

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL
INSTITUTO POLITÉCNICO NACIONAL

Unidad Saltillo
Robótica y Manufactura Avanzada

Detección de eventos en tiempo real mediante redes neuronales convolucionales y su validación con modelos largos de lenguaje multimodales

Tesis que presenta

Ángel Andrés Hernández García

para obtener el Grado de

Maestro en Ciencias
en
Robótica y Manufactura Avanzada

Director de tesis: Dr. Reyes Ríos Cabrera

Dedico a..

Agradezco a...

Resumen

En la presente tesis se explora un esquema híbrido de procesamiento de vídeo en tiempo real y la verificación de detecciones mediante modelos multimodales, con funcionamiento eficiente en plataformas de procesamiento on edge. Se realizó la implementación de una arquitectura de redes neuronales convolucionales (CNN) que detectan eventos de manera veloz para ser procesados por un Modelo Largo del Lenguaje Multimodal (MLLM), el cual determina el contexto y veracidad de cada uno. Se proponen nuevos algoritmos para reducir la latencia al momento de verificar los eventos, limitando la influencia de la marcada diferencia de tiempos de procesamiento entre modelos, mientras se reduce al mínimo la perdida de precisión.

Abstract

Índice general

Lista de Figuras	XIV
Acrónimos	XV
1. Introducción	1
1.1. Definición del problema	1
1.2. Motivación	2
1.3. Objetivos	2
1.3.1. Objetivo general	2
1.3.2. Objetivos específicos	2
1.4. Hipótesis	3
1.5. Organización del documento	3
2. Conceptos generales	5
2.1. Deep Learning	5
2.2. Redes neuronales convolucionales (CNN)	6
2.3. Transformers	7
2.4. Vision transformers	9
2.5. Cuantización de modelos	11
3. Estado del arte	13
3.1. Detectores multiclas	13
3.2. MLLM's	15
3.3. Weakly Supervised Video anomaly detection (WSVAD)	18
3.4. MLLMs en WSVAD	20
4. Banco de pruebas	23
4.1. Hardware	23
4.2. Software	23
4.3. Métodos	24
4.4. Datasets	24
4.4.1. CHAD	24
4.4.2. IITB-Corridor	25

4.4.3. NWPU Campus	26
4.4.4. Avenue Dataset	27
4.4.5. Distribución de los videos	27
5. Metodología	31
5.1. Algoritmos propuestos	31
5.1.1. Espaciado temporal entre fotogramas y limitaciones en la cantidad.	31
5.1.2. Organizador de detecciones de personas	32
5.1.3. Reglas en base a las detecciones	32
5.1.3.1. Presencia de una clase específica	33
5.1.3.2. Cambios de la bounding box de un objeto a lo largo del tiempo	33
5.1.3.3. Interacción entre bounding boxes de dos objetos a lo largo del tiempo	34
5.1.4. Elaboración del prompt	34
5.2. Arquitectura propuesta	35
5.3. Descripción de los métodos	36
5.3.1. MLLM Solo	36
5.3.2. Detector con reglas	36
5.3.3. Detector, MLLM e información.	37
5.3.4. Detector con reglas y MLLM	38
5.3.5. Detector con reglas, MLLM e información	38
5.4. Uso de CLIP	39
5.4.1. CLIP con reglas	39
5.4.2. CLIP con reglas y MLLM	40
5.5. Evaluación utilizada	40
6. Experimentación y resultados	41
6.1. Experimentos realizados	41
6.1.1. Selección del MLLM	41
6.1.2. Selección del prompt para MLLM	42
6.1.3. Análisis de los diferentes prompts	43
6.1.3.1. Prompt simple	44
6.1.3.2. Prompt con localización de los objetos	44
6.1.3.3. Comparativa	45
6.1.4. Análisis de los videos utilizados durante el desarrollo	46
6.1.5. Incorporación de los nuevos videos	47
6.2. CLIP en la arquitectura	48
6.3. Análisis	51
6.4. Comparativa	53
7. Conclusiones	55

Apéndices	59
A. Algoritmos y códigos	59
Bibliografía	61

Índice de figuras

2.1. Funcionamiento de una red neuronal convolucional [LeCun et al., 2015].	7
2.2. Arquitectura del transformer [Vaswani et al., 2023].	8
2.3. Atención de Producto Punto Escalado (SDPA) [Vaswani et al., 2023].	8
2.4. Atencion multicabeza [Vaswani et al., 2023].	9
2.5. Arquitectura del Vision Transformer [Dosovitskiy et al., 2021].	10
2.6. Vista general de CLIP	10
2.7. Esquema del proceso de cuantización de LLM.int8()	11
3.1. Visión general de RT-DETR [Zhao et al., 2024].	14
3.2. Arquitectura general de los MLLMs [Caffagni et al., 2024].	15
3.3. Modelo de MOAI [Lee et al., 2024].	16
3.4. MOAI-Mixer [Lee et al., 2024].	16
3.5. Modelo de Florence-2 [Xiao et al., 2023].	17
3.6. Arquitectura propuesta para SmolVLM.	18
3.7. La arquitectura de VadCLIP.[Wu et al., 2023]	19
3.8. Estrategias de muestreo para los MLLMs [Ding and Wang, 2024].	20
3.9. Arquitectura de Holmes-VAU [Zhang et al., 2025].	21
3.10. Pipeline de AnomalyRuler.	22
4.1. Vistas de las cámaras [Danesh Pazho et al., 2023].	25
4.2. Eventos anómalos de CHAD [Danesh Pazho et al., 2023].	25
4.3. Ejemplos de los videos y sus anomalías identificadas [Rodrigues et al., 2020] .	26
4.4. Eventos anómalos en los videos [Cao et al., 2023].	26
4.5. Ejemplos de los escenas y sus eventos anomalous de NWPU Campus dataset [Cao et al., 2023]	27
4.6. Videos de CHAD entre los 13 eventos.	28
4.7. Distribucion de los 98 videos de las demás datasets.	28
4.8. Distribucion de los 252 videos de las demás datasets.	29
5.1. Vista general de la arquitectura, incluyendo todos los elementos [Caffagni et al., 2024].	35
5.2. Uso del MLLM para la detección de anomalías en videos.	36
5.3. Uso de las reglas para la detección de anomalías en videos.	36
5.4. Uso del MLLM con información en los prompts para la detección de anomalías en videos.	37

5.5. Uso del MLLM y las reglas de forma conjunta para la detección de anomalías en videos.	38
5.6. Uso del detector y las reglas para detectar anomalías en videos con CLIP	39
5.7. Uso de todos los componentes de la arquitectura detectar anomalías en videos con CLIP.	40
6.1. Average Precision y FPS de cada MLLM.	42
6.2. Average Precision de las diversas preguntas a 107 videos de la CHAD dataset en 6 eventos.	43
6.3. Average Precision y FPS de cada método con información de los objetos detectados.	44
6.4. Average Precision y FPS de cada método con información de los objetos detectados y su ubicación en las escenas.	45
6.5. Average Precision y FPS de cada método con todos los videos de la CHAD dataset.	46
6.6. Average Precision y FPS de cada métodos con los videos de las demás datasets	47
6.7. Average Precision y FPS de cada método con todos los videos	48
6.8. Resultados del uso de CLIP en los videos de CHAD Dataset.	49
6.9. Resultados del uso de CLIP en los videos de las demás datasets.	50
6.10. Resultados del uso de CLIP en todos los videos.	50

Acrónimos

CINVESTAV	Centro de Investigación y de Estudios Avanzados del IPN
GRyMA	Grupo de Robótica y Manufactura Avanzada
CNN	Red Neuronal Convolucional
MLLM	Modelos largos del lenguaje multimodales.

Capítulo 1

Introducción

Con el auge de la inteligencia artificial, diversos tipos de modelos han surgido para llevar a cabo múltiples tipos de tareas, yendo desde la detección en imágenes, hasta cuestiones lingüísticas, contando cada uno con su propia estructura y algoritmos desarrollados por su equipo con la finalidad de cumplir su propósito de la manera mas eficiente posible.

Aunque cada modelo cuenta con su principal utilidad, se puede ver limitado por diversos factores, como el poder de computo, la velocidad de procesamiento, precisión, o inclusive el entrenamiento, los cuales reducen su aplicación.

Sin embargo, es posible el uso conjunto de múltiples modelos, tomando las fortalezas de cada uno para un rendimiento global destacable, todo para una aplicación como puede ser la detección de eventos en tiempo real, la cual se basa en los detectores multiclases, ya que destacan por su velocidad , sin embargo, pueden presentar redundancias en los resultados o fallos en la clasificación, mientras que los modelos capaces de solventar estas limitaciones son los MLLMs, ya que son capaces de entender el contexto de la información que reciben, pero con un coste computacional mayor, lo que conlleva a una menor velocidad de procesamiento.

En este trabajo se propone el uso conjunto de detectores multiclases, basados en redes convolucionales, y MLLMs, desarrollando una metodología para conseguir la menor latencia en la comunicación entre modelos, y buscando la mayor precisión en las detecciones.

1.1. Definición del problema

Los detectores multiclases pueden identificar eventos en tiempo real, pero carecen de entendimiento del contexto de las imágenes, lo cual puede ocasionar dificultades al distinguir entre

dos eventos contextualmente diferente, pero que cuentan con un parecido visual. Por su parte, los modelos largos de lenguaje multimodales interpretan los eventos dentro de su entendimiento del contexto. Estos modelos son altamente poderosos, pero su limitante es la velocidad de procesamiento, ya que son computacionalmente demandantes y lentos.

1.2. Motivación

Al combinar los detectores multiclasses con los modelos largos del lenguaje multimodales, se busca que los eventos pueden ser identificados velozmente y siendo validados con precisión.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar una arquitectura multi-frames por segundo de redes neuronales convolucionales (CNN) para un detector multiclasses capaz de funcionar eficientemente en plataformas de procesamiento on-edge para análisis de vídeos en tiempo real, y con verificación de eventos aprovechando los modelos largos de lenguaje multimodales (MLLMs), los cuales se encargar de analizar en detalle la evidencia proveída por el detector y decidir si es un falso positivo o un evento real.

1.3.2. Objetivos específicos

- Proponer una metodología y algoritmos para combinar detectores multiclas y modelos largos de lenguaje multimodales (MLLMs)
- Implementar una arquitectura de redes neuronales convoluciones multi-frame para la detección eficiente de eventos en una Nvidia Jetson/Plataforma de OpenVino.
- Optimizar el sistema para el procesamiento en tiempo real al combinar la interacción entre CNNs y modelos largos del lenguaje multimodales (MLLM) con el fin de minimizar la latencia al realizar la verificación de eventos.
- Implementar el detector multiclas en plataformas de procesamiento on-edge, centrándose en el eficiente uso de recursos para el análisis de vídeos en tiempo real.

1.4. Hipótesis

Una eficiente arquitectura de análisis de vídeos capaz de funcionar con información en tiempo real utilizando redes neuronales convolucionales, al ser combinada con un modelo largo del lenguaje multimodal (MLLM) para validar las situaciones detectadas, puede aprovechar el poder de las diversas modalidades para discernir e interpretar imágenes más eficientemente mediante el entendimiento contextual.

1.5. Organización del documento

El presente artículo cuenta con

Capítulo 2

Conceptos generales

2.1. Deep Learning

El aprendizaje profundo, o Deep Learning permite a los modelos computacionales, compuestos de múltiples capas de procesamiento, aprender representaciones de la información con múltiples niveles de abstracción [LeCun et al., 2015]. La clave del Deep Learning es aprender desde la información utilizando un procedimiento de aprendizaje.

La manera de aprendizaje mas común para el Deep Learning es el aprendizaje supervisado (Supervised Learning), en el cual se cuenta con una gran dataset con anotaciones individuales para elemento. Durante el entrenamiento, se produce la salida del modelo frente a los elementos de la dataset, siendo un vector de puntajes, al esperarse que el puntaje de la categoría anotada para cada elemento sea la mayor.

Calculando una función objetivo que mide el error entre los puntajes obtenido por el modelo y los deseados por las anotaciones, modificando sus parámetros internos llamados pesos,los cuales definen la relación entre la entrada y salida de los modelos.

Para ajustar los pesos, el algoritmo de aprendizaje calcula el gradiente, así para cada peso se indica la cantidad del error a ajustar en dirección opuesta al gradiente. En la practica se utiliza un procedimiento llamado Descenso del gradiente estocástico (SGD), el cual realiza el ajusta en los pesos con pocos ejemplos, siendo un proceso que se repite para conjuntos de elemento de la parte de entrenamiento de la dataset hasta que la función objetivo deje de decrecer; después del entrenamiento se mide el desempeño del modelo en un conjunto diferente de elementos, siendo la parte de prueba de la dataset, sirviendo para probar la capacidad de generalización del sistema, en concreto su capacidad de producir respuestas a entradas que nunca ha visto en el entrenamiento .

Una arquitectura de Deep Learning es una pila de múltiples capas de módulos simples, que son sometidos a aprender, y muchos realizan mapeos no lineares de entrada-salida. Cada modulo transforma su entrada para incrementar la selectividad e invarianza de la representación de la información. Con las capas no lineales permiten al modelo ser sensible a detalles minúsculos en la información, mientras es insensible a grandes variaciones independientes.

Desde los primeros días del área de reconocimiento de patrones, la investigación se enfoco en reemplazar la extracción de características manual por medio de redes multi capas entrenables, resultado en arquitecturas multicapas que pueden ser entrenadas por medio del Descenso del gradiente estocástico (SDG), permitiendo el procedimiento de Backpropagation, el cual calcula el gradiente de la función objetivo con respecto a los pesos de la pila de múltiples capas de módulos, siendo una aplicación práctica de la regla de la cadena para derivadas. La clave de este procedimiento es que la derivada (o gradiente) del objetivo con respecto a la entrada de un módulo puede calcularse utilizando el gradiente con respecto a la salida de ese módulo (o la entrada del módulo subsiguiente) [LeCun et al., 2015]

Backpropagation puede ser aplicado repetidamente para propagar los gradientes a todos los módulos, empezando de la salida, hasta los módulos de la entrada.

Muchas de las aplicaciones utilizan arquitecturas de redes neuronales prealimentadas (FFN), las cuales aprenden a mapear una entrada de tamaño definido a una salida de tamaño definido. Para pasar de capa, se calcula la suma ponderada de sus entradas de capas previas, y pasa el resultado mediante una función no lineal.

2.2. Redes neuronales convolucionales (CNN)

Un particular tipo de red neuronal es la red neuronal convolucional (CNN), ideadas para procesar información en forma de múltiples arreglos, como audio, imágenes o videos. La arquitectura de una típica red neuronal convolucional esta estructurada como una serie de fases.

Las primeras fases están compuestas de dos tipos de capas, las convolucionales y las de agrupación (Pooling). Las unidades de una capa convolucional están organizados en mapas de características, donde cada una esta conectada a una sección local en los mapas de la anterior capa mediante un conjunto de pesos, cuya suma ponderada es pasada por una función de activación no-lineal como ReLU. El rol de las capas convolucionales es detectar conjuntos locales de características de la capa anterior, mientras que el de la capa de agrupamiento es combinar semanticamente características similares en una. Una unidad de agrupamiento típica calcula el máximo de una sección local de unidades en un mapa de características.

Dos o tres fases de capas convolucionales, funciones de activación no lineales y agrupamiento apiladas, seguidas de mas capas convolucionales y capas totalmente conectadas (Fully connected). El algoritmo de Backpropagation en una red neuronal convolucional es igual que en una red neuronal normal.

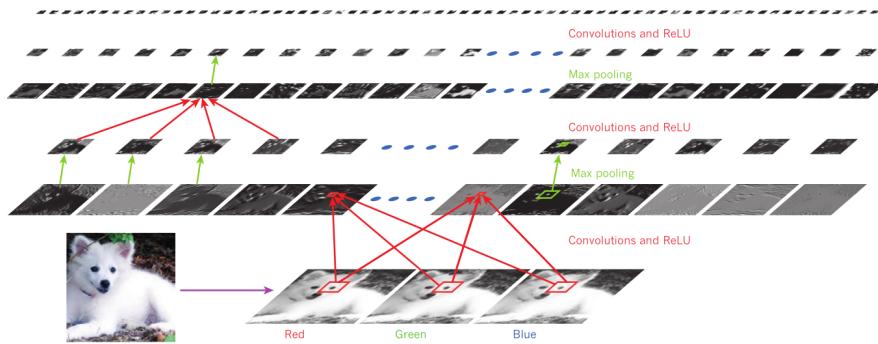


Figura 2.1: Funcionamiento de una red neuronal convolucional [LeCun et al., 2015].

2.3. Transformers

Los modelos de transducción de secuencias neuronales tienen una estructura codificador-decodificador, en donde el codificador mapea una secuencia de representaciones simbólicas a una secuencia de representaciones continuas; el decodificador usa la secuencia dada para generar una secuencia de salida de símbolos, un elemento a la vez.

