciente interés en la detección automática de violencia en videos mediante la combinación de redes convolucionales (CNN) y redes de memoria a largo plazo (LSTM)(Sudhakaran y Lanz, 2017a). Las CNN se utilizan para extraer características espaciales relevantes de los fotogramas, permitiendo identificar patrones visuales y contextos que podrían indicar situaciones violentas. Posteriormente, estas características son procesadas por las LSTM, las cuales se encargan de capturar la dinámica temporal y las dependencias de largo plazo entre los fotogramas, mejorando la capacidad de detectar eventos violentos que se desarrollan a lo largo del tiempo.

En diversos estudios se ha demostrado que la integración de ambas arquitecturas permite explotar de forma sinérgica las ventajas de cada una(Mohammadi y Yen, 2022): las

CNN fortalecen la comprensión del contenido visual a nivel de detalle y textura, mientras que las LSTM aportan una perspectiva temporal que es crucial para identificar patrones de comportamiento complejos y secuenciales propios de la violencia. Este tipo de *pipeline* ha sido evaluado en diferentes escenarios, incluyendo videos de vigilancia y grabaciones deportivas, logrando así mejores tasas de detección en comparación con métodos que utilizan únicamente análisis espacial o temporal de manera aislada.

Un claro ejemplo de lo mencionado anteriormente, es el artículo de Jeff Donahue, *et al.* (Donahue y cols., 2014). Su investigación se centra en el desafío de trabajar con datos visuales que contienen información espacial y temporal. La idea central es integrar ambas arquitecturas en un único sistema end-to-end: las CNNs se encargan de extraer representaciones ricas de contenido visual, y las LSTMs modelan la dinámica secuencial, permitiendo aplicaciones en reconocimiento de actividades en video y generación de descripciones de imágenes.

Entre sus principales contribuciones, este trabajo logró sentar las bases las bases en la integración de arquitecturas visuales y secuenciales. Su enfoque ha influido en numerosos trabajos posteriores en áreas como la descripción automática de videos, el análisis de secuencias complejas y el desarrollo de modelos más integrados para tareas multimodales.

Utilizando esta lógica, el trabajo de Orozco, *et al.* (Orozco y cols., 2021) el cual destaca la efectividad de estas estrategias al combinar etapas de preprocesamiento, extracción de características espaciales mediante CNN, y análisis secuencial temporal con LSTM. Asimismo, se discuten los desafíos asociados a la variabilidad de escenarios y la detección en tiempo real, aspectos que continúan motivando la investigación y optimización de estos sistemas. Como resultado final de su experimentación, los autores lograron un F1-score de 91 % lo cual indica que fue *pipeline* robusto el problema que intentó resolver basado en la clasificación de acciones humanas basado en videos. Los pipelines híbridos basados en CNN y LSTM representan una tendencia robusta en el campo de la detección de violencia, contribuyendo significativamente a la mejora en la precisión y eficiencia de los sistemas de análisis de video.

Otro artículo que también utiliza esta misma lógica es el de Swathikiran Sudhakaran

and Oswald Lanz(Sudhakaran y Lanz, 2017b). Su objetivo consistió en el mismo del presente trabajo, la detección de violencia. Para ello, se utilizó el *dataset* de Hockey Fights, el cuál contiene videos de eventos violentos en partidos de hockey. Este *dataset* es bastante usado como *benchmark* y consiste de un conjunto de imágenes etiquetadas de manera homogénea. La variación más significativa que realizaron fue el uso de celdas convLSTM en vez de LSTM regulares para poder obtener un *accuracy* más elevado. Con su *pipeline* lograron obtener una *accuracy* de 97 %, generando un nuevo record para este *benchmark* y marcando un nuevo estado del arte.

De la misma manera y utilizando el mismo *dataset* de Hockey Fights, se encuentra el trabajo de Al-Maamoon R. Abdali y Rana F. Al-Tuma(Abdali y Al-Tuma, 2019). Ellos se basaron en el trabajo anteriormente mencionado para su implementación. Para ello, utilizaron la misma configuración del *pipeline* pero incluyeron capas conv3d y 40 celdas para el aprendizaje de la LSTM sin ser convolucionales. Utilizando aquella configuración logrando obtener un *accuracy* de 96.33 % pero al mismo tiempo obteniendo una mejora de 4 veces en la velocidad (representado en el número de fps), representando una mejora del estado del arte en su tiempo al mantener el mismo nivel de *accuracy* pero mejorando el *performance*.

El último artículo a revisar es el de Patel Mann (Mann, 2021). En su trabajo utilizó Resnet50, InceptionV3 y VGG19 como extractores de características. En cambio, utilizaron solo una celda LSTM para la clasificación y como resultados obtuvieron 90 % de precisión para el dataset de Hockey fights, representando una disminución a comparación de los anteriores trabajos, aunque permitió optimizar el uso de memoria sin perder drásticamente el *performance* de todo el *pipeline*.

Como se puede ver, el *pipeline* CNN-LSTM propuesto ha sido utilizado a lo largo de los últimos años para resolver problemas similares e iguales al nuestro. En ese sentido nos hace sentido tratar de extender la aplicación de este y optimizarlo para tratar de ver como es que diferentes configuraciones del mismo permiten mejorar ya sea el *performance* o el *accuracy* de la propuesta.

3. Objetivos

3.1. Objetivos

En este capítulo se presentan los objetivos y contribuciones de esta tesis. Una comprensión clara de estos aspectos es esencial para contextualizar el alcance y la relevancia de esta investigación. Las siguientes secciones ofrecerán una discusión detallada de los objetivos clave perseguidos en este trabajo, así como de las contribuciones que se pretende aportar al campo.

El objetivo principal de esta tesis es el diseño e implementación de un *pipeline* que combina el uso de un extractor de características basado en CNN's junto con capas Bi-LSTM para la detección y clasificación de escenas violentas. Junto con ello, se planea optimizar la precisión y la velocidad del mismo haciendo múltiples experimentos modificando el clasificador y el número de celdas BI-LSTM con el fin de construir el pipeline más efectivo para la detección automática de violencia en videos. Este objetivo surge de la necesidad de comprender cómo interactúan estos dos componentes fundamentales del aprendizaje profundo en tareas espaciotemporales complejas, como lo es la detección de violencia, y cómo su configuración afecta tanto la precisión como la eficiencia del sistema.

