

# Sistema de Asistencia y Participación con Visión Computacional

Ana Cardenas A01284042, Diego Rodríguez A00829925

Elias Garza A01284041, José Romo A01197772

28 de noviembre de 2023

## Resumen

Este proyecto introduce un sistema automatizado de visión computacional que utiliza reconocimiento facial y de pose para monitorizar la asistencia y participación estudiantil en tiempo real. Desarrollado con dos aplicaciones complementarias, una webapp para acceso general y un ejecutable para uso exclusivo en el aula, el sistema garantiza la precisión y eficiencia, abordando la necesidad de métodos de gestión más objetivos y menos laboriosos. A través de pruebas rigurosas, se ha optimizado el modelo para funcionar bajo diversas condiciones ambientales y técnicas. El reconocimiento facial es efectivo hasta 4 metros con una confianza mínima de 85 %, mientras que la detección de poses alcanza hasta 8 metros con una confianza mínima de 65 %, ambos sujetos a las capacidades del hardware utilizado. Este avance representa un paso significativo hacia la integración de la inteligencia artificial en la educación, promoviendo entornos de aprendizaje más dinámicos y comprometidos.

## 1. Introducción

La visión computacional, una subdisciplina de la inteligencia artificial, se dedica al desarrollo de algoritmos capaces de interpretar y comprender el contenido visual del mundo humano. A través de la imitación del complejo procesamiento visual humano, esta tecnología permite que las máquinas detecten, sigan y clasifiquen objetos, y reaccionen a lo que "ven". Con aplicaciones que se extienden por un amplio espectro, desde la seguridad y vigilancia hasta la medicina y la navegación autónoma, la visión computacional está transformando innumerables industrias y prácticas sociales.

No obstante, mientras la visión computacional avanza, también surgen desafíos significativos. Las preocupaciones por la privacidad y el uso ético, especialmente en el reconocimiento facial, resaltan la necesidad de un enfoque equilibrado que valore tanto la seguridad como las libertades individuales. La precisión de los sistemas puede verse comprometida por variaciones ambientales, y la dependencia de la automatización plantea preguntas sobre el impacto en las habilidades humanas. A pesar de estos desafíos, las ventajas de la automatización y el incremento en la eficiencia prometen mejorar la capacidad de las máquinas para asistir en tareas complejas y reducir la carga de actividades repetitivas.

En el ámbito educativo, la visión computacional se presenta como una herramienta revolucionaria para la administración de asistencia y la participación de estudiantes en el aula. El uso de técnicas avanzadas de reconocimiento facial y estimación de pose facilita la creación de sistemas capaces de registrar automáticamente la presencia y contribuciones de los estudiantes, ofreciendo un panorama detallado y objetivo de la dinámica de la clase. Este enfoque libera a los docentes de tareas

administrativas y enriquece el ambiente educativo al proporcionar datos valiosos para la participación estudiantil. La implementación de estas tecnologías no solo tiene el potencial de mejorar la calidad de la educación y la eficiencia del tiempo en clase, sino que también permite a los educadores ajustar su enseñanza a las necesidades individuales y fortalecer la experiencia de aprendizaje, garantizando un seguimiento integral de la implicación del estudiante en el proceso educativo.

En este reporte, exploraremos cómo se diseñó e implementó un sistema de visión computacional en un entorno de aula para llevar a cabo estas tareas, analizando su eficacia, las métricas de rendimiento relevantes y las lecciones aprendidas durante su desarrollo y despliegue.

## **2. Organización del Proyecto**

### **2.1. Memorando de Entendimiento**

Este proyecto se ha desarrollado en colaboración con NDS Cognitive Labs, una empresa líder en la vanguardia de soluciones de inteligencia artificial. El socio formador se ha comprometido con la innovación tecnológica y ha apoyado a nuestro equipo con su experiencia y orientación a lo largo de la realización de este proyecto. El trabajo conjunto se ha regido por un Memorando de Entendimiento (MoU), que establece un acuerdo de colaboración para la creación de un sistema de pase de lista y detección de participación utilizando técnicas avanzadas de machine learning y visión computacional.