El modelo es autorregresivo en cada paso, consumiendo los símbolos anteriormente generados como una entrada adicional.

El transformador es una arquitectura modelo que se basa en un mecanismo de atención para capturar las dependencias globales entre la entrada y la salida.

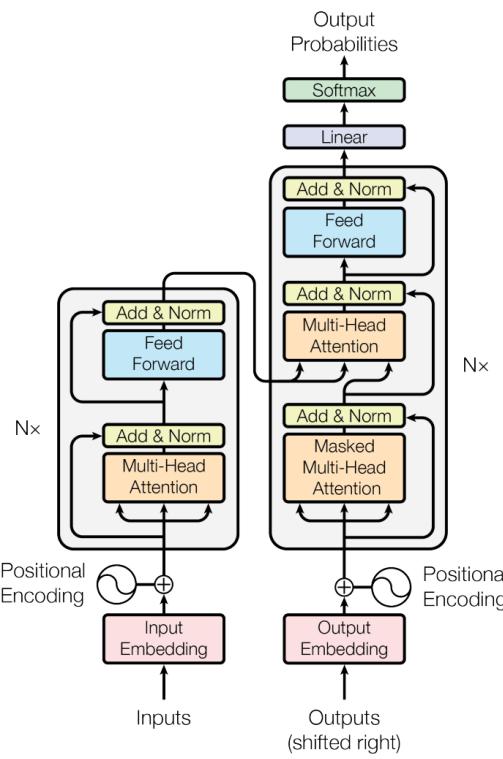


Figura 2.2: Arquitectura del transformer [Vaswani et al., 2023].

Una función de atención puede ser descrita como un mapear una consulta y un conjunto de pares clave-valor a una salida, que es calculada como la suma ponderada de los valores, con pesos obtenidos al usar el producto punto escalado.

Scaled Dot-Product Attent

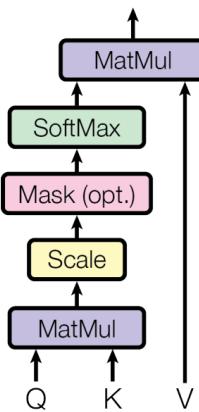


Figura 2.3: Atención de Producto Punto Escalado (SDPA) [Vaswani et al., 2023].

La atención multicabeza permite al modelo atender conjuntamente a la información de distintos subespacios en diferentes posiciones, donde en vez de realizar una función de atención simple, es beneficioso proyectar las claves, valores y consultas múltiples veces.

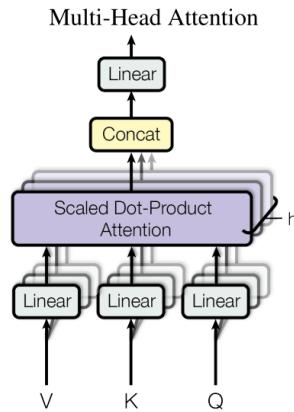


Figura 2.4: Atencion multicabeza [Vaswani et al., 2023].

La función de atención se realiza en paralelo, obteniendo valores de salida que se concatenan y proyectan de nuevo hasta obtener los valores finales.

Debido a la capacidad de la arquitectura de los transformers para crear modelos de gran capacidad y pre entrenarlos, es que es posible utilizarlos en una gran variedad de tareas, por lo surge una librería de código abierto que busca liberar estos avances a la comunidad de Machine Learning, dando pie a la librería Transformers [Wolf et al., 2020]. La librería consiste una colección de modelos pre entrenados disponibles bajo un API unificada, facilitando la distribución de los modelos, así como apoyar las implementaciones.

La librería esta diseñada para seguir el pipeline estándar de los modelos de NLP, que es procesar la información, aplicar el modelo y hacer predicciones. La libreria cuenta con herramientas que facilitan el entrenamiento y desarrollo. Los modelos de la libreria son compatibles con PyTorch y TensorFlow, así como capaz de exportar los modelos a formatos intermedios, como ONNX.

2.4. Vision transformers

Desarrollado para aplicar la arquitectura estándar del transformer directamente en imágenes con las menores modificaciones posibles, es como surge el Vision Transformer (ViT).

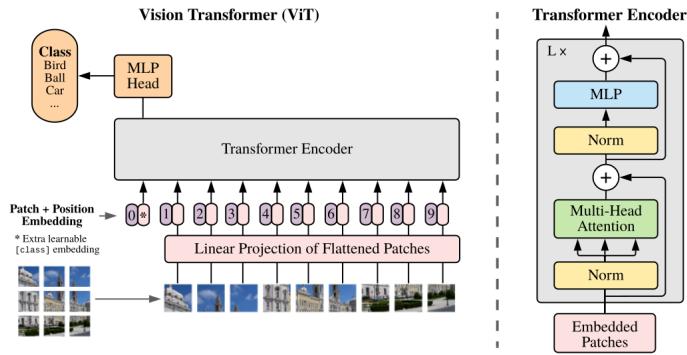


Figura 2.5: Arquitectura del Vision Transformer [Dosovitskiy et al., 2021].

El ViT trabaja con secciones de imágenes, las cuales son procesadas en embeddings lineales que sirven de entrada al transformer, las secciones son tratadas de manera análoga a los tokens de las palabras.

Existiendo modelos que aprovechan las capacidades de los transformers en el área de visión por computador, como CLIP (Contrastive Language-Image Pre-Training), la cual se introduce como una red neuronal entrenada en una variedad de pares de imágenes y texto. CLIP entrena de manera conjunta un codificador de imágenes y un codificador de texto para predecir el correcto emparejamiento de un lote de ejemplos de entrenamiento de imágenes y texto. Durante las pruebas, el codificador de texto con entrenamiento sintetiza un clasificador lineal zero-shot al realizar embeddings de los nombres o descripciones de las clases de los datasets. [Radford et al., 2021].

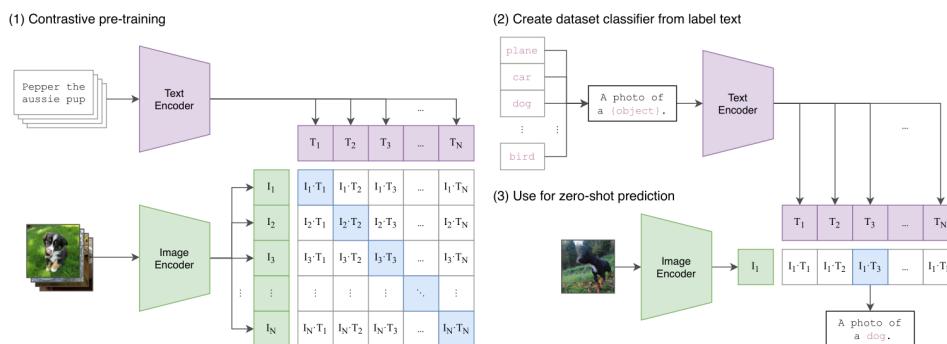


Figura 2.6: Vista general de CLIP

2.5. Cuantización de modelos

Los LLMs requiere una significativa memoria para la inferencia, y para modelos 6.7 billones de parámetros en adelante, las capas de atención y feed-forward con sus operaciones de multiplicación de matrices son responsables del 95 % de los parámetros utilizados, así como del 65 – 85 % de todo costo computacional. Una manera de reducir el tamaño de los parámetros es cuantizarlos hacia un número menor de bits y usar multiplicación de matrices de bits de baja precisión. Mientras estos métodos reducen el uso de memoria, degradan el desempeño. Buscando evitar la degradación de los modelos, incluso en aquellos con billones de parámetros es que se propone LLM.int8(), convirtiendo los pesos del transformers de 16/32-bits a 8-bit para utilizar en la inferencia.

[Dettmers et al., 2022]

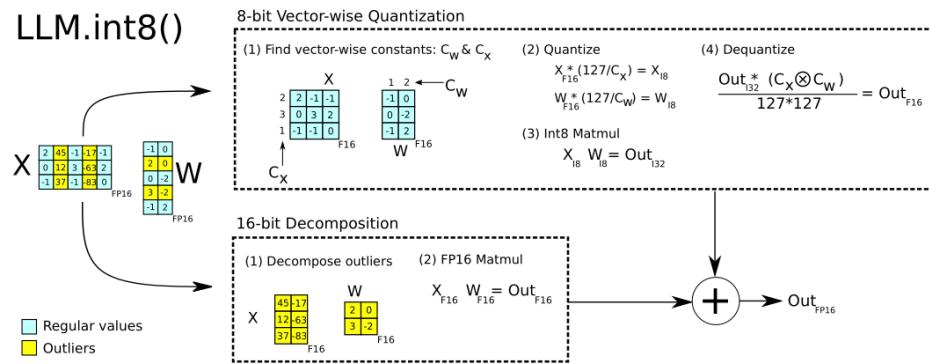


Figura 2.7: Esquema del proceso de cuantización de LLM.int8()

Como se puede apreciar en la figura 2.7, LLM.int8() es un método que combina dos métodos, cuantización vectorial (vector-wise quantization) y descomposición de precisión mixta (mixed precision decomposition). Donde las entradas y pesos en punto flotante de 16-bit (FP16), son separadas en submatrices, en una con valores atípicos no se cuantiza y realiza una multiplicación de matrices en FP16, mientras la otra matriz con los valores normales se cuantiza vectorialmente para ejecutar un multiplicación matricial en enteros de 8-bits, para descuantizar el resultado a FP16. Ambas salidas de las submatrices son acumuladas una única de punto flotante de 16-bits. La separación en 8-bits y 16-bits permite una multiplicación de alta precisión para los valores atípicos, mientras que la multiplicación matricial eficiente en memoria con 8-bits se lleva a cabo para el 99.9 % de los valores restantes.

Siendo una de las propuestas cuantizar los transformers con múltiples billones de parámetros sin degradar el desempeño.

Capítulo 3

Estado del arte

En la actualidad, la tecnología del Machine Learning es la base para una gran diversidad de aplicaciones, y cada vez se hace mas uso de las tecnicas de Deep Learning.

A diferencia de las técnicas convencionales, que requieren de sistemas especializados para convertir la información en una apropiada entrada para los modelos, los métodos de Deep Learning transforman la información hacia una representación adecuada, en la cual pueden aprender como destacar sus características importantes mediante un proceso de aprendizaje general.

Una estructura de Deep Learning es multicapa de módulos simples, los cuales todos o la mayoría se someten a aprendizaje, calculando relaciones no lineales de entrada-salida; la forma mas común en la que estos modelos aprenden es con aprendizaje supervisado, donde con una gran cantidad de información etiquetada en su categoría se procesa durante el entrenamiento, e internamente se ajustan los parámetros al ir aprendiendo [LeCun et al., 2015].

3.1. Detectores multiclasses

Con el uso de redes neuronales convolucionales, se ha facilitado la incorporación de sus habilidades en aplicaciones en tiempo real, siendo utilizadas en los detectores multiclasses.

Siendo YOLO un paradigma dominante en el campo de la detección de objetos en tiempo real debido a su balance entre costo computacional y rendimiento, este carece de comprensión, resultando en redundancias que limitan la capacidad del modelo, por lo que en la versión diez se exploró el uso de convolución de kernels grandes y un módulo de auto-atención parcial eficaz para mejorar la capacidad del modelo, aprovechando el potencial de la mejora del rendimiento a un bajo costo computacional.

Para lograrlo se presento una versión de YOLO libre del post procesamiento con NMS (Non-Maximum Supression) en el entrenamiento, ya que al suprimir las detecciones redundantes causaba una eficiencia de inferencia suboptima para las implementaciones, por lo que se desarrollo una estrategia con asignación de etiquetas duales y una métrica consistente de concordancia.

El módulo de autoatención parcial (PSA) se coloca con la menor resolución, evitando la sobrecarga excesiva de la complejidad computacional cuadrática de la autoatención. De este modo, se pueden utilizar los mecanismos de atención, que son base para de los transformers, para incorporarse a YOLO con bajo coste computacionale, mejorando la capacidad del modelo y su rendimiento [Wang et al., 2024a].

Los detectores basados en transformadores (DETR) han surgido como otra alternativa para eliminar el NMS (Non-Maximum Supression) en el post-procesamiento, pero su alto costo computacional limita su practicidad, limitando su aprovechamiento, siendo la razón detrás del desarrollo para buscar su funcionamiento en tiempo real.

Para conseguirlo se rediseño el codificador, ya que era el punto donde se generaba el mayor cuello de botella computacional, resultando en un codificador híbrido, el cual mejora la velocidad de inferencia al desacoplar la interacción y fusión de las características con diversas escalas, así como la selección de consultas, optimizando la precisión.

El RT-DETR consiste de un backbone, el codificador híbrido y un Transformer decodificador; el codificador transforma las características multiescalas obtenidas del backbone a una secuencia de características de las imágenes, consta de dos módulos, un modulo basado en atención y otro basado en redes convolucionales, buscando reducir el costo computacional.

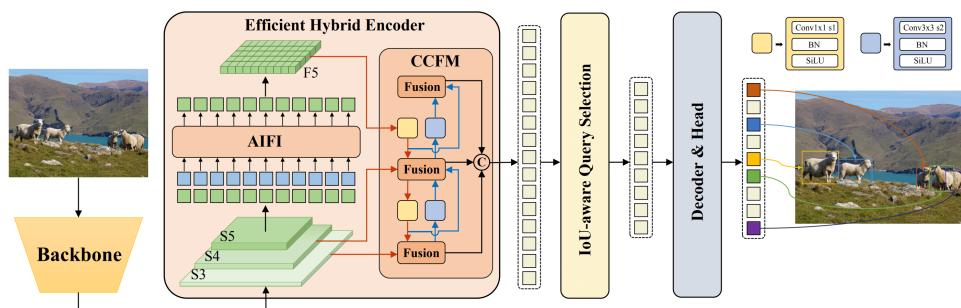


Figura 3.1: Visión general de RT-DETR [Zhao et al., 2024].

El selector de consultas se emplea para seleccionar un numero fijo de características dadas por el codificador para servir como consultas iniciales de objetos para el decodificador, para que al final el decodificador iterativamente optimiza las consultas para generar las categorías y las cajas [Zhao et al., 2024].

3.2. MLLM's

Los modelos largos del lenguaje (LLM) y su capacidad para aprendizaje contextual ha permitido su extensión desde el texto hasta múltiples modalidades. Surgiendo los modelos largos del lenguaje multimodales (MLLM), cuyas adaptaciones permiten mezclar arquitecturas de una sola modalidad, y con las correctas metodologías de entrenamiento permiten el alineamiento de las modalidades.

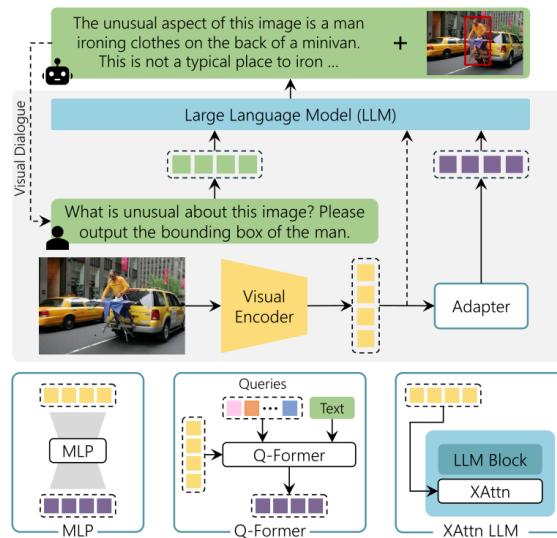


Figura 3.2: Arquitectura general de los MLLMs [Caffagni et al., 2024].

El Modelo largo del lenguaje y vision (LLVM) llamado MoAI (Mixture of All Intelligence) comenzó con la meta de integrar capacidades fundamentales de percepción de diversas funciones cognitivas, utilizando modelos de visión por computadora (CV) para mejorar el entendimiento del mundo real del modelo.

Para incrementar la eficiencia en el uso de esta información, se introdujeron dos módulos-, el Compresor-MoAI y el Mezclador-MoAI, el compresor alinea y condensa las salidas verbales de los modelos de visión por computadora en información visual auxiliar.