Adicionalmente, se plantean varios objetivos secundarios. En primer lugar, se busca realizar un preprocesado de datos que permita al *pipeline* ser entrenado con ellos. En segundo lugar, se busca evaluar el impacto de diferentes arquitecturas CNN en la calidad de las características espaciotemporales extraídas, entendiendo cómo influyen estas variaciones en el rendimiento del modelo. Por último, se pretende investigar cómo la variación en el número de celdas LSTM afecta la modelación temporal y, por ende, la eficacia en la clasificación de eventos violentos.

3.2. Contribuciones

La contribución principal de esta tesis es el desarrollo de un pipeline optimizado para la detección de violencia que equilibra la complejidad de la extracción de características basada en CNN y el número de celdas LSTM, con el objetivo de lograr una mayor precisión y eficiencia. Al analizar sistemáticamente los compromisos entre estos dos componentes, este trabajo proporciona un enfoque estructurado para el diseño de arquitecturas de aprendiza-

je profundo orientadas al reconocimiento espaciotemporal de patrones violentos en video.

Este enfoque contribuye significativamente al avance del análisis de video asistido por inteligencia artificial para aplicaciones de vigilancia en tiempo real. A través del diseño y evaluación de múltiples configuraciones arquitectónicas, se demuestra cómo el ajuste preciso de los módulos CNN y LSTM puede resultar en modelos más efectivos y menos costosos computacionalmente. Esta tesis, por tanto, ofrece una guía práctica para investigadores y desarrolladores interesados en implementar soluciones robustas y escalables en escenarios del mundo real donde la detección temprana de violencia es crítica.

4. Metodología

En el presente capítulo se expone los métodos utilizados para conseguir los objetivos específicos detallados en el Capítulo 3.2. Cada subsección corresponde con cada uno de los objetivos en los que se explicará en qué consiste cada objetivo de forma más extendida, la herramienta escogida para su realización y la justificación de su elección.

4.1. Descripción del *Dataset*

Para la realización de este trabajo se decidió elegir el “*Hockey Fight Detection Dataset*” el cual es un conjunto de datos ampliamente utilizado en la investigación sobre detección automática de violencia en videos. Fue desarrollado por Enrique Bermejo Nievas, Óscar Deniz Suárez, Gloria Bueno García y Rahul Sukthankar, y publicado en 2011 (Nievas, Suárez, García, y Sukthankar, 2010). Este dataset fue concebido para abordar la necesidad de sistemas capaces de identificar comportamientos agresivos en entornos de vigilancia, como prisiones, centros psiquiátricos o residencias de ancianos, donde la detección temprana de violencia es crucial.

El conjunto de datos consta de 1,000 secuencias de video, divididas equitativamente en dos categorías: 500 videos que muestran peleas durante partidos de hockey sobre hielo y 500 videos sin violencia. Cada video tiene una duración aproximada de 1 a 2 segundos y está etiquetado manualmente para facilitar su uso en tareas de clasificación supervisada. Los videos fueron recopilados de diversas fuentes, incluidos sitios web de noticias y plataformas de video en línea, asegurando una variedad de escenarios y condiciones de iluminación. En la figura 4.1 se muestra una comparación de un frame de cada uno de las etiquetas:

4.2. Pre-procesamiento de la base de datos

Para preparar los datos del conjunto *Jockey Fights* y garantizar su compatibilidad con los modelos de aprendizaje profundo utilizados, es necesario el siguiente proceso de pre-procesamiento estructurado en varias etapas:

- **Obtención de frames:** Cada video del dataset se descompone en sus respectivos fotogramas (frames) utilizando una frecuencia de muestreo fija. Estos frames fueron almacenados como matrices RGB, representando la información visual en tres canales



(a) Frame conteniendo violencia

(b) Frame no conteniendo violencia

Figura 4.1: Comparación entre frames con y sin violencia en el “Hockey Fight Detection Dataset”

(rojo, verde y azul), lo cual permitió capturar el contenido visual de cada instante del video.

- **Recorte y redimensionamiento:** Los frames extraídos se redimensionaron a una resolución fija (como 224x224 píxeles), adaptándose así a los requerimientos de entrada de los modelos de red neuronal convolucional (CNN) utilizados en la etapa de extracción de características.
- **Normalización:** Con el objetivo de mejorar la eficiencia del entrenamiento de los modelos, se normalizaron los valores de los píxeles de las imágenes. Esto se realizó escalando los valores del rango [0, 255] al rango [0, 1] o aplicando una normalización estadística basada en la media y desviación estándar de los canales RGB, especialmente útil cuando se emplean modelos preentrenados.
- **Etiquetado de los datos:** A partir de los nombres de los videos y su contexto, se asignaron etiquetas binarias a cada secuencia de imágenes. Los videos fueron clasificados como *violentos* o *no violentos*, y dicha etiqueta fue propagada a todos los frames correspondientes al video, asumiendo homogeneidad del contenido en cada segmento.
- **Filtrado y muestreo:** Para reducir la redundancia y el volumen de datos, se aplicó una estrategia de muestreo temporal, extrayendo únicamente un subconjunto de los frames. Para ello, se propuso un muestreo con reemplazo en donde se seteó un número final de frames y se procedió a obtener un frame cada x de los videos. Se utilizó este algoritmo para manejar el caso en el que el número de frames no sea el mismo no solo para el *dataset* actual, sino para que sea utilizado en otros ejemplares.

Este pre-procesamiento es esencial para estandarizar los datos, minimizar el ruido y facilitar el posterior entrenamiento de los modelos de detección de violencia basados en aprendizaje profundo.

4.3. Métricas de Evaluación

Para la evaluación del conjunto de datos *Jockey Fights*, se utilizan las siguientes métricas de rendimiento:

- **F1-score:** penaliza fuertemente los errores y obliga a capturar los positivos y no tener demasiados falsos positivos.

$$F1 \text{ Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4.1)$$

- **Tiempo de Inferencia:** Mide el tiempo que el modelo tarda en procesar una secuencia completa o un solo frame. Es crucial para aplicaciones en tiempo real o streaming.

4.4. Software y hardware utilizados

A lo largo de toda la experimentación, se utilizará un computador de escritorio con las siguientes características:

Componente	Descripción
Procesador	Procesador Core i5-10300H 2.50GHz
Memoria RAM	16GB DDR4 - 3600MHZ
Tarjeta Gráfica	Nvidia RTX 3050 4GB

Por otra parte, se utilizó Python, junto con Keras, Sklearn y Tensorflow para el desarrollo del *pipeline* en general y el pre y post procesamiento de los datos. Para el procesamiento de las imágenes se utilizó OpenCV compilado con cudnn para la habilitación del uso de tarjeta gráfica.