El alcance de nuestra colaboración ha sido bien definido, con responsabilidades y obligaciones claras para ambas partes. Nuestro equipo, referido como Equipo 4 en el MoU, ha sido responsable de la creación y desarrollo del sistema completo, incluyendo un dashboard para visualizar la participación de los alumnos y una aplicación o página web para la gestión de las clases y la asistencia en tiempo real. Por su parte, NDS Cognitive Labs se ha comprometido a proporcionar retroalimentación semanal sobre nuestros avances y a resolver cualquier duda que surgiera durante el proceso de desarrollo. Este acuerdo ha tenido una duración de 11 semanas, estableciendo un marco temporal claro para la entrega y evaluación de los avances del proyecto. El Memorando de Entendimiento completo se puede encontrar en la carpeta **Organizational Documents** del repositorio del proyecto cuya liga se encuentra en los anexos.

La sinergia entre Equipo 4 y NDS Cognitive Labs ha sido un componente crucial para el éxito del proyecto, demostrando cómo la colaboración entre la academia y la industria puede conducir a innovaciones significativas en el campo de la educación y la tecnología.

### **2.2. Requerimientos del Proyecto**

Para cumplir con los objetivos del proyecto, se han establecido requerimientos funcionales y no funcionales claros. Funcionalmente, el sistema está diseñado para detectar e identificar a los alumnos por su rostro (RF1), captar el movimiento de los alumnos (RF2), procesar y almacenar la asistencia y participación de cada alumno (RF3), y convertir el audio de la clase a texto (RF7). Además, se ha desarrollado un Dashboard en tiempo real que refleja la participación por alumno (RF4 y RF6), y un portal para la administración efectiva de las clases (RF5). En cuanto a los requerimientos no funcionales, se ha priorizado la accesibilidad, asegurando que tanto el portal como el Dashboard sean accesibles desde cualquier navegador web estándar (RNF1). Esta estructura de requerimientos garantiza que el sistema sea integral, interactivo y fácil de utilizar para los administradores y educadores, facilitando una transición suave hacia la integración tecnológica en el entorno educativo. El documento completo de los requerimientos se puede encontrar en la carpeta **Organizational Documents** del repositorio.

### **2.3. Privacidad y Seguridad de los Datos**

La integridad y confidencialidad de los datos son piedras angulares en nuestro proyecto. Se han implementado protocolos de anonimización de datos rigurosos para salvaguardar la información personal de los estudiantes. Esto incluye el enmascaramiento de nombres, edades, matrículas y la alteración de fotografías para prevenir la identificación directa. Además, la infraestructura de almacenamiento de datos en Deta Space se ha seleccionado por su capacidad para implementar estas medidas de privacidad antes de cualquier procesamiento o almacenamiento, y se ha reforzado con cifrado y otras técnicas de seguridad de datos.

En concordancia con la normativa de privacidad de datos, como la Ley de Protección de Datos en México, nuestro proyecto requiere el consentimiento explícito de los usuarios para el manejo de sus datos personales. Esto se complementa con derechos detallados para los titulares de los datos, medidas de seguridad estrictas, y procedimientos claros en caso de violaciones de datos. La ley enfatiza la necesidad de contar con un responsable de datos dentro de la organización y establece sanciones por el incumplimiento de sus estipulaciones. El tratamiento de los datos se realiza con cautela, almacenándose en servidores seguros o utilizando servicios en la nube de alta reputación, como Google Cloud Storage. Se ha dado prioridad a la red privada y al uso de VPN para el acceso remoto a los datos, con autenticación requerida en cada acceso. Además, el personal autorizado como profesores y administrativos deben firmar acuerdos de confidencialidad antes de acceder a los datos.

Finalmente, para mantener un registro transparente del acceso a los datos, se utilizan herramientas de monitoreo y auditoría. Estas herramientas no solo registran quién accedió a la información y cuándo, sino también supervisan las actividades dentro del sistema para alertar sobre cualquier actividad sospechosa o no autorizada, asegurando así una supervisión continua y una respuesta rápida a cualquier anomalía. El documento completo de la privacidad y seguridad de los datos se puede encontrar en la carpeta **Organizational Documents** del repositorio.

## **3. Antecedentes**

### **3.1. Face Recognition**

La evaluación del desempeño de sistemas de asistencia en aulas mediante tecnologías de reconocimiento facial se ha convertido en un área de interés creciente, evidenciada por la diversidad de métricas y metodologías empleadas en investigaciones recientes. Un estudio notable demostró que las Redes Neuronales Convolucionales (CNN) alcanzan una precisión del 95 %, sobresaliendo frente a técnicas más tradicionales como Local Binary Patterns Histograms (LBPH), Eigenfaces, Análisis de Componentes Principales (PCA) y Haar Cascade. La superioridad de las CNN se debe a su profunda estructura de aprendizaje, que les permite capturar y procesar características faciales complejas y distintivas, lo que es esencial para identificar con precisión a los individuos en diversas condiciones de iluminación y postura facial.