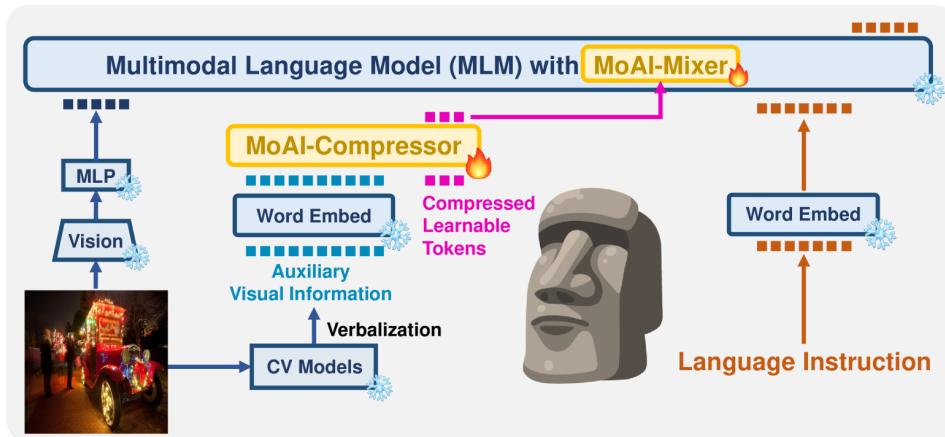


Figura 3.3: Modelo de MOAI [Lee et al., 2024].

Posteriormente, el Mezclador-MoAI emplea módulos de atención cruzada y autoatención para construir seis módulos expertos, usando tres tipos de inteligencia, las características visuales, lingüísticas y auxiliares de los modelos CV para abordar tareas complejas de respuestas a preguntas [Lee et al., 2024].

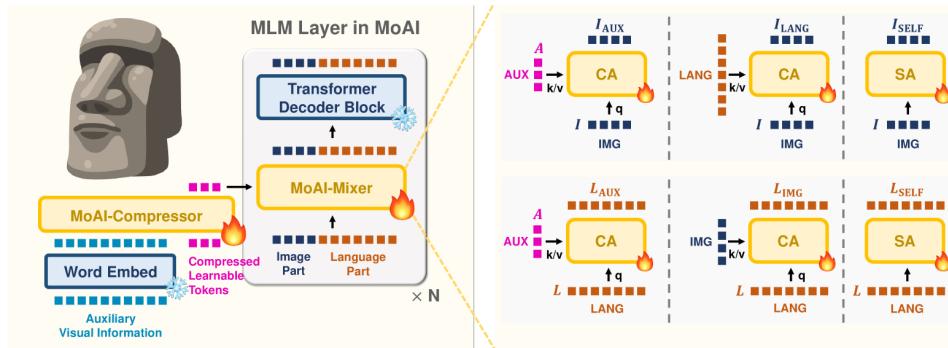


Figura 3.4: MOAI-Mixer [Lee et al., 2024].

Con el deseo de desarrollar un modelo capaz de adaptarse a diversas tareas de visión por computadora, incluso si se cuenta con la mínima o nula afinación para hacerla, es como Florencia fue concebido.

Siendo pionero en la integración de aspectos espaciales, temporales y multimodales en visión por computadora mediante un pre-entrenamiento unificado y una arquitectura de red, es capaz de realizar diversas tareas, pero dependiendo de grandes datasets especializadas para las tareas, teniendo bajo rendimiento al realizar acciones con dos tareas distintas, teniendo llaves duales.

Es por eso que su sucesor, Florence-2 consigue resultados mediante el aprendizaje multitema con anotaciones visuales, resultando en un modelo unificado, con una representación basada en las consultas para una diversidad de tareas de visión, dirigiendo de manera eficaz los retos de estar limitado en información comprensible y la ausencia de una arquitectura ideal.

En vez de utilizar grandes datasets con anotaciones manuales, se basa en una visualmente comprensible a la cual por medio de dos módulos se generan automáticamente anotaciones y pasa por un proceso de refinamiento hasta llegar a ser un resultado confiable, lo que permite acomodar el modelo a una diversidad de tareas de visión sin la necesidad de realizar modificaciones.

Florence-2, diseñado para ser capaz de manejar diversas tareas de visión solamente con un conjunto de pesos y una arquitectura unificada, toma imágenes en conjunto con las consultas de las tareas como instrucciones, y genera el resultado deseado en forma de texto.

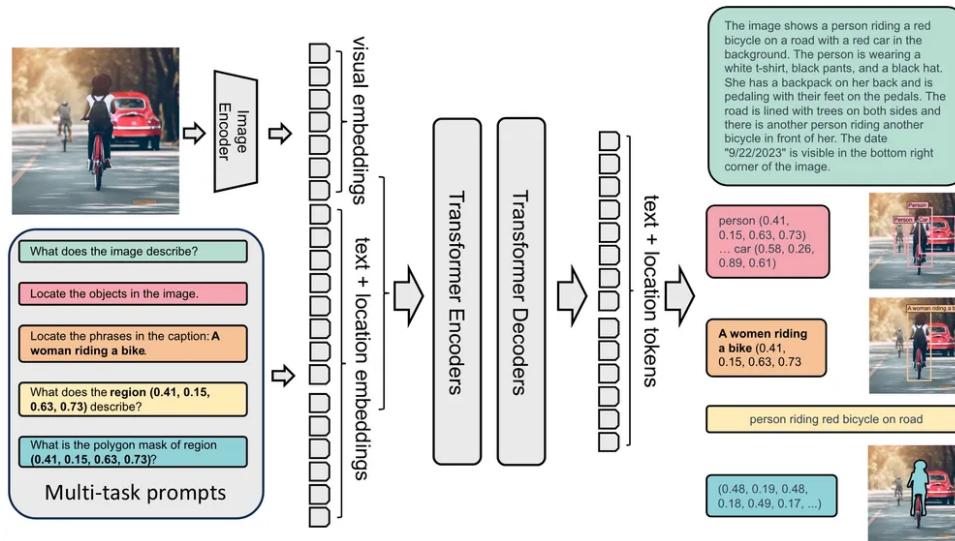


Figura 3.5: Modelo de Florence-2 [Xiao et al., 2023].

El modelo usa un codificador para volver las imágenes a tokens concatenados al texto, que son procesados con un transformador multimodal, obteniendo la respuesta [Xiao et al., 2023].

Por otro lado surgen MLLMs que buscan incorporar nuevas capacidades, como es el caso de LLaVA-OneVision [Li et al., 2024], un modelo de código abierto, siendo un modelo Qwen2 entrenado en información multimodal generada por GPTs, obteniendo la capacidad de funcionar en diferentes modalidades del área de visión por computadora, una simple imagen, múltiples imágenes, y videos.

Para permitir al modelo procesar diferentes resoluciones, Qwen2-VL [Wang et al., 2024b] introduce un mecanismo de resolución dinámica, incrementando las capacidades de percepción visual del modelo.

Por su parte, para los modelos de las series, se busca unificar el entendimiento multimodal con la generación, desacoplando los encoder de cada tarea; Janus-Pro [Chen et al., 2025] es la versión avanzada del modelo base Janus, incorporando una estrategia de entrenamiento optimizada, una expansión en la información de entrenamiento, y escalando el tamaño del modelo.

Con el avance de los VLMs dando un excepcional desempeño con un costo computacional considerable, es que surge la alternativa de VLMs de menor tamaño, aunque realizar las mismas decisiones en su diseño al igual que modelos mas grandes lleva a un uso ineficiente de memoria en la GPU y una limitada practicidad en aplicaciones. Es por lo que surge SmolVLM, una serie de modelos multimodales compactos pensados para la inferencia eficiente en recursos computacionales [Marafioti et al., 2025]. Estos modelos presentan optimizaciones estratégicas en la arquitectura, una tokenización agresiva pero efectiva, así como datos de entrenamiento cuidadosamente cuidados, que llevan mejorar el rendimiento multimodal, facilitando los deployments en escalas signifativamente mas chicas.

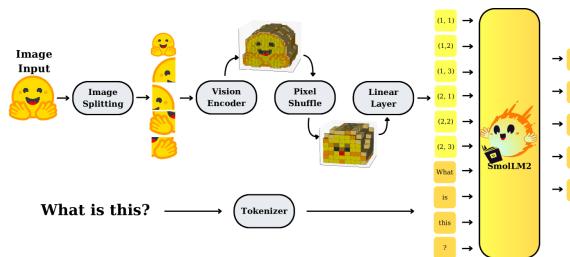


Figura 3.6: Arquitectura propuesta para SmolVLM.

3.3. Weakly Supervised Video anomaly detection (WSVAD)

La búsqueda de la detección semi-supervisado de anomalías en video (WSVAD) ha crecido gracias amplia cantidad de aplicaciones en las que pueda destacar. Como sistemas de vigilancia, de revisión de contenido en videos, análisis de contenido violento, y conducción autónoma. En el área de WSVAD, se busca que el detector de anomalías genere respuesta como puntuajes de anomalías por frame, y se entrene teniendo únicamente anotaciones del video de los frames anómalos.

La mayoría de la investigación en el área apunta a un proceso sistemático de extracción de características usando modelos visuales pre entrenados, para actuar de entrada a un aprendizaje

de múltiples instancias (Multiple instance learning por sus siglas MIL) basado en clasificadores binarios con el propósito de entrenar el modelo, y finalizar con la detección de eventos anormales basados en puntajes de confianza sobre las predicciones de anomalías.

Con el progreso en el desarrollo de modelos pre entrenados de vision-lenguaje (VLP), como CLIP, para el aprendizaje mas generalizado de representaciones visuales con conceptos semánticos, permitiendo alinear imágenes y textos en un entendimiento conjunto. Siendo aplicado a muchas tareas de visión, demostrando un gran potencial para ser adaptado a tareas de detección de anomalías en videos (VAD).

Lo que llevó al desarrollo de VadCLIP, en donde a diferencia del proceso normal del área, se extraen las características del video mediante el encoder de imágenes de CLIP, y se utiliza su encoder de texto de aprovechar la asociación entre imagen y texto.

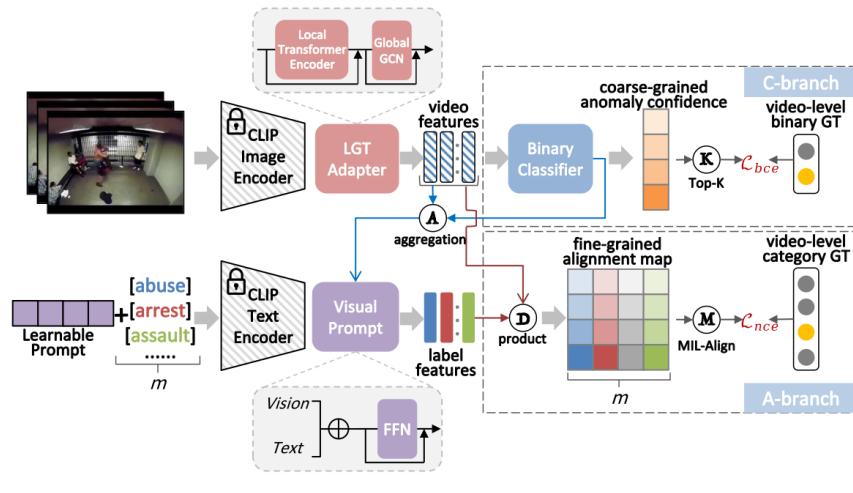


Figura 3.7: La arquitectura de VadCLIP.[Wu et al., 2023]

VadCLIP toma módulos para tratar con las características proveídas por CLIP, así como dos ramas, para llevar a cabo tareas de detección e identificación de anomalías, siendo esta última la que toma las características extraídas de los textos y posibles categorías de anomalías para seleccionar el tipo de anomalía presentada.

Entre las datasets destacadas en esta área se encuentra XD-Violence [Wu et al., 2020] 24 FPS También está UCF-Crime [Sultani et al., 2019]

3.4. MLLMs en WSVAD

El área de detección de anomalías en videos ha presentado avances mediante la integración de modelos largos del lenguaje (LLM) y modelos de visión-lenguaje (VLM), enfocándose en retos críticos como interpretabilidad, razonamiento temporal, y generalización en escenario de mundo abierto dinámicos. Los retos que enfrenta el área de VAD se deben a la naturaleza dinámica de los videos, la complejidad de detectar las anomalías en varios contextos, y la dificultad de obtener información catalogada para entrenar modelos robustos [Ding and Wang, 2024]

Al contar los LLMs y VLMs con un entendimiento profundo del contexto visual y textual de los videos, ofrece nuevas posibilidades para detectar y explicar anomalías, pudiendo captar dependencias temporales, entender las relaciones contextuales, e incluso general descripciones textuales del contenido de los videos, los hace una herramienta versátil para mejorar la detección de anomalías en escenarios de mundo abierto, en el mundo real. [Ding and Wang, 2024]

Para balancear la resolución temporal, la eficiencia computacional y el rendimiento general del modelo, es que se crean las estrategias de muestreo de fotogramas. En la figura 3.8 se presentan las estrategias más comunes, la decisión de cuál usar debe alinearse con la naturaleza de las anomalías y las restricciones de operación.

TABLE I: Comparison of different sampling strategies for temporal reasoning.

Sampling	Interval	Frame count	Redundancy	Target use case	Cost
Uniform	Fixed	Medium	Medium	Global trend	High
Random	Random	Medium	Low	Data augmentation	High
Key frame	Adaptive	Low to Med.	Low	Key event extraction	Medium
Dense	One	High	High	Fine-grained modeling	Low
Sliding window	Adaptive	Medium	Medium	Local temporal details	Medium
Adaptive	Dynamic	High	Low	Comprehensive modeling	Medium

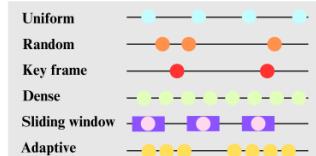


Fig. 2: Various sampling strategies.

Figura 3.8: Estrategias de muestreo para los MLLMs [Ding and Wang, 2024].

El uso de los LLMs y VLMs en VAD se ha centrado en su integración para cuatro puntos principales en los que han existido avances: modelado temporal y contexto (buscando capturar las dinámicas temporales mientras se mantiene la eficiencia computacional y escalabilidad), interpretabilidad y transparencia (buscando generar sistemas de detección de anomalías más comprensibles para los usuarios finales), detección libre de entrenamiento o few shot (busca usar modelos pre entrenados y las mínimas anotaciones para facilitar la detección de anomalías en entornos escasos de información) y detección de mundo abierto (Buscando sistemas capaces de detectar anomalías que no se han visto y adaptarse a escenarios impredecibles).

Entre los LLMs y VLMs utilizados en VAD se encuentran CogVLM-17B en AnomalyRuler [Yang et al., 2024], Video-LLAMA en VADor [Lv and Sun, 2024] y CALLM [Ntelopoulos and Nasrollahi, 2024]

, Video-LLaVA-7B en Holmes-VAD [Zhang et al., 2024], Llama-2-13b en LAVAD [Zanella et al., 2024] y HAWK [Tang et al., 2024], y InternVL2, de 2B para Holmes-VAU [Zhang et al., 2025] y 8B para VERA [Ye et al., 2025].

Entre los enfoques en que se utilizan esta Holmes-VAU [Zhang et al., 2025], enfocado en el área de Entendimiento de anomalías en videos (VAU), involucrando VAD y explicabilidad de las anomalías. Como se observa en la figura 3.9, su propuesta consiste de procesar un numero de fotogramas continuos por un codificador visual, para obtener los tokens visuales, que posteriormente pasaran por un algoritmo de muestreo, que es una red VAD basada de características que devuelve un puntaje de anomalía para seleccionar aquellos fotogramas clave. Los tokens correspondientes de los fotogramas de muestra se mapean con un proyector hacia el espacio de características del lenguaje como embeddings visuales, para concatenarlos con los embeddings textuales para procesarlos por el LLM y obtener la salida de entendimiento de las anomalías.

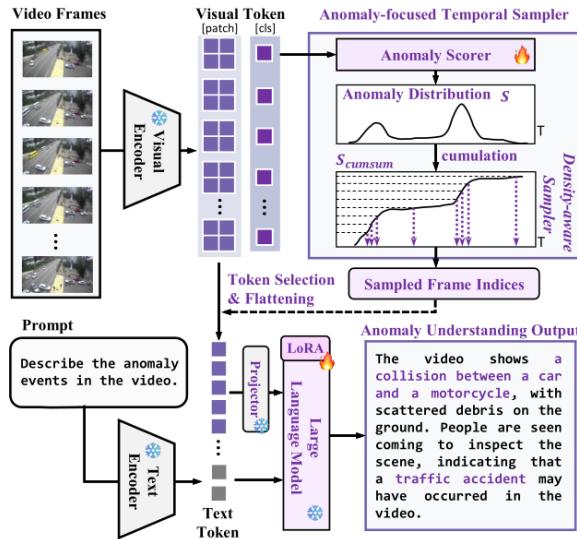


Figura 3.9: Arquitectura de Holmes-VAU [Zhang et al., 2025].

Así como AnomalyRuler [Yang et al., 2024], que buscando incrementar las capacidades de los LLMs en el area de VAD, presenta un framework de razonamiento basado en reglas para VAD con LLMs. Ya que el conocimiento implícito dado por el pre entrenamiento de los LLMs se enfoca en el contexto general, podría no aplicarse para todos los escenarios específicos de VAD, llevando a una inflexibilidad y inexactitud, se busca un enfoque sin la necesidad de un entrenamiento completo que permite su rápida adopción en varios escenarios de VAD. Ya que preguntar directamente al LLM puede no alinearse con las necesidades específicas del área de VAD, es propone un enfoque de flexible en los prompts que dirige a las fortalezas de razonamiento de los LLMs para diferentes usos de VAD.