4.4.1. Evaluación de CNN's para la experimentación

En 2020, Orhan Yalsin (Yalcin, 2020) realizó una resumen con algunas métricas para la evaluación de diferentes modelos de vanguardia, los cuales son mostrados en la Tabla ??.

Nos basamos en sus resultados para seleccionar aquellos modelos que son mas eficientes tanto en tiempo, espacio y precisión.

Esta tabla destaca varios modelos notables por su equilibrio entre tamaño, cantidad de parámetros y precisión. Entre ellos VGG19 (si consideramos solo las capas convolucionales), ResNet50, InceptionV3, y los EfficientNetBX resultan tener una buena relación entre el peso, el número de parámetros y la precisión que estas obtienen, esta última siendo de las más altas dentro de todos estos modelos. Se tuvo en consideración la ejecución en videos en tiempo real y por ello se necesitó que el modelo sea el más pequeño posible evitando perder la precisión. Por otra parte, están las MobileNet, las cuales ya han sido utilizadas para el procesamiento de imágenes en tiempo real por ser más ligeras, enfocadas a ser utilizadas en celulares. Creemos que todas estas redes eran aptas para este problema y pasarán a ser evaluadas en el siguiente capítulo, en base al primer *dataset* mencionado anteriormente.

Por otra parte, en un trabajo previo de mi autoría, titulado “Machine Learning Pipeline Para el Etiquetado Automático en Imágenes de Especies de Peces Peruanos” (Madera y López, 2024), se experimentó con todas las redes anteriormente mencionadas para la tarea de clasificación de imágenes de peces peruanos. La tabla 4.2 referencia los resultados obtenidos en aquella experimentación:

En la Tabla 4.2 se puede ver como los mejores resultados fueron obtenidos por EfficientNetB0 y Resnet50 tanto en precisión, velocidad por batch y épocas de entrenamiento, por lo que se utilizarán para el presente trabajo. Se utilizará MobilenetV3 como modelo control y además, se utilizará otro modelo EfficientnetV2, el cual es un modelo un poco más pesado pero que brinda una gran mejora en la precisión general.

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159
Inception ResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-
EfficientNetB0	29 MB	0.771	0.933	5,330,571	-
EfficientNetB1	31 MB	0.791	0.944	7,856,239	-
EfficientNetB2	36 MB	0.801	0.949	9,177,569	-
EfficientNetB3	48 MB	0.816	0.957	12,320,535	-
EfficientNetB4	75 MB	0.829	0.964	19,466,823	-
EfficientNetB5	118 MB	0.836	0.967	30,562,527	-
EfficientNetB6	166 MB	0.840	0.968	43,265,143	-
EfficientNetB7	256 MB	0.843	0.970	66,658,687	-

Tabla 4.1: Evaluación de diferentes modelos. Tabla brindada por Orhan Yalcin (Yalcin, 2020).

	Precisión en training	Precisión en validation	Precisión en testing	Perdida final en training	Velocidad por batch (8 imágenes) en ms	Número de épocas
Vgg19	0.9193	0.9928	0.9926	0.2299	71	21
Resnet50	0.9596	0.9993	0.9950	0.1282	52	14
Inception V3	0.7362	0.6620	0.6681	0.6901	42	40
MobileNet V2	0.8521	0.9052	0.9104	0.4143	28	31
MobileNet V1	0.8694	0.9288	0.9519	0.3628	23	23
Xception	0.7244	0.7085	0.7178	0.7490	48	25
EfficientNet B0	0.9577	0.9993	1.0000	0.1270	41	19

Tabla 4.2: Datos obtenidos de la experimentación previa con dataset de peces

5. Desarrollo del trabajo

En el presente capítulo se explica la experimentación realizada para la clasificación de los videos del *dataset Hockey Fights*. La estructura que se detalla a continuación consiste de: explicación del *pipeline*, la experimentación con las diferentes configuraciones de BI-LSTM y la comparativa de los resultados.

5.1. Pipeline propuesto

La Imagen 5.1 ilustra la metodología propuesta, la cual se procederá a explicar a continuación:

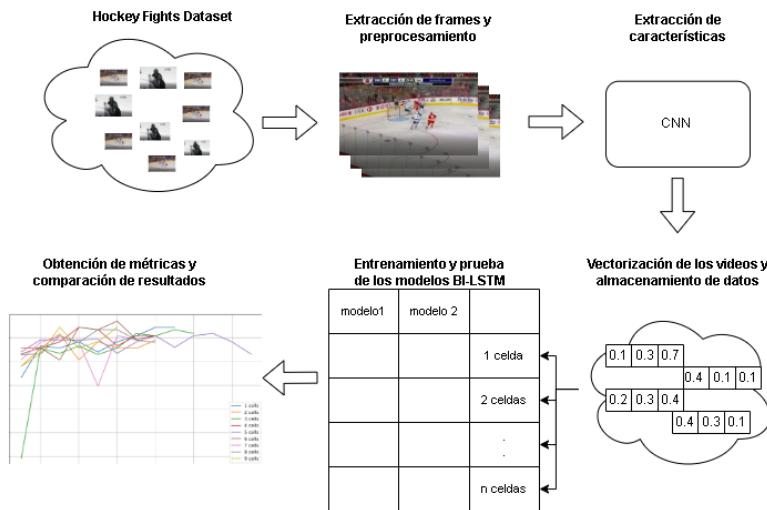


Figura 5.1: Metodología propuesta. Creación propia.

El *pipeline* consiste en la obtención, preprocesamiento y almacenamiento de las características obtenidas del *dataset*, para luego ser pasado a las diferentes configuraciones de los modelos BI-LSTM, los cuales se encargarán cada video basado en sus frames y finalmente se evaluarán los resultados basados en la métrica f1 y el tiempo de inferencia.

5.1.1. Extracción de frames y preprocesamiento

Como primer paso de la metodología se necesitó realizar un preprocesamiento de los datos para su análisis posterior a través de los modelos mencionados en el anterior capítulo. Como primer paso, se procedió a descargar el *dataset* a través de su repositorio de Kaggle.

Una vez obtenidos los datos, se procedió a etiquetar cada uno de los videos basados en los nombres de cada uno de ellos. Se puede diferenciar su etiqueta debido a que aquellos que comienzan con "fi" contienen violencia, mientras que los demás, no. Con los videos cargados, se recolectaron los frames tal como fue mencionado anteriormente. Para ello, se decidió hacer un muestreo de 10 frames de manera equitativamente separada del video (1 por cada 4 frames). Cada uno de estos frames fueron redimensionados a un formato estandar (224 x 224) para luego ser normalizados.

5.1.2. Obtención de las características

Con cada conjunto de frames por video, estos pasan a través de las diferentes arquitecturas de CNN mencionadas en el anterior capítulo. Para ello, se tuvieron las siguientes consideraciones:

- Se obtendrán los modelos pre-entrenados con el dataset ImageNet.
- Se eliminan las capas de clasificación.
- Se agrega una capa GlobalAveragePooling2D para reducir la dimensionalidad de kernels(resultado de las CNN).
- Se marca al modelo final como no entrenable.

Este ultimo paso es necesario debido a que se utilizará el conocimiento pre-entrenado de los modelos para obtener únicamente el vector característico de cada una de las imágenes. En ese sentido, se hace innecesario el proceso de entrenamiento de estas. Además, sería contraproducente tanto entrenar una CNN basado únicamente en frames porque pueden existir casos en el que existen frames donde no hay violencia y viceversa y también debido a que se perdería parte del conocimiento que ya tienen estos modelos. Finalmente se procedió a guardar estos vectores de características en formato h5.

Cabe resaltar que en la realidad podría implementarse un modelo conjunto con la CNN y LSTM, pero algunos de los modelos (especialmente los EfficientNet) contienen capas que no son fácilmente mapeables a las entradas de las BI-LSTM, por lo cual se tuvo que hacer esto como preprocesamiento.

5.1.3. Entrenamiento y prueba del clasificador

Como clasificador, se tuvo un modelo de BI-LSTM con diferente número de celdas. Para esta experimentación, se decidió usar un número entre 1 y 9 celdas, ya que consistía en un dataset bastante sencillo y altamente estudiado anteriormente.

Para cada uno de las CNN's mencionadas en la anterior sección, y para cada una de las configuraciones, se procedió a entrenar y testear cada combinación de ellas. Las consideraciones para esta arquitectura fueron las siguientes:

- De entrada se tienen los vectores de características obtenidos del paso anterior (se debe considerar que los tamaños de salida pueden variar con respecto a la CNN).
- Se crea un modelo BI-LSTM con X celdas, 10 timesteps (en otras palabras los 10 frames obtenidos) y el tamaño del output del anterior paso.
- Capas densas de 1024, 50 y 2 neuronas con funciones de activación ReLU y la última de tipo softmax.
- Función de pérdida Binary CrossEntropy, optimizador adam y de métricas accuracy y f1-score.
- Función de early stopping con *patience* de 3 y *min delta* de 0.005 y batch de 16 videos por vez.

En la siguiente sección se presentarán los resultados generados utilizando las anteriores consideraciones para cada uno de los modelos mencionados.

5.2. Experimentación

Como primer paso, se procedió a realizar el pre-procesamiento de las imágenes y la obtención de sus características tal como fue mencionado anteriormente. Para ello, se recolectaron las métricas acerca de la cantidad de parámetros y el tiempo de cálculo de cada uno de los modelos, los cuales se pueden observar en la Tabla ??:

Se puede apreciar que el tamaño final de los modelos fue menor que el esperado por la Tabla 4.1. Esto se debió a que no se están utilizando las capas densas de los modelos y se están reemplazando con una capa GlobalAveragePooling2D, la cual obtiene el resultado del kernel final de las capas convolucionales y las reduce a un solo resultado promediado.

Modelo	Parámetros	Tímpo de inferencia (por batch de 10) en ms/batch
efficientnetb0	4,049,571	46 ms
resnet50	23,587,712	51 ms
efficientnetv2-s	20,331,360	75 ms
MobilenetV3large	2,996,352	45 ms

Habiendo obtenido las características de cada uno de las imágenes y los diez frames generados por cada uno de ellas, se entrenaron diferentes modelos BI-LSTM como fue mencionado anteriormente, teniendo desde 1 a 10 celdas para cada uno de los CNN utilizados. A continuación se muestran las gráficas de pérdida y f1 score para cada una de los experimentos.

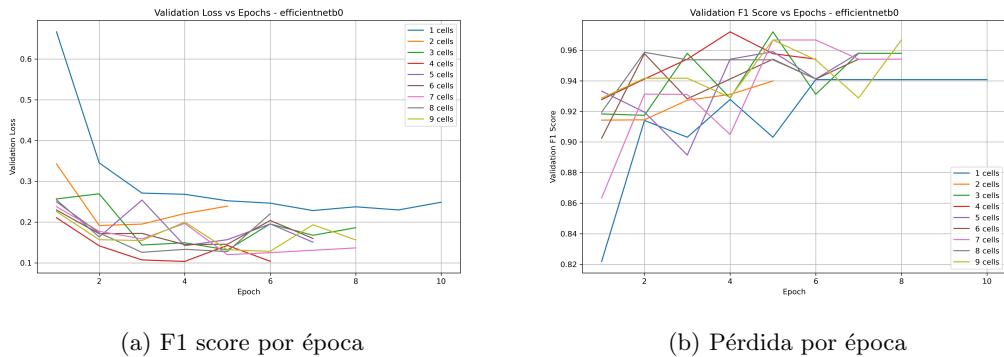


Figura 5.2: Gráficos de pérdida y f1 score por época usando EfficientNetB0

Los gráficos de la figura 5.2 muestran un descenso de la pérdida generada de manera estable, en donde el clasificador con 1 capa se puede apreciar muy separado de las demás configuraciones. Fuera de ella, todas las demás configuraciones obtuvieron resultados muy similares. Sin embargo, se puede apreciar que la configuración con 4 celdas fue la que mejor resultado obtuvo en esa métrica. Algo muy similar se ve en la gráfica de f1 score. La configuración con una y dos celdas generaron peores resultados, mientras que las demás obtuvieron mejores resultados. La diferencia radica en que si consideramos el resultado final después de 8 épocas, la configuración con 9 celdas es la ganadora. Pero si se considera que el early stopping tenía una paciencia de 3 épocas, tendríamos que considerar el resultado en la época 6, en donde la configuración con 7 celdas fue la que generó mejores resultados.