Dando como resultado un proceso de consiste de inducción y deducción, el LLM procesa descripciones visuales de un VLM sobre algunos fotogramas de referencias para regresar un conjunto de reglas robustas que determinan la normalidad. En la etapa de deducción se siguen las reglas determinadas para identificar fotogramas anómalos en las secuencias de videos mediante las descripciones dadas por el VLM. Este proceso de puede observar en la figura 3.10.

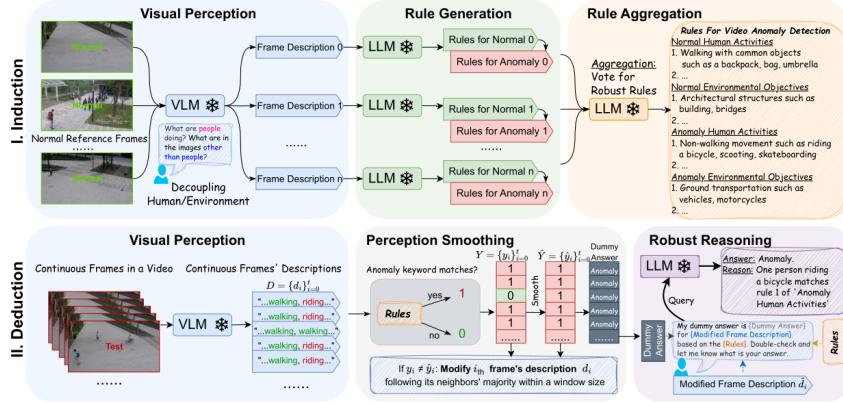


Figura 3.10: Pipeline de AnomalyRuler.

Capítulo 4

Banco de pruebas

4.1. Hardware

El equipo de computo utilizado durante la fase de desarrollo y evaluación cuenta con los siguientes componentes:

- **Procesador:** AMD Ryzen 7 7840HS w/ Radeon 780M Graphics
- **GPU:** NVIDIA GeForce RTX 4060 Max-Q / Mobile
- **RAM:** 16GB

4.2. Software

El software utilizado en el equipo de computo cuenta con las siguientes características:

- **Sistema Operativo:** Ubuntu 24.04.2 LTS
- **Versión de Python:** Python 3.10.14
- **Bibliotecas y frameworks:**
 - **PyTorch:** Versión 2.1.2+cu121
 - **Transformers:** Versión 4.45.0.dev0
 - **Janus**
 - **YOLOv10:** Versión 1.0
- **Otras herramientas:** CUDA 12.2
- **Modelos preentrenados:**

- LLaVa-OneVision-0.5b-ov
- Janus-Pro-1B
- Qwen2-VL-2B-Instruct
- YOLOv10x

4.3. Métodos

Con la finalidad de probar el impacto de los componentes incorporados en la arquitectura propuesta, estos serán evaluados con diversas combinaciones, tal como se presenta en la tabla 4.1.

Metodo	Componentes			
	MLLM	Detector	Reglas	Info. Detecciones
MMLM Solo	✓			
Detector con reglas		✓	✓	
Detector, MLLM e información	✓	✓		✓
Detector con reglas y MLLM	✓	✓	✓	
Detector con reglas, MLLM e información	✓	✓	✓	✓

Tabla 4.1: Métodos de arquitectura y componentes que la integran.

Contando así con los métodos considerados en la evaluación, presentando en la tabla 4.2 dos nuevos métodos a evaluar.

Método	Componentes		
	CLIP	Reglas	MLLM
CLIP Solo	✓		
CLIP con reglas	✓	✓	
CLIP con reglas y MLLM	✓	✓	✓

Tabla 4.2: Métodos de arquitectura con CLIP (el componente reglas incluye el detector así como MLLM con información).

4.4. Datasets

4.4.1. CHAD

Charlotte Anomaly Dataset es una dataset del área de detección de anomalías en videos (VAD), enfocada en la vigilancia. Incluye 412 videos, resultando en un aproximado de 1.15

millones de fotogramas, de los cuales 1.09 millones son normales y 59 mil son anómalos.

Los videos son provenientes de cuatro cámaras distintas de alta resolución (tres Full-HD y una HD), que dan cuatro puntos de vista de las escenas, cada cámara cuenta con un numero identificador que viene escrito en el nombre de cada video.



Figura 4.1: Vistas de las cámaras [Danesh Pazho et al., 2023].

La dataset incluye el etiquetado de los fotogramas de los videos, indicando si son anómalos o normales. Los comportamientos considerados anómalos en los videos son su totalidad de acciones humanas.

Group Activities		Individual Activities	
Fighting	Punching	Throwing	Running
Kicking	Pushing	Riding	Falling
Pulling	Slapping	Littering	Jumping
Strangling	Body Hitting	Hopping	Sleeping
Theft	Pick-Pocketing		
Tripping	Playing with Ball		
Chasing	Playing with Racket		

Figura 4.2: Eventos anómalos de CHAD [Danesh Pazho et al., 2023].

4.4.2. IITB-Corridor

IITB-Corridor es una dataset de videos de actividad anómala de personas, en la cual los videos fueron grabados por una única cámara Full-HD con vista a lo largo de un pasillo. Los videos son de múltiples eventos anómalos, apreciados en la figura 4.3, mientras que las actividades como caminar y estar parado son consideradas normales.



Figura 4.3: Ejemplos de los videos y sus anomalías identificadas [Rodrigues et al., 2020]

Cuenta con un total de 108,278 fotogramas anómalos en total de los 181,567 fotogramas anotados.

4.4.3. NWPU Campus

NWPU Campus es una dataset propuesta para el área de detección de anomalías en videos (VAD), especialmente para la modalidad semi-supervisada (WSVAD). Cuenta con 43 escenas, con 28 tipos de eventos anómalos apreciados en la figura 4.4, entre los cuales una mayoría sobre personas, mientras el resto se enfocan principalmente en vehículos.

Table 1. The list of anomaly classes in NWPU Campus dataset. "s.d." stands for a scene-dependent anomaly.			
Climbing fence	Car crossing square	Cycling on footpath (s.d.)	Kicking trash can
Jaywalking	Snatching bag	Crossing lawn	Wrong turn (s.d.)
Cycling on square	Chasing	Loitering	Scuffle
Littering	Forgetting backpack	U-turn	Battering
Driving on wrong side	Falling	Suddenly stopping cycling in the middle of the road	Group conflict
Climbing tree	Stealing	Illegal parking	Trucks (s.d.)
Protest	Playing with water	Photographing in restricted area (s.d.)	Dogs

Figura 4.4: Eventos anómalos en los videos [Cao et al., 2023].

Cuenta con 318,793(3.54h) fotogramas normales y 65,266(0.73h) fotogramas anómalos, con anotaciones que indican su estado. Los videos son de resoluciones variadas.



Figura 4.5: Ejemplos de los escenas y sus eventos anómalos de NWPU Campus dataset [Cao et al., 2023]

4.4.4. Avenue Dataset

Avenue Dataset es una dataset de videos, los videos cuentan con resolución 640x360 y presentan una escena de una avenida, en la cual suceden 14 tipos de eventos anómalos. Cuenta con 37 videos y su conjunto de prueba presenta anotaciones por fotograma. [Lu et al., 2013]

4.4.5. Distribución de los videos

La principal dataset utilizada durante el desarrollo y la evaluación de la arquitectura es CHAD [Danesh Pazho et al., 2023], considerando los eventos que se aprecian en la figura 4.2, y realizando una evaluación, en la cual eventos similares fueron combinados en uno solo, siendo el caso mas notorio en el que todos los eventos asociados a pelear, quedaron en el propio evento de pelea, resultando en 13 eventos únicos, presentados en la tabla 4.3

Eventos individuales	Eventos grupales
Andar en bicicleta	Pelear
Correr	Jugar
Estar acostado	Perseguir otra persona
Saltar	Guinar a otra persona
Tirar basura	Robar a otra persona
Caerse	Carterismo
Tropezar con otra persona	

Tabla 4.3: Eventos anómalos en la dataset

Entre los eventos se encuentran algunos claramente diferentes a los demás, como andar en bicicleta, cuyo parecido a los demás es mínimo. Por su parte, se encuentran aquellos eventos

que se relacionan directamente con otros, como correr, que es la parte del evento perseguir a otra persona, básicamente un evento individual es parte de un evento grupal, que cuenta con mayor complejidad. Bajo la idea anterior, es que un video puede pertenecer a dos eventos distintos, por lo que aunque la dataset CHAD, cuente con 134 videos anómalos, es que se usan como 154 por los duplicados, dando un total de 67.3k frames anómalos y 77.6k frames normales.

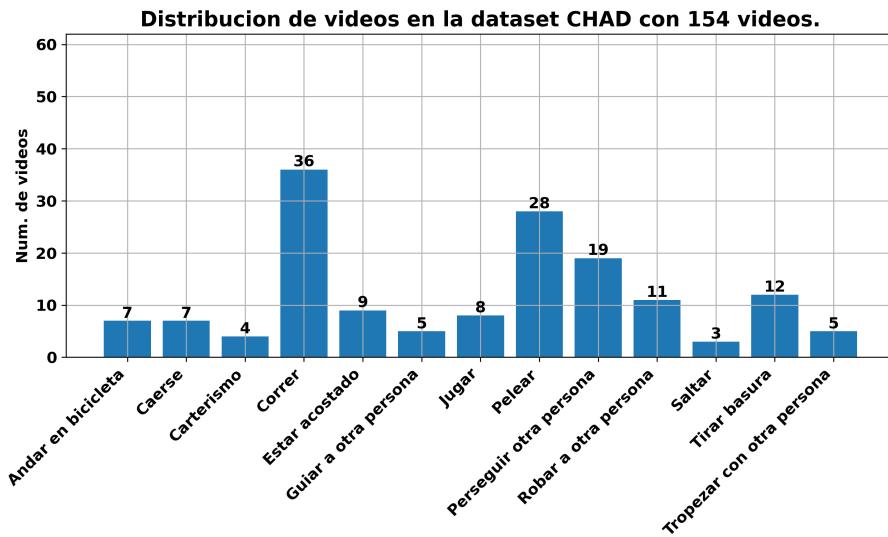


Figura 4.6: Videos de CHAD entre los 13 eventos.

Para complementar CHAD, se usaron las dataset de IITB Corridor y NWPU campus, seleccionando los videos que pertenezcan a los eventos previamente seleccionados, juntando un total de 51.4k frames anomalos y 47.2k frames normales.

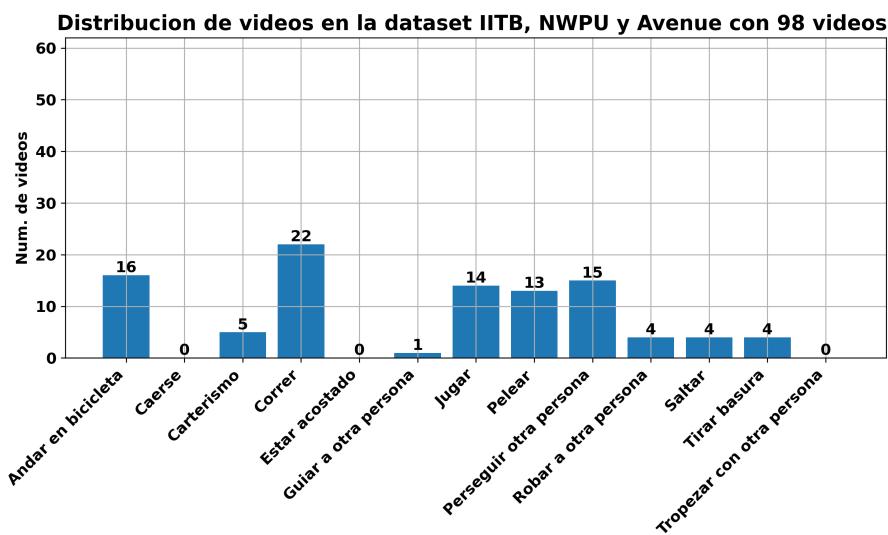


Figura 4.7: Distribucion de los 98 videos de las demás datasets.

Dando un total de 252 videos con 118.8k frames anomalos y 124.8k frames normales.

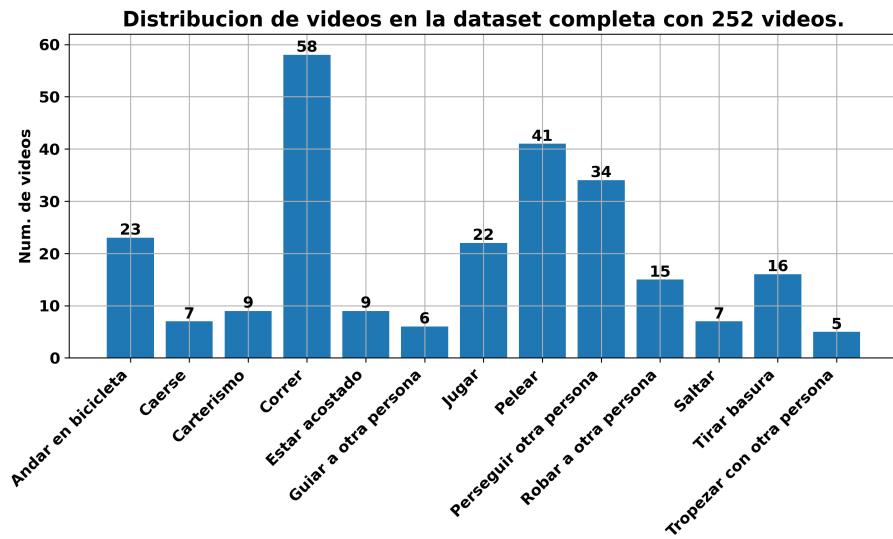


Figura 4.8: Distribucion de los 252 videos de las demas datasets.

Capítulo 5

Metodología

Una vez preparado el equipo de cómputo y definidos los métodos a utilizar y datasets, se procede a describir la metodología empleada para el desarrollo de la arquitectura.

Entre los componentes de la arquitectura se encuentran aquellos proveídos de repositorios oficiales, por otro lado algunos elementos fueron desarrollados para apoyar al desempeño de la arquitectura, evaluando para cada uno de ellos su función individual, forma de ejecución y su aporte en la arquitectura.

A continuación se detallan los elementos desarrollados así como una descripción de los métodos previamente establecidos.

5.1. Algoritmos propuestos

Durante el desarrollo de la arquitectura, se observó la necesidad de algoritmos que permitan ayudar en la detección de eventos y la reducción del uso de la validación, así como ayudar en la conexión entre componentes de la arquitectura, a continuación se describirá cada uno de ellos

5.1.1. Espaciado temporal entre fotogramas y limitaciones en la cantidad.

Debido al uso de las detecciones brindadas por el detector en tiempo real, es recomendable que los fotogramas para evaluación estén espaciados temporalmente entre ellos, ya que las detecciones cambian significativamente y permite una mejor evaluación de la información intrínseca de estas. Siendo una estrategia de muestreo inspirada en el Sliding window de la figura 3.8, concentrándose en secciones locales del video.

Por lo que el algoritmo propuesto guarda un fotograma cada cierto numero junto a sus detecciones, siguiendo un orden temporal; el espaciado temporal entre frames es arbitrario, en este trabajo se decidió seleccionar 6 fotogramas cada segundo, dando un espaciado de 4 para videos de 30 fps. Para asegurar un numero constante de fotogramas guardados, es que se coloca un limite, tras el cual cada fotograma nuevo es guardado, eliminando el mas antiguo en el proceso, el limite es arbitrario, para este trabajo establecido en 6, ya que con el espaciado previamente mencionado al termino del primer segundo, se cuenta con suficientes fotogramas para realizar una evaluación sobre la posibilidad de un evento.

La aportación de esta acción es limitar los fotogramas procesados por la validación por parte de los MLLMs, contando con un tiempo de ejecución constante, así como brindar información al algoritmo de decisión que facilite el modelado de las reglas que se basen en cambios temporales de las detecciones.

5.1.2. Organizador de detecciones de personas

Al contar con los fotogramas organizados con sus correspondientes detecciones, surge la problemática de identificar las personas, ya que con el espaciado temporal, el cambio puede ser abrupto entre detecciones, por lo se busco un algoritmo capaz de identificar las personas con sus correspondientes detecciones y retornarlas de forma organizada. Para lo cual el algoritmo requiere recibir todas las detecciones de los fotogramas organizados provenientes del anterior algoritmo; partiendo del primer fotograma, compara las detecciones con los siguientes, obteniendo el mas cercano a su centro (delimitando una distancia máxima aceptable, igual al valor máximo entre el ancho y largo para evitar problemas por las vistas, evitando detecciones lejanas), para en ese caso proseguir y organizar las detecciones en listas, para finalmente ser devueltas.

Su principal función es apoyar en el modelado y ejecución de las reglas, permitiendo un orden claro al momento de extraer información intrínseca de las detecciones progresivas y decidir la presencia de un evento.

5.1.3. Reglas en base a las detecciones

Ya que los fotogramas organizados y limitados a un numero constante, así como sus detecciones, son utilizadas como punto de partida para el algoritmo de decisión basado en reglas de los eventos.

La información utilizada por las reglas proviene de las detecciones, en concreto de las bounding boxes, el puntaje de confianza, y la clase correspondiente. Esta información permite obtener una decisión sobre la posibilidad de un evento en las escenas, existiendo diversos procesamientos de información para conseguirlo, dividiendo así las reglas en los siguientes grupos

5.1.3.1. Presencia de una clase específica

Todos los algoritmos de decisión parten de un proceso de verificación sobre las clases de interés, en caso de no existir ninguna instancia de interés o con poco puntaje de confianza, es automáticamente rechazada la existencia del evento. Por lo general la clase de interés sera el de las personas, sin embargo para este tipo de reglas se requiere la existencia de clases específicas, como en el caso de 'andar en bicicleta', que requiere mínimo una detección de 'bicicleta', o el evento grupal 'jugar', que requiere las clases 'balón', 'frisbee' o afín. Posteriormente, se necesita verificar la interacción entre los objetos con las personas, con reglas, como estar tocando bounding boxes en las escenas.

Al ser diferentes a los demás eventos, estas son las reglas mas simples, siendo de la mayor importancia la existencia de una instancia específica para indicar la posibilidad del evento.

5.1.3.2. Cambios de la bounding box de un objeto a lo largo del tiempo

El siguiente tipo de reglas se basa en los cambios de las bounding boxes de las detecciones organizadas, basándose en sus valores de ancho, largo y posición a lo largo del tiempo. Principalmente se encuentran eventos individuales, donde cada objeto es tratado de esta forma.

Entre la posible información obtenible de las detecciones se encuentra el movimiento de los objetos, para el cual se ha utilizado como métrica la área compartida entre detecciones consecutivas, para evitar los posibles problemas con las vistas de las cámara, un valor promedio bajo indicaría un gran movimiento, mientras uno elevado indicaría un movimiento estático en el video, siendo las reglas para los eventos 'correr' y 'estar acostado' respectivamente.

También se puede obtener como varia el ratio entre el ancho y el alto de las detecciones, para indicar cambios frenéticos, como en el evento de 'tirar basura', ya que los videos de esta acción se observa un repentino movimiento de brazos que regresa a su estado normal. Para el caso del evento 'caerse', se usa el mismo ratio, pero ya que el cambio de estar parado a tumbado en el suelo incrementa este ratio y no regresa a su estado anterior, se utiliza como regla del respectivo evento.

Por ultimo, se utiliza la información relacionada a los cambios de las bounding boxes para determinar la dirección en la que los movimientos son realizados, como en el caso del evento 'saltar', en donde se comprueban los movimiento en el eje Y donde se realizan los saltos con la cámara horizontalmente, verificando las diferencias súbitas entre las detecciones.

5.1.3.3. Interacción entre bounding boxes de dos objetos a lo largo del tiempo

Basándose en los anteriores tipos de reglas, este grupo de reglas se centra en la relación entre dos diferentes objetos, siendo en su totalidad reglas de eventos grupales, obteniendo la información de dos objetos distintos que son comparados.

Entre las reglas se encuentran aquellas de cercanía entre objetos, como es el caso de 'pelear', que verifica que las bounding boxes de ambos objetos se toquen, o 'tropezar con otra persona', que realiza lo mismo, pero ademas incorpora una regla individual, en donde uno de las dos personas cercanas debe de contar con movimientos súbitos en su bounding box.

Existen aquellas que toman parte de las reglas individuales para funcionar, como 'perseguir', que verifica que alguna persona este primero corriendo para luego evaluar que la distancia de esta persona con otra se este reduciendo, siendo una subclase de otra regla; así como 'robar a otra persona' es subclase de perseguir, ya que ademas de correr y acercarse a la persona, debe de llegar a tocarla en algún punto del tiempo.

Otras reglas aprovechan concepto de las anteriores, como 'guiar a otra persona', que utiliza la propiedad de dirección de 'saltar', para evaluar si ambas personas caminan al mismo sentido y ademas se encuentran cercanas, al verificar el toque entre estas; así como 'carterismo' aprovecha los cambios súbitos de tropezarse, pero con movimientos rápidos que se acerca y alejan de la otra detección.

5.1.4. Elaboración del prompt

La plantilla estándar que reciben los MLLMs es escrita en inglés, dejando secciones en las que se agregara información dependiendo del método, siendo 'text Does the video contain event? Just yes or no'; en esta plantilla se puede apreciar un texto inicial, el cual puede proveer de información al modelo, así como un texto intermedio indicando que evento es de interés para buscar, en la ultima parte se pide al modelo que regrese solamente una respuesta entre si y no, indicando si es correcto lo observado en los fotogramas procesados.

Para el texto inicial existen diversos prompts elaborados, la base mas simple establece 'Watch the video.', indicando el enfoque en los fotogramas otorgados, utilizado principalmente en los métodos observados en la table 4.1 que no contienen información, en los demás se indica el numero de objetos observados en el ultimo fotograma, siempre que sean de las clases de interés para los eventos, siendo una breve sentencia en ingles, por ejemplo 'There are 2 persons in the video.'

También se desarollo un prompt detallado, en el cual ademas de indicar el numero de los objetos de las clases de interés, se indica la ubicación de los mismos en las 9 secciones, dividiendo el eje Y como 'Top', 'Middle', 'Bottom' y del eje X como 'Left', 'Center' y 'Right'.

5.2. Arquitectura propuesta

Con los componentes establecidos, así como algoritmos auxiliares, se idearon formas de conectarlos e interactuar entre ellos, surgiendo así la arquitectura propuesta, la cual puede albergar los diferentes métodos observados en la tabla 4.1. La arquitectura completa se aprecia en la siguiente imagen

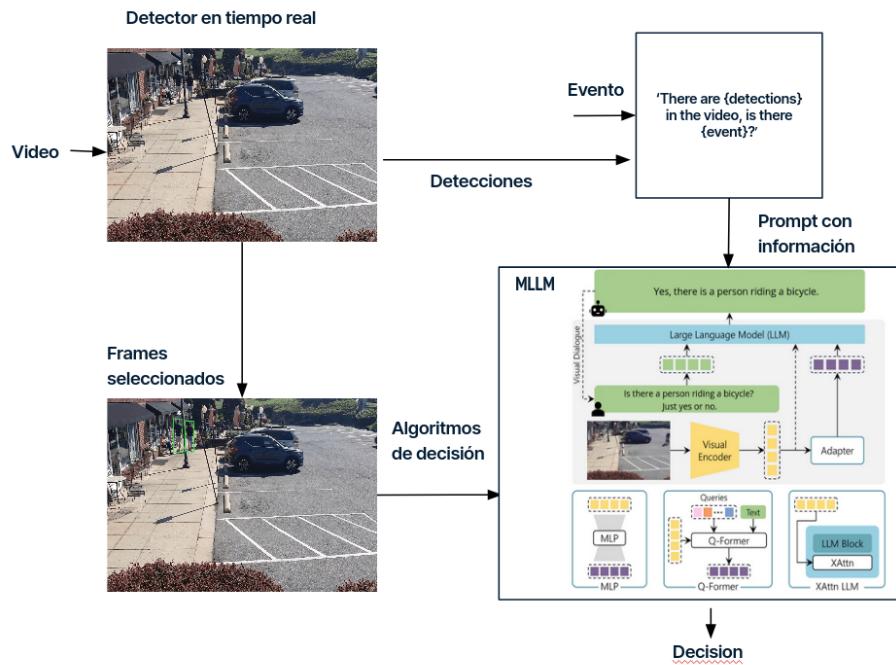


Figura 5.1: Vista general de la arquitectura, incluyendo todos los elementos [Caffagni et al., 2024].

5.3. Descripción de los métodos

Para comprender cada uno de los métodos establecido por la tabla 4.1 y la el impacto de los componentes en comparación con los demás, se presentaran cada uno de ellos, con su respectivas características.

5.3.1. MLLM Solo

El primero de los métodos es el uso del MLLM de forma aislada, para evaluar su capacidad para identificar y determinar un evento específico en un video. El modelo recibe un numero establecido de fotogramas, así como la plantilla básica, sin información extra mas que el evento a validar, retornando una respuesta entre si o no.

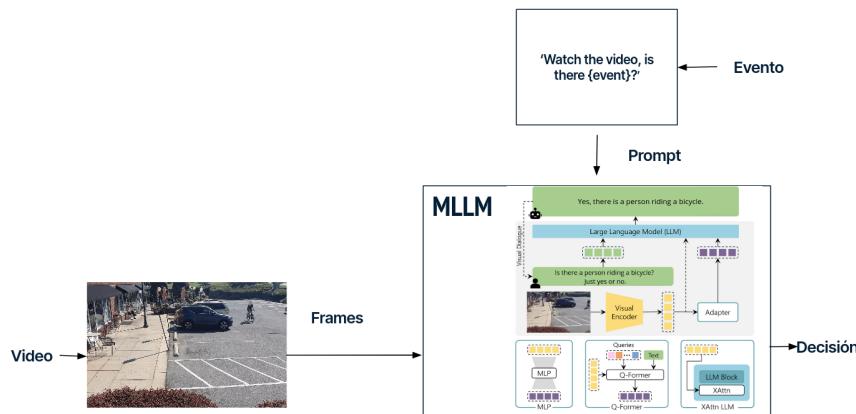


Figura 5.2: Uso del MLLM para la detección de anomalías en videos.

Este método pide de un uso constante del MLLM, el elemento mas lento de la arquitectura, resultando en un punto de comparación con los demás elementos reducir la latencia.

5.3.2. Detector con reglas

En este método se evalúan las reglas basadas en las detecciones descritas anteriormente, tomando la información de los fotogramas, y regresando una respuesta sobre la posible presencia del evento.



Figura 5.3: Uso de las reglas para la detección de anomalías en videos.

Este método sirve principalmente para probar las reglas desarrolladas y compararlas, dando un punto de comparación, ya que es un método del que se espera baja latencia con poca precisión, presenta la contraparte del método anterior.

5.3.3. Detector, MLLM e información.

En este método el detector es incorporado al metodo 'MLLM solo' para brindar información de los objetos vistos en las escena de los videos, usando el algoritmo encarga de los prompts para incluirla en la plantilla que recibe el MLLM.

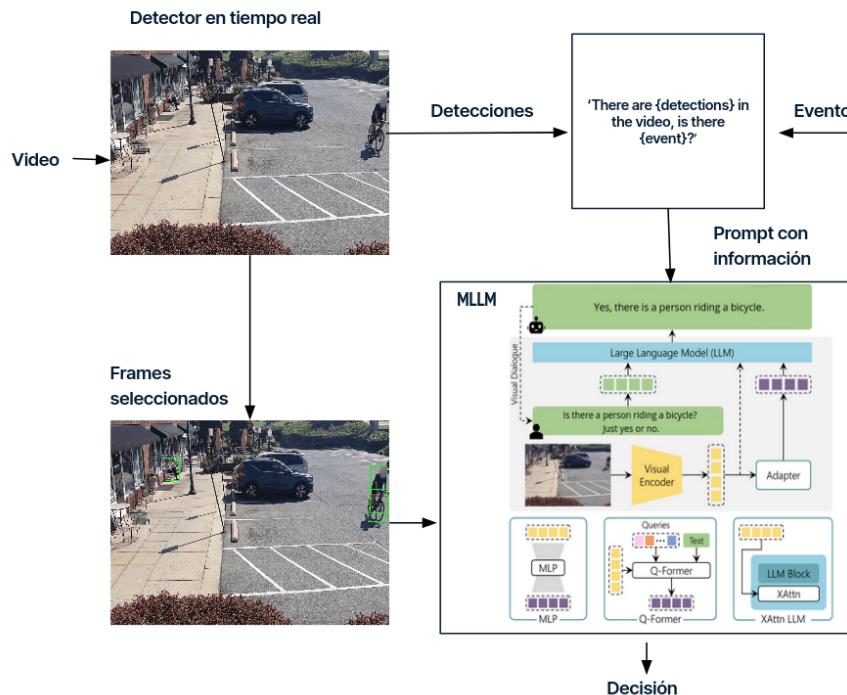


Figura 5.4: Uso del MLLM con información en los prompts para la detección de anomalías en videos.

En este método se busca verificar el impacto de la información en el prompt del MLLM, comparando principalmente con su contraparte sin información, y buscando un análisis sobre la viabilidad de un decremento en la latencia por uso del detector y un prompt mas largo para incrementar la precisión.

5.3.4. Detector con reglas y MLLM

Para este método, las detecciones son procesadas mediante las reglas, indicando la posibilidad de un evento, para posteriormente proseguir con el proceso de validación por medio del MLLM, confirmando o negando su estatus.

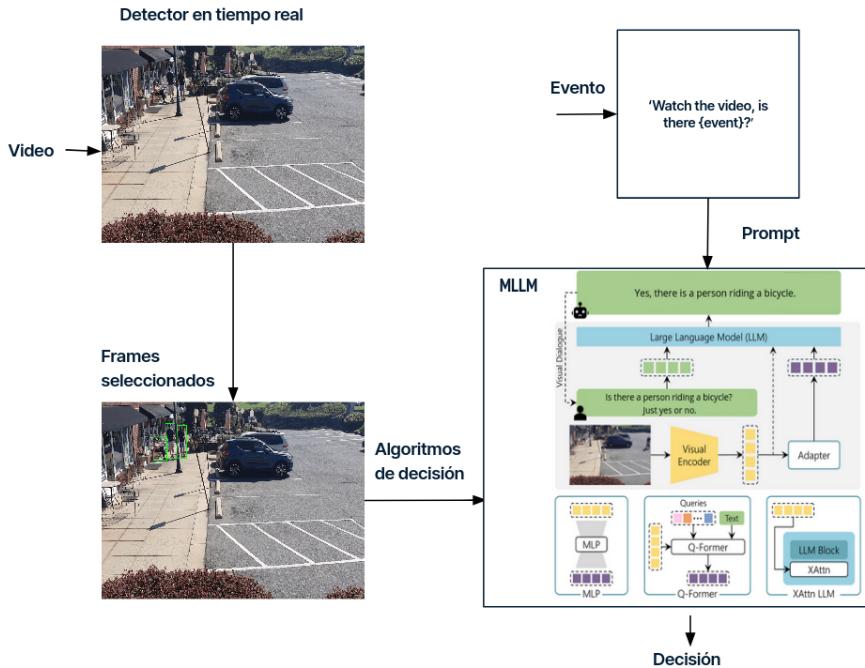


Figura 5.5: Uso del MLLM y las reglas de forma conjunta para la detección de anomalías en videos.

Con este método se busca llevar limitar el uso del MLLM, al requerir el llamado de las reglas para detectar un posible evento, permitiendo una latencia fluida en la mayoría de fotogramas normales, mientras se reducen los falsos positivos que las reglas pudieran ocasionar.

5.3.5. Detector con reglas, MLLM e información

Por ultimo, este método abarca la arquitectura completa previamente observada en la figura 5.1, el cual realiza el mismo procedimiento que el anterior método, pero incluyendo información en el prompt que recibe el MLLM, con el propósito de evaluar su impacto en el uso de reglas y MLLM.

5.4. Uso de CLIP

Para ejecutar los componentes en tiempo real, sin especificar el evento a buscar dentro de las escenas, se incorpora el modelo CLIP [Radford et al., 2021], que recibiendo las descripciones de los todos los eventos contemplados en la tabla 4.3 junto a los fotogramas elegidos, permite identificar el evento presente en los videos. El modelo utilizado sera clip-vit-base-patch16, al igual que en el modelo VadCLIP [Wu et al., 2023], presentando una ventaja en el uso en tiempo real al ser el segundo modelo de menor tamaño de su serie y al haber presentado previamente su potencial en el área de VAD. Ademas se detallaron las descripciones de los eventos para aprovechar la capacidad de CLIP para establecer asociaciones robustas entre contenido visual y texto.

Como se observar en la tabla 4.2, el primer método describe al uso en solitario de CLIP, mientras los otros dos integran componentes de la arquitectura para incrementar las capacidades de CLIP, que serán descritos a continuación.

5.4.1. CLIP con reglas

En este método se hace uso del detector con las reglas, determinando los posibles eventos que pudieran estar presente en las escenas, las descripciones de los eventos elegidos son procesadas con el conjunto de fotogramas guardados, y retorna el evento cuya probabilidad promedio sea mayor. Entre las descripciones se agrega la referente a la normalidad, siendo ajena a cualquier anomalía.