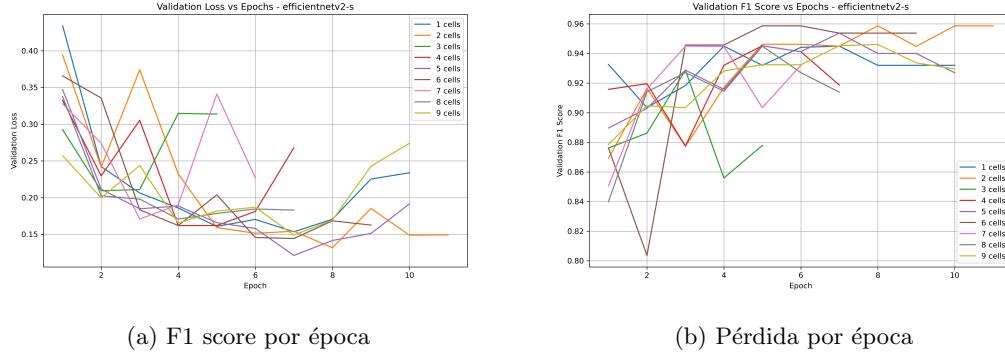


Figura 5.3: Gráficos de pérdida y f1 score por época usando EfficinetnetV2-S

A comparación de la anterior ejecución, los gráficos de la figura 5.3 muestran un descenso de la pérdida gerada menos estable. Los resultados no muestran un patrón único, por lo que la única conclusión basada en los datos es que los mejores resultados se obtuvieron con 2 y 5 celdas. Por otra parte, en términos de f1 score se pudo apreciar un mejor entrenamiento y resultado final en la configuración de 2 celdas.

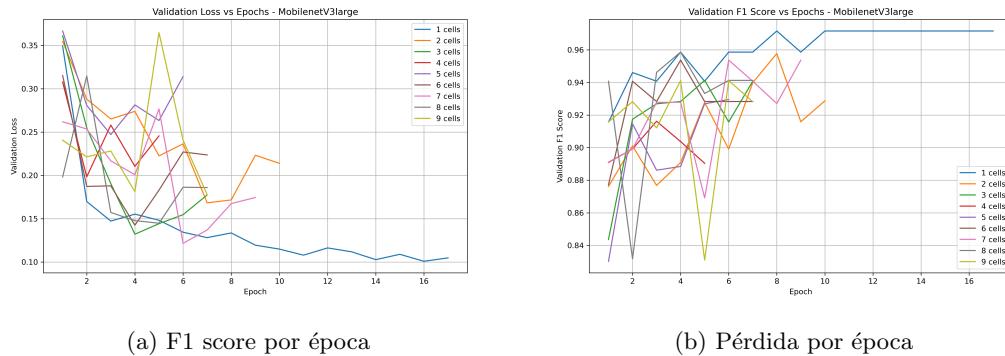


Figura 5.4: Gráficos de pérdida y f1 score por época usando MobileNetV3

Utilizando MobileNetV3 se obtuvo un patrón muy distintivo entre la gráfica de pérdida y f1 score. En ambos casos se tuvo un descenso de pérdida muy estables utilizando la configuración de una sola celda, mientras que las demás se comportaron muy erráticamente. Esto es probablemente debido a la configuración del modelo, ya que es uno de los más ligeros del estado del arte (y también de entre los utilizados para esta experimentación). En ese sentido, las configuraciones con múltiples celdas cayeron en un overfitting para los vectores de características salidos de la CNN, por ende generando valores negativos en ambas métricas.

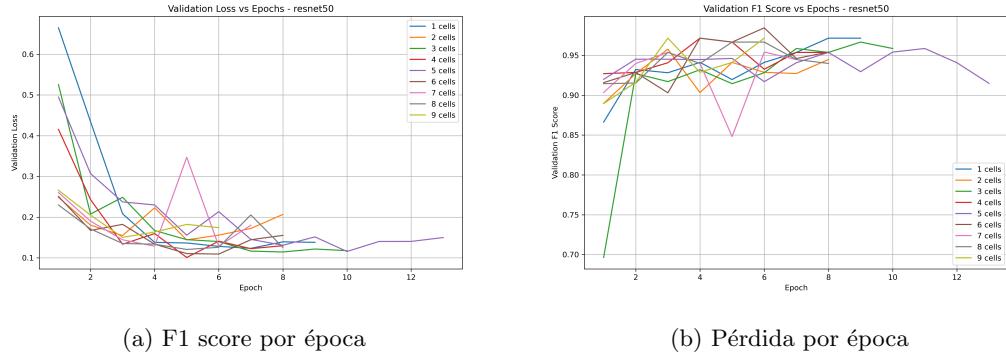


Figura 5.5: Gráficos de pérdida y f1 score por época usando Resnet50

Las ejecuciones de Resnet50 generaron patrones más estables en ambos gráficos. En ese sentido la mayoría tuvo un descenso de la pérdida bastante similar entre ellos. Al compararlos teniendo en cuenta el early stopping, en la época 6 se generó una menor pérdida con el modelo de 6 épocas, lo cuál también concordó con ser el modelo con mejor f1 score de entre todos. Un dato interesante a analizar es el de la configuración con 7 celdas, el cual generó picos erráticos en el entrenamiento, lo cual también se reflejó en el gráfico de f1 score. Además, la configuración de 5 celdas tuvo una mayor cantidad de épocas de entrenamiento a comparación de los demás.

A continuación se muestran las métricas finales por cada uno de los modelos:

lstm_cells	train_loss	train_f1	val_loss	val_f1	test_loss	test_f1
1	0.0347	0.9873	0.2488	0.9408	0.0974	0.9725
2	0.1213	0.9564	0.2392	0.9399	0.1674	0.9525
3	0.0209	0.9902	0.1862	0.958	0.0698	0.9825
4	0.047	0.9929	0.1038	0.9541	0.1133	0.9575
5	0.058	0.979	0.1506	0.9581	0.1049	0.97
6	0.029	0.9931	0.1599	0.9541	0.0919	0.9725
7	0.0258	0.9933	0.1367	0.9541	0.0691	0.985
8	0.0214	0.9929	0.2205	0.9412	0.1213	0.9625
9	0.012	0.9961	0.1564	0.9667	0.0605	0.98

Tabla 5.1: Tabla comparativa de las métricas obtenidas por configuración de celdas para EfficientNetB0.