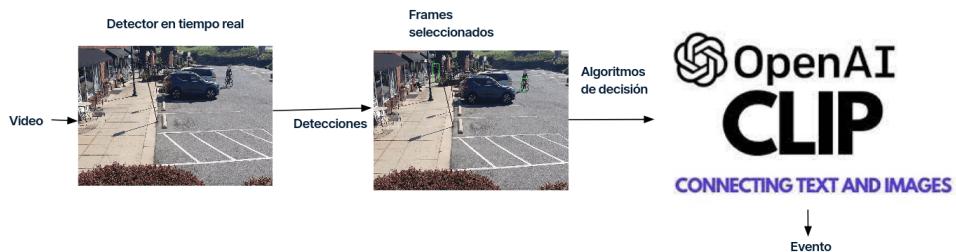


Figura 5.6: Uso del detector y las reglas para detectar anomalías en videos con CLIP

Este procedimiento permite reducir el numero de posibles eventos que puede elegir CLIP, para evaluar como mejora la precisión de CLIP la incorporación de componentes de la arquitectura.

5.4.2. CLIP con reglas y MLLM

Partiendo del método anterior, posteriormente de la elección de CLIP del evento observado en las escenas, se incorpora un proceso de validación mediante un MLLM, buscando la confirmación de la presencia del evento.

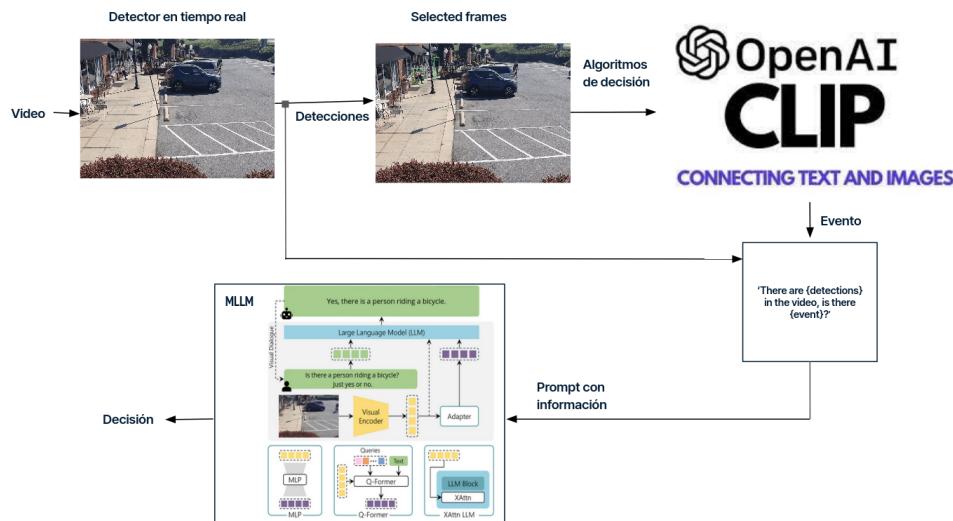


Figura 5.7: Uso de todos los componentes de la arquitectura detectar anomalías en videos con CLIP.

Con este agregado se espera una reducción en los falsos positivos por parte de CLIP a costa de un incremento en la latencia. Con este método se busca muestra la flexibilidad de la arquitectura, y como sus componentes se pueden re-acomodar para permitir la incorporación de nuevos agregados.

5.5. Evaluación utilizada

Cada uno de los videos sera procesado y sus predicciones evaluadas, para utilizar la métrica Average Precision (AP), evaluando por cada uno de los eventos, y promediado para resultar en mAP (Mean Average Precision).

Capítulo 6

Experimentación y resultados

6.1. Experimentos realizados

Buscando analizar el desempeño de la arquitectura, se realizaron diversos experimentos con su propio propósito específico, a continuación se explicara cada uno de ellos.

6.1.1. Selección del MLLM

Los primeros experimentos fueron enfocados a evaluar MLLMs en aplicaciones de detección de anomalías en videos (VAD), determinando de criterios de selección para los posibles candidatos

- Modelos de código abierto recientes
- Modelos con un numero pequeño de parámetros
- Compatible con la librería Transformers
- Modelos con enfoque en capacidades visuales.

Reduciendo la selección a los tres candidatos, descritos a continuación

- LLaVA-OneVision: Lanzado en 2024, siendo de código abierto y contando con la versión LLaVA-OV-0.5B de 0.5 billones de parámetros, contando con un fine-tuning especializado en imágenes y videos, es el modelo mas chico de la selección, [Li et al., 2024].
- Janus: Perteneceiente a una serie de modelos que busca unificar el entendimiento visual con la generación y liberado en 2025, el modelo Janus-Pro-1.3B [Chen et al., 2025] presenta una estrategia de entrenamiento optimizada que busca diferenciarlo de sus predecesores.

- Qwen2-VL: Siendo una serie de modelos de código abierto lanzados en 2024, su modelo Qwen2-VL-2B [Wang et al., 2024b] cumple con los requisitos, siendo el modelo de mayor numero de parámetros.

Para evaluar los modelos se utilizo únicamente el método 5.2 para ejecutar únicamente el MLLM sin componentes externos. Se procesaron los videos de la dataset CHAD, con sus respectivos eventos y con cada uno de los modelos planteados.

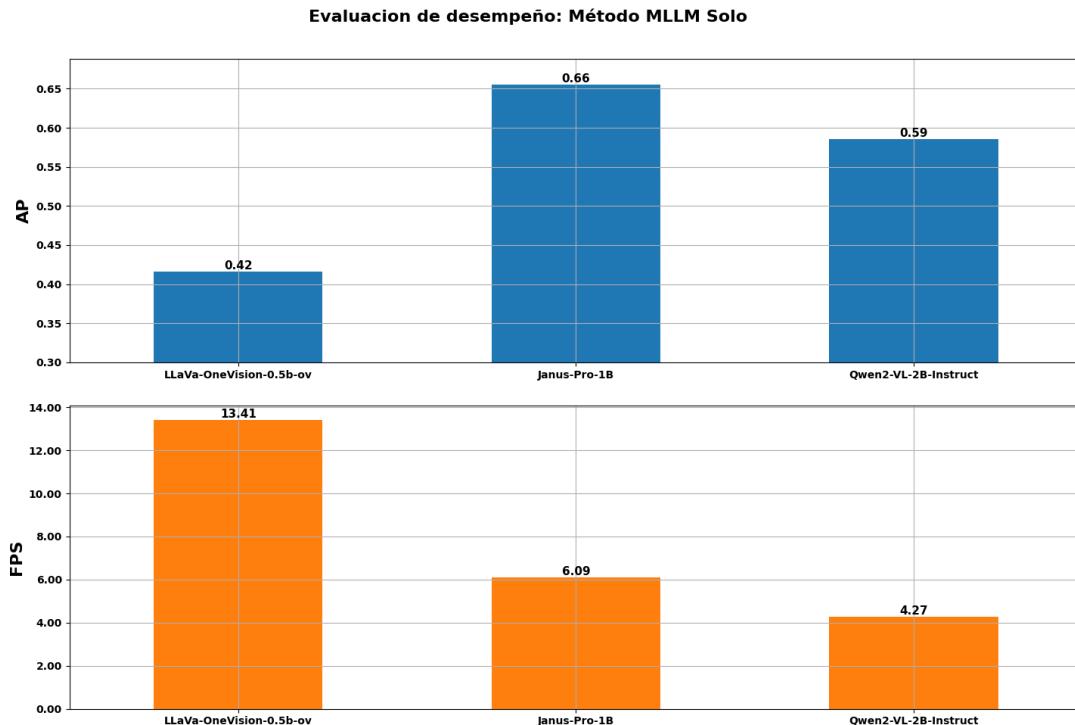


Figura 6.1: Average Precision y FPS de cada MLLM.

Como se puede apreciar en la figura 6.1, el AP de mayor valor corresponde al modelo Janus, con un porcentaje 57.14 % mayor al modelo de LLaVa y ganando a un modelo con mayor numero de parámetros como es Qwen2-VL por un 11.86 %. En lo que respecta a velocidad, la velocidad de LLava es 120.2 % mayor a Janus, y 214.05 % mayor a Qwen2-VL. Sin embargo, a pesar de ser el mas veloz de los modelos comparados, es el de menor precision, por que el modelo seleccionado es Janus-Pro-1.3B, el modelo mas preciso y que es 42.62 % mas rápido que Qwen2-VL.

6.1.2. Selección del prompt para MLLM

Para determinar la plantilla que recibe el MLLM, se evaluó la pregunta realiza al modelo, al ser un elemento fijo que puede presentar diferencias, evaluando algunas de las formas de

expresar la pregunta.

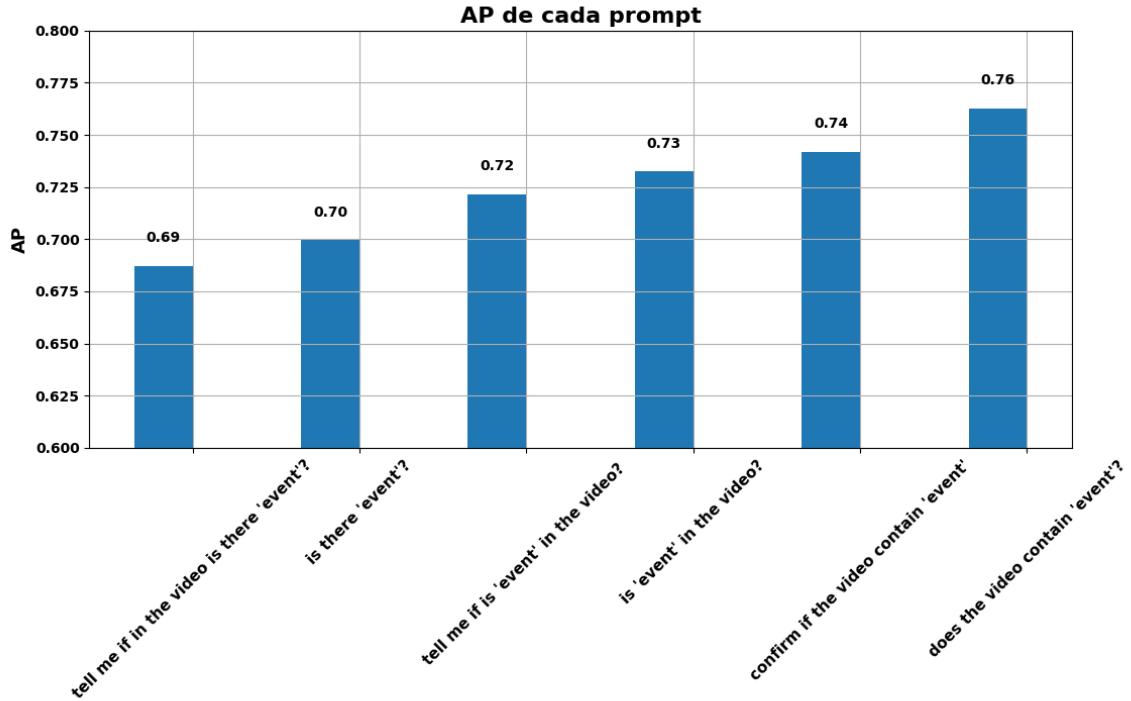


Figura 6.2: Average Precision de las diversas preguntas a 107 videos de la CHAD dataset en 6 eventos.

Se utilizaron los eventos: Andar en bicicleta, Pelear, Jugar, Correr, Estar acostado y Perseguir a alguien, al ser considerados los eventos de menor dificultad de entendimiento para el MLLM, y los eventos con el mayor numero de videos. De igual forma se recurrió al método de MLLM Solo 5.2 para evitar la interacción con los demás componentes de la arquitectura.

Analizando la diferencia de AP , entre el menor valor y el mayor existe una brecha de 0.07, siendo porcentualmente mayor 10.14% al de la pregunta de menor valor, por lo que la forma de la pregunta elaborada al MLLM impacta en los resultados considerablemente. La mejor pregunta entre las evaluadas es 'does the video contain {event}', siendo elegida para las siguientes pruebas.

6.1.3. Análisis de los diferentes prompts

Al contar con la parte fija de la plantilla, se procedió a probar los prompts elaborados por los algoritmos desarrollados, evaluando su impacto en el desempeño de la arquitectura. Para lo

cual se utilizaran todos los videos de la dataset CHAD con todos los eventos en los métodos que cuenten con información.

6.1.3.1. Prompt simple

En este experimento el algoritmo brinda un texto con la información de la cantidad de objetos detectados en las escenas, y su correspondiente clase, siendo un texto consiso y directo, los resultados se muestran en la figura 6.3.

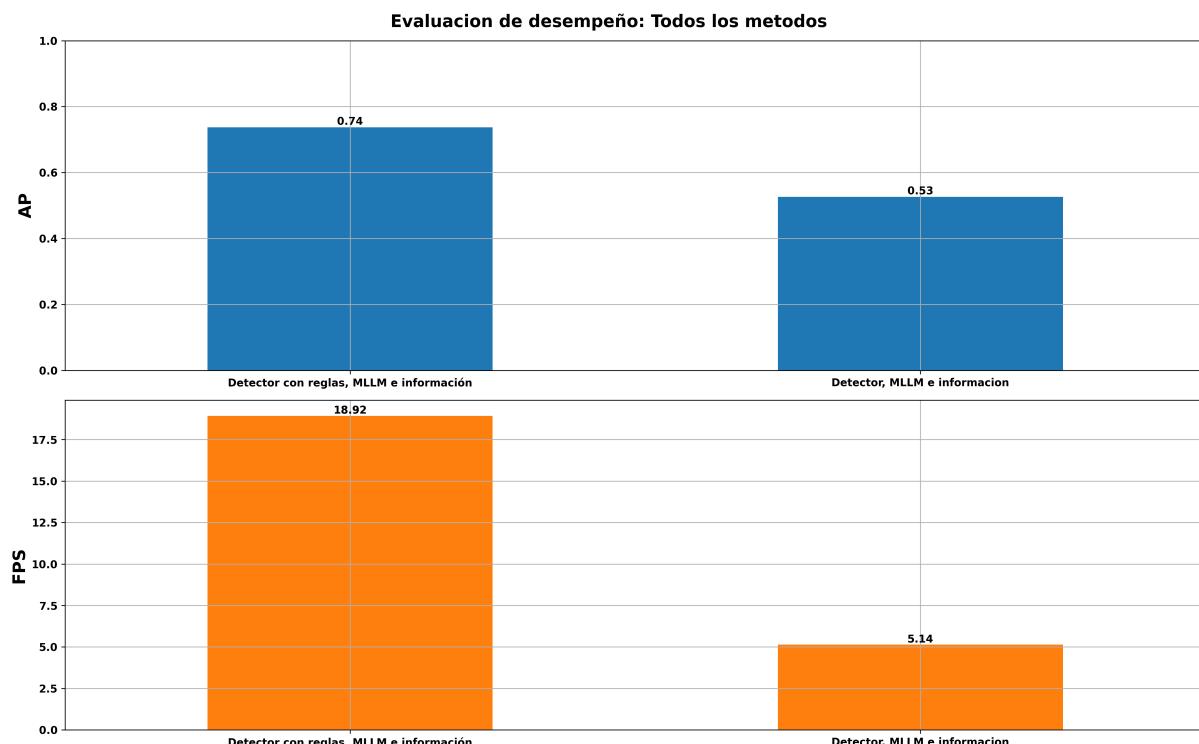


Figura 6.3: Average Precision y FPS de cada método con información de los objetos detectados.

6.1.3.2. Prompt con localización de los objetos

Para este experimento la información indicada en el prompt es la cantidad de objetos detectados, y su correspondiente clase, indicando su respectiva información entre los diversos cuadrantes que componen los fotogramas, teniendo un texto mas extenso y detallado cuyos resultados se aprecian en la siguiente figura 6.4.