En la tabla 5.1 se tienen métricas de pérdida y f1 score para todas las etapas: entrenamiento, validación y testeо para cada una de las configuraciones. En ella se puede ver que los mejores resultados de manera casi general fue la configuración de 9 celdas, lo cual no se refleja con los resultados de validación que se vieron en los gráficos de pérdida y f1 score anteriormente mostrados.

lstm_cells	train_loss	train_f1	val_loss	val_f1	test_loss	test_f1
1	0.0833	0.9693	0.2338	0.932	0.12	0.9675
2	0.048	0.9869	0.1492	0.9586	0.0849	0.9775
3	0.185	0.9349	0.3138	0.8779	0.2181	0.9175
4	0.1111	0.9605	0.268	0.9187	0.1375	0.96
5	0.0228	0.9936	0.1915	0.927	0.0666	0.9825
6	0.0396	0.9875	0.1626	0.9537	0.0789	0.98
7	0.1473	0.9402	0.2269	0.932	0.1458	0.96
8	0.0504	0.9872	0.1829	0.9139	0.1408	0.95
9	0.018	0.9969	0.2741	0.9295	0.0649	0.9825

Tabla 5.2: Tabla comparativa de las métricas obtenidas por configuración de celdas para EfficientnetV2.

A comparación de la anterior tabla, utilizando los datos de la tabla 5.2 no se puede sacar una conclusión específica acerca de cual configuración es la mejor a comparación del resto. Esto se puede ver debido a que los mejores resultados (obtenidos con 9 celdas) son diferentes para validación que para entrenamiento y testing. Además, al revisar los datos de los gráficos de validación mostrados anteriormente, se pudo ver que este estaba generando un ligero overfitting.

La tabla 5.3 muestra resultados más concluyentes a comparación de las demás. Al igual que con los gráficos anteriores, se puede ver que si bien no se obtuvieron los mejores resultados finales para entrenamiento, si lo hizo para prueba y testeо. Además, se puede ver que estos empeoran mientras más celdas de BI-LSTM se usan. En ese sentido, se puede afirmar que esta es la mejor configuración para este modelo,

Observando los datos de la tabla 5.5 es notorio que los mejores resultados estuvieron en las configuraciones medias (4 y 5 celdas). Además también se logra idntificar que mientras más o menos celdas, todas las métricas se reducen, salvo con el caso de 9 celdas, el cual

lstm_cells	train_loss	train_f1	val_loss	val_f1	test_loss	test_f1
1	0.0562	0.9875	0.1048	0.9716	0.0721	0.9825
2	0.0527	0.9845	0.2139	0.9287	0.1006	0.9775
3	0.0237	0.9969	0.1774	0.9408	0.1117	0.965
4	0.092	0.9655	0.2456	0.8904	0.1914	0.935
5	0.1141	0.9649	0.3142	0.9295	0.1982	0.9325
6	0.0376	0.987	0.2237	0.9283	0.1158	0.97
7	0.0376	0.986	0.1745	0.9537	0.0706	0.9775
8	0.0497	0.986	0.186	0.9413	0.1219	0.965
9	0.0807	0.9772	0.1777	0.9282	0.1368	0.9625

Tabla 5.3: Tabla comparativa de las métricas obtenidas por configuración de celdas para MobileNetV3.

lstm_cells	train_loss	train_f1	val_loss	val_f1	test_loss	test_f1
1	0.1039	0.9769	0.1382	0.9716	0.116	0.9725
2	0.0907	0.9808	0.2067	0.9448	0.1111	0.97
3	0.0837	0.9771	0.1177	0.9586	0.1183	0.9675
4	0.0471	0.9904	0.1298	0.9537	0.0858	0.98
5	0.049	0.993	0.1499	0.9148	0.0866	0.9775
6	0.0925	0.9649	0.1553	0.9541	0.0973	0.975
7	0.1106	0.9551	0.18	0.9448	0.1374	0.9525
8	0.1262	0.9576	0.1257	0.9399	0.1074	0.975
9	0.0624	0.9872	0.1742	0.9716	0.1188	0.97

Tabla 5.4: Tabla comparativa de las métricas obtenidas por configuración de celdas para Resnet50.

es un dato anómalo en esa distribución.

Por último, a continuación se mostrará una tabla comparativa resumen de toda la experimentación realizada en el presente trabajo:

Con los resultados finales se puede corroborar que todas las combinaciones entre CNN y LSTM lograron valores similares para todas las métricas de testeo al tener un f1 score . En ese sentido, y con el fin de optimizar el *pipeline* final para el *dataset* propuesto, sería

cnn_model	lstm_cells	train_loss	train_f1	val_loss	val_f1	test_loss	test_f1
EfficientNetB0	9	0.012	0.9961	0.1564	0.9716	0.0605	0.98
EfficientNetV2	9	0.018	0.9969	0.2741	0.9295	0.0649	0.9825
MobileNetV3	1	0.0562	0.9875	0.1048	0.9716	0.0721	0.9825
ResNet50	4	0.0471	0.9904	0.1298	0.9537	0.0858	0.98

Tabla 5.5: Tabla comparativa resumen final de la experimentación

recomendable utilizar MobileNetV3 con una celda LSTM, ya que proporcionó el mismo resultado pero tanto el peso del modelo como el número de celdas es menor que en todas las demás configuraciones.

6. Conclusiones y Trabajo Futuro

A lo largo del desarrollo experimental de este trabajo, se exploraron diversas configuraciones arquitectónicas con el objetivo de determinar la combinación óptima entre complejidad estructural y rendimiento en la tarea de detección de violencia en videos del *dataset* Hockey Fights. Las pruebas realizadas incluyeron variantes en los modelos de CNN utilizados para la extracción de características y el número de unidades LSTM. No obstante, a pesar de estas diferencias en la arquitectura, los resultados obtenidos de las mejores configuraciones para cada CNN en el conjunto de prueba fueron notablemente consistentes.

Todos los modelos evaluados alcanzaron un desempeño muy similar en términos del F1 score, con un valor promedio de 98.25 %, lo que pone de manifiesto la solidez del enfoque adoptado. Este comportamiento sugiere que la calidad de las representaciones generadas por la red convolucional utilizada como extractor de características (MobileNetV3) resulta suficientemente informativa para que incluso una configuración secuencial mínima (como una única unidad LSTM) sea capaz de capturar de manera efectiva la dinámica temporal relevante para la clasificación. En consecuencia, se concluye que el sistema propuesto no solo ofrece un alto rendimiento en términos de precisión, sino que también presenta ventajas en cuanto a eficiencia computacional y simplicidad, factores especialmente valiosos para futuras implementaciones en entornos con recursos limitados o aplicaciones en tiempo real.

Como línea de trabajo futuro, se propone profundizar en el análisis de arquitecturas más complejas que integren mecanismos de atención o variantes modernas de redes recurrentes, tales como GRU o Transformers, con el fin de evaluar si estos enfoques pueden aportar mejoras adicionales en tareas de clasificación de video más exigentes o en contextos con mayor variabilidad visual. Asimismo, sería pertinente validar la robustez del modelo actual frente a escenarios del mundo real mediante su evaluación en datasets más extensos, heterogéneos y no balanceados, que representen condiciones variadas de iluminación, movimiento de cámara, resolución y presencia de ruido. También se considera de interés explorar técnicas de aprendizaje auto-supervisado o semi-supervisado que permitan aprovechar grandes volúmenes de datos no etiquetados, con el objetivo de reducir la dependencia de anotaciones manuales. Finalmente, la implementación del sistema en dispositivos con capacidad limitada de cómputo (como cámaras inteligentes o sistemas embebidos) podría abrir nuevas oportunidades de aplicación práctica, siempre que se ga-

rantice un equilibrio adecuado entre rendimiento, eficiencia y capacidad de inferencia en tiempo real.

Referencias

- Abdali, A. M. R., y Al-Tuma, R. F. (2019, 3). Robust real-time violence detection in video using cnn and lstm. *SCCS 2019 - 2019 2nd Scientific Conference of Computer Sciences*, 104-108. doi: 10.1109/SCCS.2019.8852616
- Amidi, S. (2018). *Cs 230 - recurrent neural networks cheatsheet*. Descargado de <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks>
- Bisca, P. M., Chau, V., Dudine, P., Espinoza, R. A., Fournier, J.-M., Guérin, P., ... Salas, J. (2024, 11). Violent crime and insecurity in latin america and the caribbean – a macroeconomic perspective. *Departmental Papers, 2024*. Descargado de <https://www.elibrary.imf.org/view/journals/087/2024/009/article-A001-en.xml> doi: 10.5089/9798400288470.087.A001
- DataScientest. (2024). *Memoria a largo plazo a corto plazo (lstm): ¿qué es?* Descargado de <https://datascientest.com/es/memoria-a-largo-plazo-a-corto-plazo-lstm> (Accedido el 13 de abril de 2025)
- de Justicia y Seguridad Pública, M. (2025). *Sistema de alerta temprana (sat)*. Descargado de https://www.seguridad.gob.sv/cna/?page_id=3742
- de la Ciudad de México, G. (2008). *Implementación del botón de auxilio en la ciudad de méxico*. Descargado de <https://www.c5.cdmx.gob.mx/canales-de-atencion-emergencias/boton-de-auxilio> (Accedido el 1 de mayo de 2025)
- Diego Calvo. (2019a). *red-neuronal-convolucionar*. Descargado de <https://www.diegocalvo.es/red-neuronal-convolucionar/> ([Online; accessed December 05, 2021])
- Diego Calvo. (2019b). *red-neuronal-convolucionar-arquitectura*. Descargado de <https://www.diegocalvo.es/red-neuronal-convolucionar/red-neuronal-convolucionar-arquitectura/> ([Online; accessed December 05, 2021])
- Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., y Darrell, T. (2014). Long-term recurrent convolutional networks for visual

- recognition and description. *CoRR*, *abs/1411.4389*. Descargado de <http://arxiv.org/abs/1411.4389>
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... Houlsby, N. (2020, 10). An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR 2021 - 9th International Conference on Learning Representations*. Descargado de <https://arxiv.org/pdf/2010.11929.pdf>
- Elman, J. L. (1990). Finding structure in time. *Cognitive science*, *14*(2), 179–211. doi: 10.1207/s15516709cog1402_1
- Emsley, C. (2007, 9). Crime, police, and penal policy: European experiences 1750-1940. *Crime, Police, and Penal Policy: European Experiences 1750-1940*, 1-284. Descargado de <https://academic.oup.com/book/32821> doi: 10.1093/ACPROF:OSO/9780199202850.001.0001
- Gilliver, K. (2005, 11). The complete roman army. by a.k. goldsworthy. thames and hudson, london, 2003. pp. 224, illus 245. price: £24.95. isbn 0 500 05124 0. *Britannia*, *36*, 512-513. doi: 10.3815/000000005784016757
- He, K., Zhang, X., Ren, S., y Sun, J. (2015). Deep residual learning for image recognition.
- Hochreiter, S., y Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.
- IchiPro. (2020). 4 modelos de cnn previamente entrenados para usar en visión artificial con aprendizaje por transferencia. Descargado de <https://ichi.pro/es/4-modelos-de-cnn-previamente-entrenados-para-usar-en-vision-artificial-con-aprendizaje-por-transferencia-9370731228668> ([Online; accessed December 05, 2021])
- INEGI. (2024). Encuesta nacional de victimización y percepción sobre seguridad pÚblica (envipe) 2024 (Inf. Téc.). Autor. Descargado de https://www.inegi.org.mx/contenidos/saladeprensa/boletines/2024/ENVIPE/ENVIPE_24.pdf
- Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., NanoCode012, Kwon, Y., ... Jain, M. (2022, noviembre). ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation. Zenodo. Descargado de <https://doi.org/10.5281/zenodo.7347926> doi: 10.5281/zenodo.7347926
- Kulkarni, A., Batarseh, F. A., y Chong, D. (2021). Chapter 5: Foundations of data imbalance and solutions for a data democracy. *ArXiv*.
- Madera, C., y López, C. (2024). Machine learning pipeline para el etiquetado automático