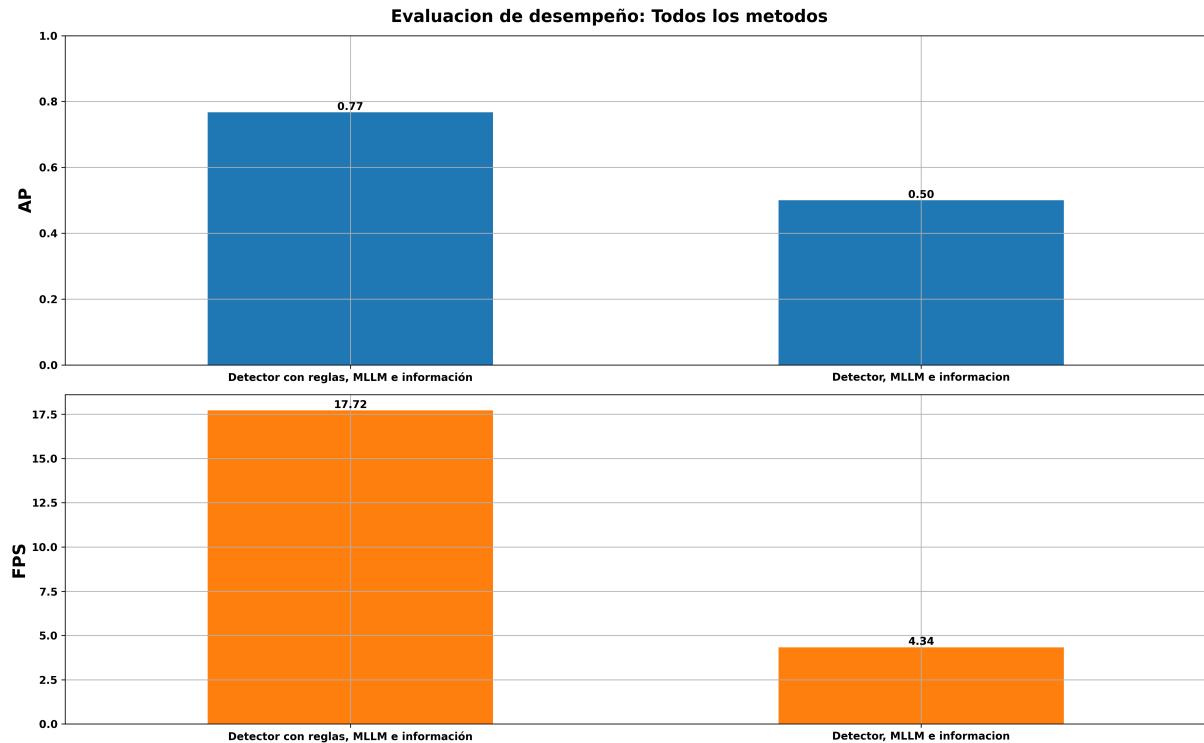


Figura 6.4: Average Precision y FPS de cada método con información de los objetos detectados y su ubicación en las escenas.

6.1.3.3. Comparativa

Contando con los resultados de ambos experimentos, estos son expuestos en la tabla 6.1 para proseguir con el análisis entre ambos.

Metodo	Prompt simple		Prompt con localizacion	
	AP	FPS	AP	FPS
Detector con reglas, MLLM e información	0.74	18.92	0.77	17.72
Detector, MLLM e informacion	0.53	5.14	0.5	4.34

Tabla 6.1: Comparativa entre prompts en cada metodo.

Como se puede observar, el impacto de la información extra incrementa 0.03 el AP del método Detector con reglas, MLLM e información 5.5 que integra todos los componentes de la arquitectura, mientras que el método Detector, MLLM e información 5.4 disminuye 0.03, mientras que ambos métodos presencian una caída en el numero de FPS. Aunque porcentualmente, el método Detector con reglas, MLLM e información presenta un incremento del 4.05 % en AP, es a costa de una disminución del 6.34 % en los FPS.

Resultado que la información extra de localización de los objetos permite al MLLM tener una mejor comprensión de las escenas cuando se le indica al modelo en que fotogramas realizar la evaluación por medio de un componente externo, como son las reglas. En el caso de la validación constante, esta información puede confundir al modelo y obtener una reducción de precisión. Ademas de ser cuestionable la disminución de FPS por un ligero incremento en el AP, siendo una consideración a tomar en cuenta.

6.1.4. Análisis de los videos utilizados durante el desarrollo

Continuando con la evaluación de la arquitectura, es que se procesaron todos los videos de la dataset CHAD por medio de todos los métodos establecidos con el prompt de información adicional en su respectivo evento, que incluye la localización, los resultados se presentan en la figura 6.5.

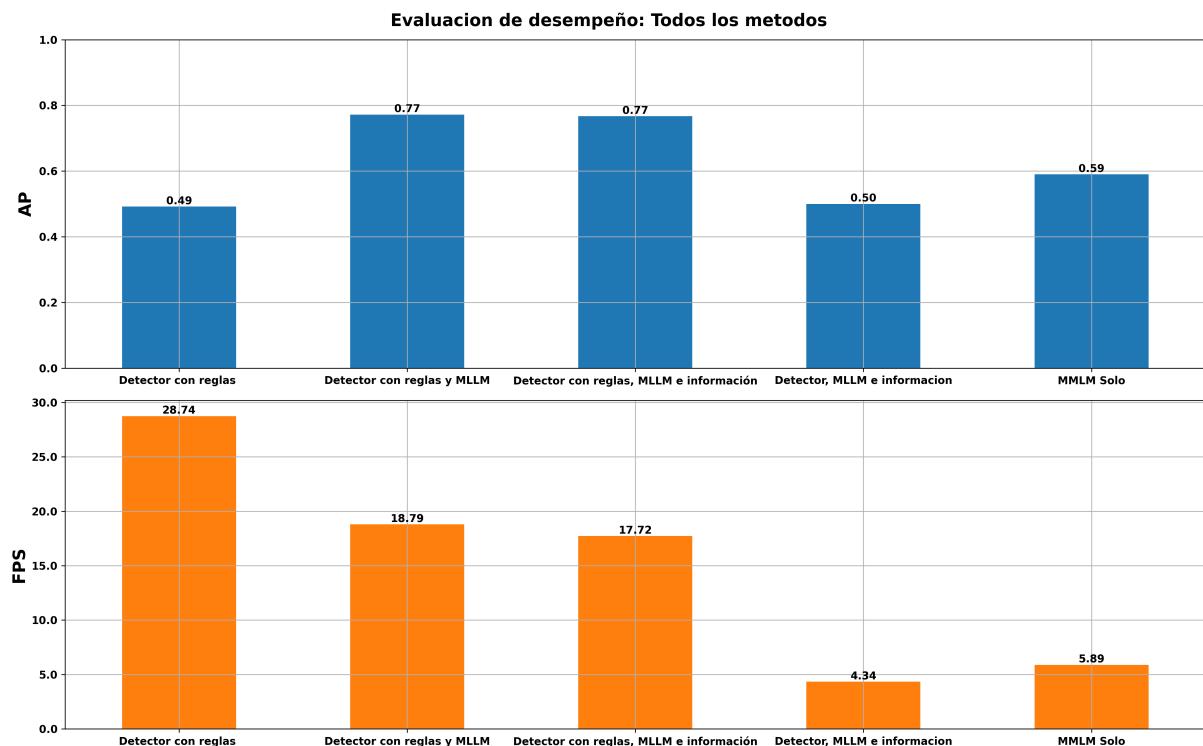


Figura 6.5: Average Precision y FPS de cada método con todos los videos de la CHAD dataset.

Observando una clara igualdad entre los 5.5 y el de la arquitectura completa en AP, aunque esta ultima presenta menor FPS que su contraparte, por la parte del método 5.3, este es capaz de ejecutarse a un valor próximo a los 30 FPS, y cuenta con un valor de AP menor pero cercano a los otros dos métodos. Entre los métodos 5.2 y 5.4 el MLLM presenta un superior AP con un mayor FPS.

6.1.5. Incorporación de los nuevos videos

El experimento anterior sera repetido con los videos de las datasets IITB CORRIDOR [Rodrigues et al., 2020], NWPU Campus [Cao et al., 2023] y Avenue dataset [Lu et al., 2013], procesados con la finalidad de evaluar el rendimiento de la arquitectura ante videos ajenos a aquellos probados durante el desarrollo, así como evaluar su capacidad de funcionar con vistas distintas a las únicas 4 que contiene la dataset CHAD.

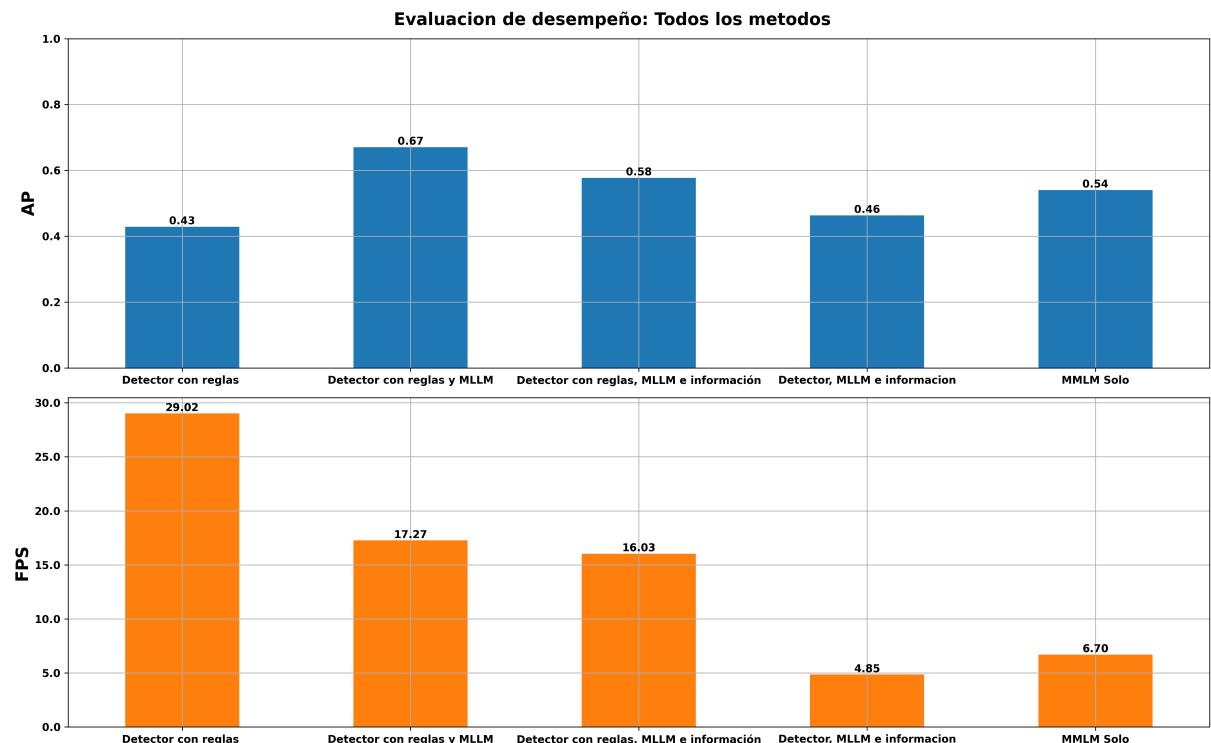


Figura 6.6: Average Precision y FPS de cada métodos con los videos de las demás datasets

Como se puede apreciar en la figura 6.6, comparando con 6.5 se aprecia como en todos los métodos se ve una disminución en el AP, sobre todos en aquellos que incluyen reglas. Es remarcable que debido a la distinta resolución de los videos, el método de Detector con reglas incrementa los FPS para estos videos.

Evaluando ambas datasets unidas, los resultados se muestran en la figura 6.7

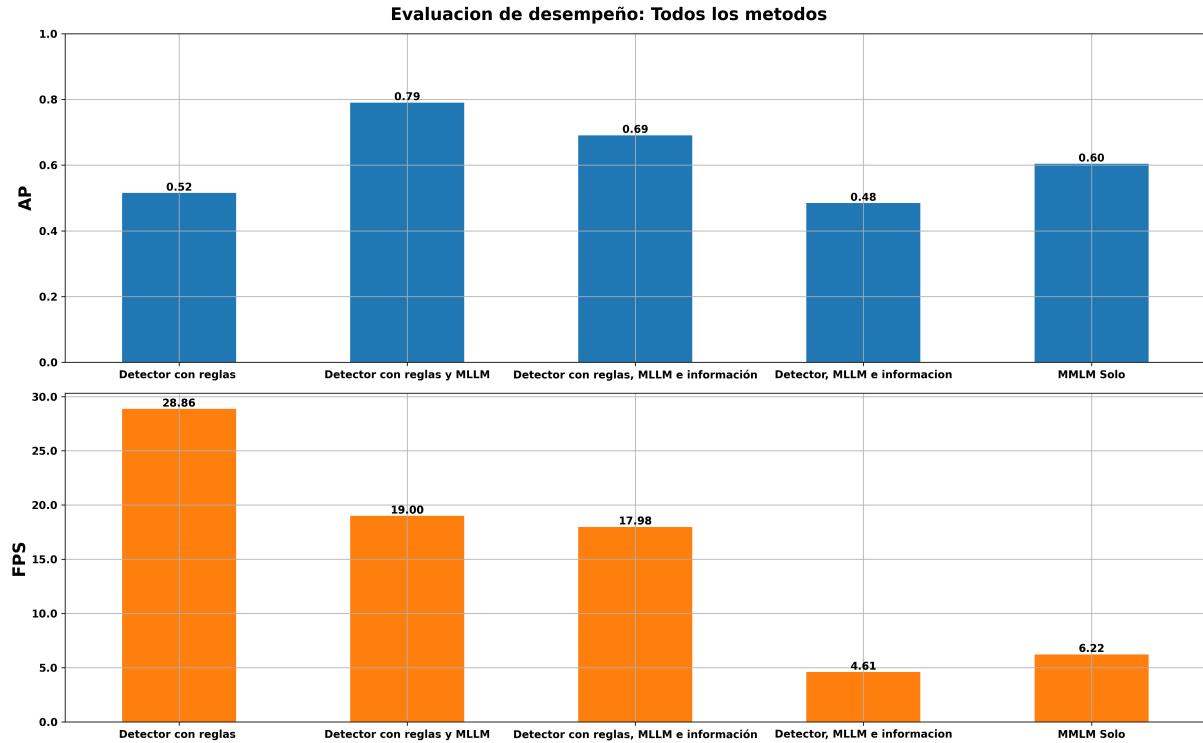


Figura 6.7: Average Precision y FPS de cada método con todos los videos

Contando con resultados semejantes a los de la dataset CHAD vista en la figura 6.5, sin embargo se presenta el cambio que el metodo con menor AP es el Detector, MLLM e informacion 5.4, siendo superado por el Detector con reglas 5.3, el cual es capaz de correr a 30 FPS, superado por el MLLM solo 5.2 en precisión. Por ultimo, la arquitectura completa presenta un valor de AP inferior pero cercano a su contraparte sin información, que a su vez es capaz de correr a mayor FPS, esto debido a su bajo rendimiento en los videos de las demás datasets, siendo el método 'Detector con reglas y MLLM' aquel con mayor valor de AP, con la mitad de FPS que el método 5.3.

6.2. CLIP en la arquitectura

Con la incorporación de a la arquitectura, y manejando los métodos observados en 4.2 se procede a observar los resultados, se omitira el primer metodo al no contar con componentes de la arquitectura, por lo que se mostraran únicamente los resultados de CLIP con reglas, y CLIP con reglas y MLLM.

Para comenzar, se procesaron los videos de la CHAD dataset, observando los resultados en la figura 6.8 y presentando una mejoría notoria al incluir la verificación del MLLM, sin

embargo, conlleva una gran reducción en los FPS.

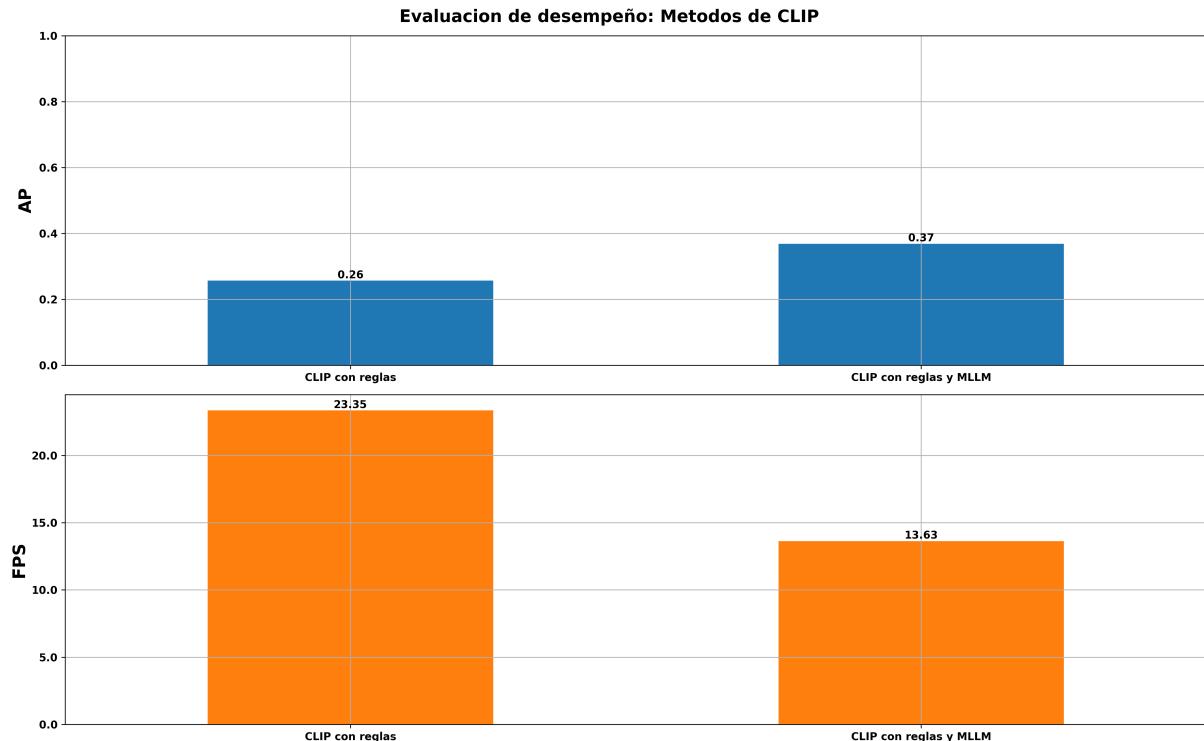


Figura 6.8: Resultados del uso de CLIP en los videos de CHAD Dataset.