- co en imágenes de especies de peces peruanos. *Repositorio Institucional UTEC.* Descargado de <http://repositorio.utec.edu.pe/handle/20.500.12815/396>
- Mann, P. (2021, 7). Real-time violence detection using cnn-lstm. *charotar university of science and technology.* Descargado de https://www.researchgate.net/publication/353330450_Real-Time_Violence_Detection_Using_CNN-LSTM doi: 10.48550/arXiv.2107.07578
- Marois, A., Hodgetts, H. M., Chamberland, C., Williot, A., y Tremblay, S. (2021, 7). Who can best find waldo? exploring individual differences that bolster performance in a security surveillance microworld. *Applied Cognitive Psychology, 35*, 1044-1057. doi: 10.1002/ACP.3837
- Mohammadi, S. V., y Yen, K.-C. (2022). A cnn-rnn combined structure for real-world violence detection in surveillance cameras. *Applied Sciences, 12*(3), 1021. Descargado de <https://www.mdpi.com/2076-3417/12/3/1021> doi: 10.3390/app12031021
- Muhamad Yani, Budhi Irawan, Casi Setianingsih. (2019). *Application of transfer learning using convolutional neural network method for early detection of terry's nail.* Descargado de https://www.researchgate.net/figure/Illustration-of-Max-Pooling-and-Average-Pooling-Figure-2-above-shows-an-example-of-max_fig2_333593451 ([Online; accessed December 05, 2021])
- Negre, P., Alonso, R. S., Prieto, J., Dang, C. N., y Corchado, J. M. (2024, 3). Systematic mapping study on violence detection in video by means of trustworthy artificial intelligence. *SSRN Electronic Journal.* Descargado de <https://papers.ssrn.com/abstract=4757631> doi: 10.2139/SSRN.4757631
- Negre, P., Alonso, R. S., Prieto, J., Garcia, O., y Corchado, J. M. (2024, 5). Violence detection in video models implementation using pre-trained vgg19 combined with manual logic, lstm layers and bi-lstm layers. *SSRN Electronic Journal.* Descargado de <https://papers.ssrn.com/abstract=4832475> doi: 10.2139/SSRN.4832475
- Nievas, E. B., Suarez, O. D., Garcia, G. B., y Sukthankar, R. (2010). Violence detection in video using computer vision techniques. *Computer Vision and Image Understanding, 114*(2), 158–169.
- OMS. (2014). Violencia y salud mental. *Organismo Mundial de la Salud.* Descargado de <https://www.uv.mx/psicologia/files/2014/11/violencia-y-salud-mental-oms.pdf>
- Organization, W. H. (2024). Monitoring health for the sdgs, sustainable development

- goals. *World Health Organization Journal*.
- Orozco, C. I., Buemi, M. E., y Berlles, J. J. (2021, 6). Cnn-lstm con mecanismo de atención suave para el reconocimiento de acciones humanas en videos. *Elektron*, 5, 37-44. doi: 10.37537/REV.ELEKTRON.5.1.130.2021
- Police Executive Research Forum. (2003). *Compstat: Its origins, evolution, and future in law enforcement agencies* (Inf. Téc.). Bureau of Justice Assistance, U.S. Department of Justice. Descargado de <https://bja.ojp.gov/sites/g/files/yxckuh186/files/Publications/PERF-Compstat.pdf> (Accessed: 2025-05-01)
- Presti, L. L., y Cascia, M. L. (2016, 5). 3d skeleton-based human action classification: A survey. *Pattern Recognition*, 53, 130-147. Descargado de <https://www.sciencedirect.com/science/article/abs/pii/S0031320315004392> doi: 10.1016/J.PATCOG.2015.11.019
- Redmon, J., Divvala, S., Girshick, R., y Farhadi, A. (2015, jun). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December*, 779–788. Descargado de <https://arxiv.org/abs/1506.02640v5> doi: 10.48550/arxiv.1506.02640
- Ren, B., Liu, M., Ding, R., y Liu, H. (2024). *A survey on 3d skeleton-based action recognition using learning method*.
- Services, A. W. (2024). *¿qué son los transformadores?: explicación de los transformadores en inteligencia artificial: Aws*. Descargado de <https://aws.amazon.com/es/what-is/transformers-in-artificial-intelligence/>
- Sharma, S., Sudharsan, B., Naraharisetti, S., Trehan, V., y Jayavel, K. (2021, 8). A fully integrated violence detection system using cnn and lstm. *International Journal of Electrical and Computer Engineering*, 11, 3374-3380. doi: 10.11591/IJECE.V11I4.PP3374-3380
- Sheptycki, J. (1998). Policing, postmodernism and the post-colonial. *Theoretical Criminology*, 2(1), 89–113. doi: 10.1177/1362480698002001006
- Simonyan, K., y Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition*. Descargado de <http://www.robots.ox.ac.uk/>
- Sudhakaran, S., y Lanz, O. (2017a). Learning to detect violent videos using convolutional long short-term memory. *arXiv preprint arXiv:1709.06531*. Descargado de <https://arxiv.org/abs/1709.06531>

- Sudhakaran, S., y Lanz, O. (2017b, 10). Learning to detect violent videos using convolutional long short-term memory. *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017*. doi: 10.1109/AVSS.2017.8078468
- Szegedy, C., Liu, W., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., ... Rabinovich, A. (2014). *Going deeper with convolutions*.
- Tan, M., y Le, Q. V. (2020). *Efficientnet: Rethinking model scaling for convolutional neural networks*. Descargado de <https://arxiv.org/pdf/1905.11946.pdf>
- Tashin Ahmed, N. H. S. (2020). *Classification and understanding of cloud structures via satellite images with efficientunet*. Descargado de https://www.researchgate.net/figure/Architecture-of-EfficientNet-B0-with-MBConv-as-Basic-building-blocks_fig4_344410350 ([Online; accessed December 05, 2021])
- Vilalta, C. J., Sanchez, T. W., Fondevila, G., y Ramirez, M. (2019, 3). A descriptive model of the relationship between police cctv systems and crime. evidence from mexico city. *Police Practice and Research*, 20, 105-121. Descargado de https://www.researchgate.net/publication/325268046_A_descriptive_model_of_the_relationship_between_police_CCTV_systems_and_crime_Evidence_from_Mexico_City doi: 10.1080/15614263.2018.1473770
- Weisburd, D., Mastrofski, S. D., McNally, A. M., Greenspan, R., y Willis, J. J. (2003). Reforming to preserve: Compstat and strategic problem solving in american policing. *Criminology and Public Policy*, 2(3), 421-456. doi: 10.1111/j.1745-9133.2003.tb00006.x
- Wenjin Taoa, Md Al-Aminb, Haodong Chena, Ming C. Leua, Zhaozheng Yinc, Ruwen Qinb. (2020). *Real-time assembly operation recognition with fog computing and transfer learning for human-centered intelligent manufacturing*. Descargado de https://www.researchgate.net/figure/The-architecture-of-our-transfer-learning-model_fig4_342400905 ([Online; accessed December 05, 2021])
- Yalcin, O. (2020). *Pre-trained model performances.csv*. Descargado de <https://gist.github.com/ogyalcin/052f2df49b3288e62086aa0e5fd25fc>
- Yani, M., Irawan, B., y Setiningsih, C. (2019, 5). Application of transfer learning using convolutional neural network method for early detection of terry's nail. *Journal of Physics: Conference Series*, 1201. Descargado de https://www.researchgate.net/publication/333593451_Application_of_Transfer_Learning_Using

Cesar Antonio Madera Garcés
Máster en en Ingeniería Matemática y Computación

[Convolutional_Neural_Network_Method_for_Early_Detection_of_Terry's_Nail](#)
doi: 10.1088/1742-6596/1201/1/012052

A. Apendices