El experimento se repitió para los videos de las demás datasets, apreciando los resultados en la figura 6.9 manteniendo resultado parecidos, presentando una disminución mínima de 0.01 de AP para el método sin MLLM y un incremento de 0.04 de AP cuando se utiliza en la validación. Así como un incremento en los FPS debido a la resolución de algunos videos.

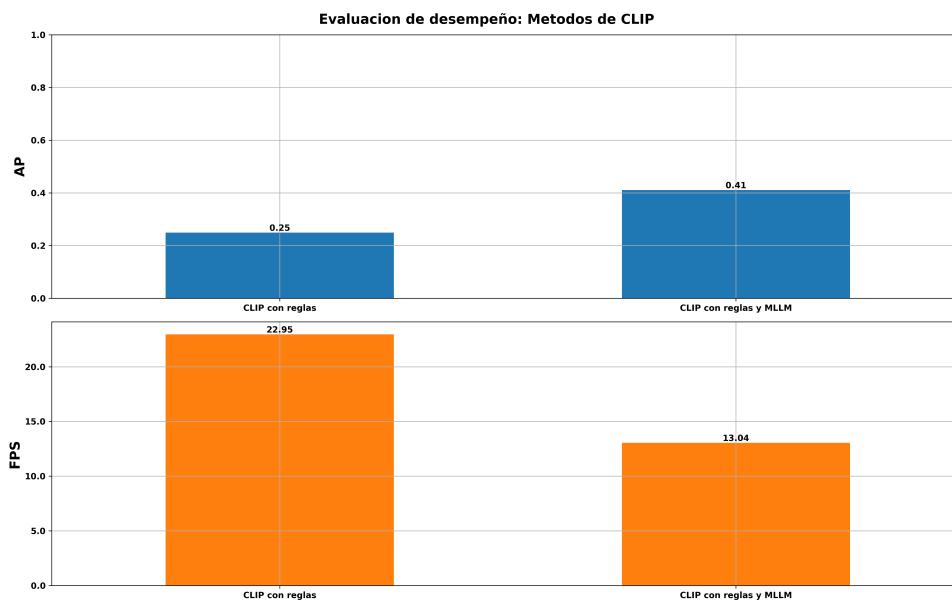


Figura 6.9: Resultados del uso de CLIP en los videos de las demás datasets.

Por ultimo, en el caso de todos los videos procesados de forma conjunta, los resultados presentan como el método con MLLM supera por 0.1 de AP, pero con una gran reducción de FPS

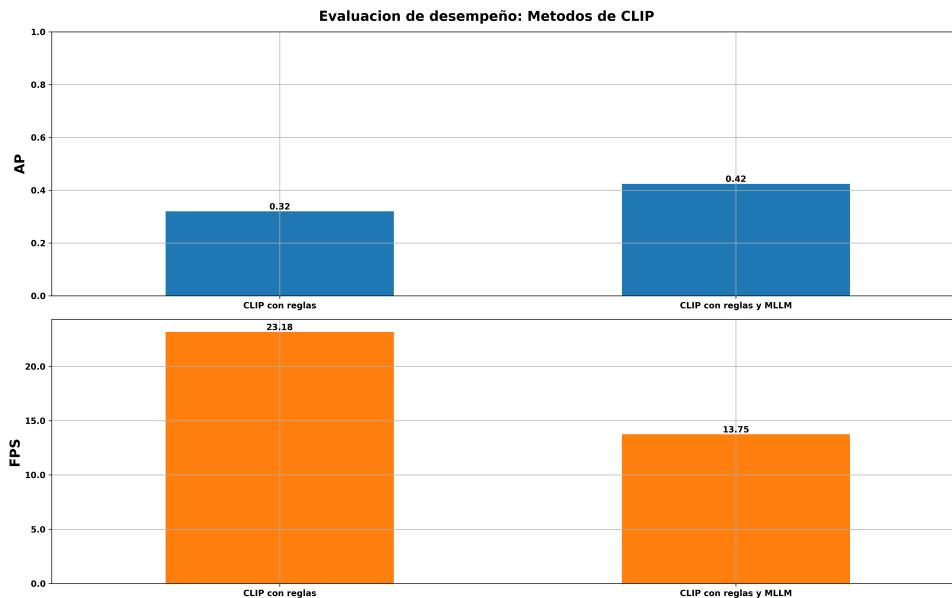


Figura 6.10: Resultados del uso de CLIP en todos los videos.

6.3. Análisis

Contando con los resultados de los experimentos, se procederá a evaluar cada uno de los métodos utilizados, comenzando con los de la arquitectura propuesta observados en la tabla 6.2.

- **Detector con reglas:** Como se esperaba es el método con el menor AP con 0.49 en CHAD, pero siendo el mas cercano a un valor de ejecución cercano a los 30 FPS, al incrementar los videos de la dataset su AP incremento en 6.12 %, mostrando que las reglas son capaces de funcionar en diversas vistas.
- **MLLM Solo:** Presenta un valor superior al anterior con 0.59 de AP, pero con la segunda menor velocidad de procesamiento entre todos los métodos, con 5.89 FPS, siendo no ideal para aplicaciones en tiempo real. Al incrementar los videos su precisión se mantuvo similar, al igual que su velocidad.
- **Detector, MLLM e información:** Presenta un AP similar al método de 'Detector con reglas' con 0.01 de diferencia, pero con la menor velocidad entre todos los métodos, esto al correr el detector multiclas, el MLLM e incluir información en el prompt, requiriendo mayor procesamiento. Al utilizar todos los videos, se ve como el AP disminuye, siendo superado por el método 'Detector con reglas' por 0.03, pasando a ser el método con menor AP.
- **Detector con reglas y MLLM:** Se observa su desempeño en la dataset CHAD, comparando el mayor AP con el propio método mas información en el prompt, mientras que con todos los videos lo deja atrás, incrementando 0.02 en su AP, y corriendo hasta 19 FPS, incrementando en un 205.47 % la cantidad del MLLM Solo, con un incremento de 31.67 % de AP.
- **Detector con reglas, MLLM e información:** Contando con un rendimiento parecido a su contraparte sin información, este método comparte el primer lugar de AP para la dataset CHAD, y con el segundo valor de FPS, pero al utilizar todos los videos, con las nuevas vistas, presenta una disminución de 0.08 de AP, siendo superada por su versión sin información, para tener el segundo mayor AP y FPS.

Con los componentes principales de la arquitectura demuestran claramente sus fortalezas y debilidades opuestas al utilizarse de forma aislada, ya que el detector con reglas presenta poca precisión con alta velocidad de procesamiento, para el MLLM Solo, se observa un incremento de la precisión a costa de un considerable mayor costo de procesamiento. Sirviendo de punto de referencia para observar como los métodos que combinan componentes influyen en el rendimiento de la arquitectura.

Es así como el metodo Detector, MLLM e información presenta como el uso de información extra en el prompt no es aprovechado por el MLLM para devolver una respuesta, sino que incluso puede reducirla al ingresar un mayor texto al modelo. Se especula que esta confusión que causa la información para el MLLM puede deberse a su tamaño, ya que como se aprecio en la figura 6.1, el precision puede incrementar con el numero de parámetros, así como con el tipo de arquitectura del modelo, por lo que piensa que un modelo mayor es capaz de aprovechar esta información extra, aunque eso conllevaría un mayor tiempo de procesamiento perjudicial para correr en aplicaciones en tiempo real.

Dejando así las variantes que combinan los dos componentes principales, Detector con reglas y MLLM y Detector con reglas, MLLM e información, mientras que el primero presenta el mejor rendimiento y precision en la dataset CHAD y con todos los videos, su contraparte con información en el prompt mantiene la misma precision con menor FPS para la dataset CHAD, mientras que para las nuevas vistas de los demás videos, a pesar que los demás métodos con detector y reglas mejoran su AP, este método disminuye por impacto que tiene la información en el.

Por lo que para modelos de poco tamaño la informacion proveniente de los detectores multiclase se aprovechan de mejor manera para indicar la posibilidad de un evento a un MLLM, como en el uso de reglas, mientras que la informacion de las detecciones puede disminuir el desempeño del MLLM al validar un evento.

Metodo	CHAD		Todos	
	AP	FPS	AP	FPS
Detector con reglas	0.49	28.74	0.52	28.86
Detector con reglas y MLLM	0.77	18.79	0.79	19.0
Detector con reglas, MLLM e información	0.77	17.72	0.69	17.98
Detector, MLLM e información	0.50	4.34	0.48	4.61
MMLM Solo	0.59	5.89	0.60	6.22

Tabla 6.2: Comparación de métodos en los videos utilizados en el desarrollo con todos completos.

Para los metodos de CLIP se puede apreciar como los FPS se mantienen en valores cercanos, aun con todos los videos de la dataset, asi como se puede apreciar como esta dataset aumentada incrementa el AP, mejorando el del metodo CLIP con reglas en un 42 % y CLIP con reglas y MLLM en un 13.51 %, disminuyendo la diferencia de ambos metodos de 0.09 de AP a 0.05.

Metodo	CHAD		Todos	
	AP	FPS	AP	FPS
CLIP con reglas	0.26	23.35	0.37	23.18
CLIP con reglas y MLLM	0.37	13.63	0.42	13.75

Tabla 6.3: Comparación de métodos de CLIP en los videos utilizados en el desarrollo con todos completos.

El leve bajo rendimiento de los métodos con CLIP en la dataset CHAD observado en la figura 6.8, con respecto a la de los demás videos de la figura 6.9 y con todos conjuntos 6.10 se puede deber al uso de videos pertenecientes a diversos eventos, ya que aunque se evalúen en diferentes eventos, las respuestas de CLIP se mantendrán igual, llevando a peores resultados, caso contrario de los videos de las demás datasets, que se decidió no repetir videos en diversos eventos, lo que permite mejores resultados.

6.4. Comparativa

Capítulo 7

Conclusiones

Apéndices

Apéndice A

Algoritmos y códigos

Bibliografía

- [Caffagni et al., 2024] Caffagni, D., Cocchi, F., Barsellotti, L., Moratelli, N., Sarto, S., Baraldi, L., Baraldi, L., Cornia, M., and Cucchiara, R. (2024). The revolution of multimodal large language models: A survey.
- [Cao et al., 2023] Cao, C., Lu, Y., Wang, P., and Zhang, Y. (2023). A new comprehensive benchmark for semi-supervised video anomaly detection and anticipation.
- [Chen et al., 2025] Chen, X., Wu, Z., Liu, X., Pan, Z., Liu, W., Xie, Z., Yu, X., and Ruan, C. (2025). Janus-pro: Unified multimodal understanding and generation with data and model scaling.
- [Danesh Pazho et al., 2023] Danesh Pazho, A., Alinezhad Noghre, G., Rahimi Ardabili, B., Neff, C., and Tabkhi, H. (2023). *CHAD: Charlotte Anomaly Dataset*, page 50–66. Springer Nature Switzerland.
- [Dettmers et al., 2022] Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. (2022). Llm.int8(): 8-bit matrix multiplication for transformers at scale.
- [Ding and Wang, 2024] Ding, X. and Wang, L. (2024). Quo vadis, anomaly detection? llms and vlms in the spotlight.
- [Dosovitskiy et al., 2021] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale.
- [LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.
- [Lee et al., 2024] Lee, B.-K., Park, B., Kim, C. W., and Ro, Y. M. (2024). Moai: Mixture of all intelligence for large language and vision models.
- [Li et al., 2024] Li, B., Zhang, Y., Guo, D., Zhang, R., Li, F., Zhang, H., Zhang, K., Zhang, P., Li, Y., Liu, Z., and Li, C. (2024). Llava-onevision: Easy visual task transfer.
- [Lu et al., 2013] Lu, C., Shi, J., and Jia, J. (2013). Abnormal event detection at 150 fps in matlab. In *2013 IEEE International Conference on Computer Vision*, pages 2720–2727.

- [Lv and Sun, 2024] Lv, H. and Sun, Q. (2024). Video anomaly detection and explanation via large language models.
- [Marafioti et al., 2025] Marafioti, A., Zohar, O., Farré, M., Noyan, M., Bakouch, E., Cuenca, P., Zakka, C., Allal, L. B., Lozhkov, A., Tazi, N., Srivastav, V., Lochner, J., Larcher, H., Morlon, M., Tunstall, L., von Werra, L., and Wolf, T. (2025). Smolvlm: Redefining small and efficient multimodal models.
- [Ntelopoulos and Nasrollahi, 2024] Ntelopoulos, A. and Nasrollahi, K. (2024). Callm: Cascading autoencoder and large language model for video anomaly detection. In *2024 IEEE Thirteenth International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6.
- [Radford et al., 2021] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021). Learning transferable visual models from natural language supervision.
- [Rodrigues et al., 2020] Rodrigues, R., Bhargava, N., Velmurugan, R., and Chaudhuri, S. (2020). Multi-timescale trajectory prediction for abnormal human activity detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.
- [Sultani et al., 2019] Sultani, W., Chen, C., and Shah, M. (2019). Real-world anomaly detection in surveillance videos.
- [Tang et al., 2024] Tang, J., Lu, H., Wu, R., Xu, X., Ma, K., Fang, C., Guo, B., Lu, J., Chen, Q., and Chen, Y.-C. (2024). Hawk: Learning to understand open-world video anomalies.
- [Vaswani et al., 2023] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention is all you need.
- [Wang et al., 2024a] Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., and Ding, G. (2024a). Yolov10: Real-time end-to-end object detection.
- [Wang et al., 2024b] Wang, P., Bai, S., Tan, S., Wang, S., Fan, Z., Bai, J., Chen, K., Liu, X., Wang, J., Ge, W., Fan, Y., Dang, K., Du, M., Ren, X., Men, R., Liu, D., Zhou, C., Zhou, J., and Lin, J. (2024b). Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution.
- [Wolf et al., 2020] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. (2020). Transformers: State-of-the-art natural language processing. In Liu, Q. and Schlangen, D., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- [Wu et al., 2020] Wu, P., Liu, J., Shi, Y., Sun, Y., Shao, F., Wu, Z., and Yang, Z. (2020). Not only look, but also listen: Learning multimodal violence detection under weak supervision.

- [Wu et al., 2023] Wu, P., Zhou, X., Pang, G., Zhou, L., Yan, Q., Wang, P., and Zhang, Y. (2023). Vadclip: Adapting vision-language models for weakly supervised video anomaly detection.
- [Xiao et al., 2023] Xiao, B., Wu, H., Xu, W., Dai, X., Hu, H., Lu, Y., Zeng, M., Liu, C., and Yuan, L. (2023). Florence-2: Advancing a unified representation for a variety of vision tasks.
- [Yang et al., 2024] Yang, Y., Lee, K., Dariush, B., Cao, Y., and Lo, S.-Y. (2024). Follow the rules: Reasoning for video anomaly detection with large language models.
- [Ye et al., 2025] Ye, M., Liu, W., and He, P. (2025). Vera: Explainable video anomaly detection via verbalized learning of vision-language models.
- [Zanella et al., 2024] Zanella, L., Menapace, W., Mancini, M., Wang, Y., and Ricci, E. (2024). Harnessing large language models for training-free video anomaly detection.
- [Zhang et al., 2024] Zhang, H., Xu, X., Wang, X., Zuo, J., Han, C., Huang, X., Gao, C., Wang, Y., and Sang, N. (2024). Holmes-vad: Towards unbiased and explainable video anomaly detection via multi-modal llm.
- [Zhang et al., 2025] Zhang, H., Xu, X., Wang, X., Zuo, J., Huang, X., Gao, C., Zhang, S., Yu, L., and Sang, N. (2025). Holmes-vau: Towards long-term video anomaly understanding at any granularity.
- [Zhao et al., 2024] Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., and Chen, J. (2024). Detrs beat yolos on real-time object detection.