No obstante, la implementación efectiva de estas tecnologías en entornos educativos no está exenta de desafíos. Las variables ambientales, como la iluminación fluctuante y la presencia de objetos que pueden ocluir parcialmente el rostro, presentan obstáculos significativos que pueden afectar la precisión del sistema. Para abordar estos problemas, se han desarrollado y aplicado técnicas avanzadas de preprocessamiento de imágenes y estrategias de aumento de datos. Estas técnicas tienen como objetivo mejorar la robustez y la capacidad del modelo para generalizar a partir de los datos disponibles. La validación de estos enfoques se realiza a menudo mediante el uso de bases de datos

estandarizadas como Labeled Faces in the Wild (LFW), así como pruebas en entornos de aula reales, donde se han logrado tasas de precisión que alcanzan el 99.5 % y 97.83 %, respectivamente, lo que demuestra la viabilidad de la tecnología en aplicaciones del mundo real [1] [2].

Además, la integración de redes neuronales en los sistemas de video vigilancia ha abierto el camino hacia sistemas más sofisticados y autónomos para la gestión de la asistencia en el aula. Un enfoque notable dentro de este ámbito es el uso de sistemas embebidos, que combinan hardware y software para ofrecer una solución integrada que puede operar de manera independiente y en tiempo real. Estos sistemas no solo mejoran la eficiencia en la identificación de estudiantes, sino que también se adaptan de manera más fluida a las necesidades del ambiente educativo, lo que se refleja en su capacidad para procesar flujos de video en vivo y proporcionar retroalimentación inmediata, un aspecto esencial para la toma de asistencia automatizada [3].

Por tanto, a través de la consolidación de tecnologías avanzadas de reconocimiento facial y la aplicación de protocolos de evaluación rigurosos, el campo está avanzando hacia sistemas de asistencia automatizados que no solo mejoran la precisión y la eficiencia, sino que también promueven un entorno de aprendizaje más seguro y gestionado eficientemente. Estos avances, alineados con el desarrollo continuo de algoritmos y hardware especializado, sugieren un futuro prometedor para la integración de la visión computacional en el ámbito educativo.

### **3.2. Pose Detection**

Por otro lado, en el área de detección de poses se ha experimentado un progreso significativo, impulsado por la creciente precisión y eficiencia de los modelos de aprendizaje automático. En el contexto educativo, la detección de la participación de los alumnos mediante la estimación de poses se ha convertido en un área de interés particular, con estudios recientes alcanzando una precisión de hasta el 90.2 % en ambientes controlados. Este nivel de precisión se ha logrado mediante la optimización de las condiciones de captura de vídeo, como la configuración de la cámara y la iluminación, lo que resalta la importancia de un entorno de grabación bien planificado.

La posición de la cámara y su ángulo de visión desempeñan roles fundamentales en la capacidad del sistema para identificar correctamente las posturas de los estudiantes. Una colocación estratégica de la cámara puede mitigar las limitaciones impuestas por las variaciones en la postura y la proximidad de los estudiantes al lente, mejorando así la precisión global del sistema. Además, la implementación de modelos de alta resolución para la estimación de poses individuales ha mostrado una mejora notable en la velocidad de procesamiento, un factor crítico para la aplicación en tiempo real. La incorporación de algoritmos de seguimiento ha contribuido a una reducción en los tiempos de procesamiento del 30.8 %, permitiendo que el sistema opere de manera más fluida y dinámica durante las sesiones de clase activas.

Para evaluar de manera exhaustiva el rendimiento de estos sistemas, se han utilizado diversas métricas. La precisión en la estimación de poses es, por supuesto, un parámetro clave, pero otros factores, como la capacidad de detección de objetos y el rendimiento computacional, también son críticos. Las métricas de rendimiento, como el uso de la GPU y la CPU, el consumo de memoria, las operaciones de acumulación de multiplicaciones (MACs) y los cuadros por segundo (FPS), proporcionan una visión holística de la viabilidad operativa del sistema. Estas métricas no solo reflejan la capacidad del sistema para realizar estimaciones de pose en tiempo real, sino que también ofrecen una perspectiva sobre la escalabilidad y la eficiencia energética del modelo implementado.

Este enfoque multifacético para la evaluación del rendimiento garantiza que la solución seleccionada para la detección de poses en el aula no solo sea precisa, sino también sostenible y práctica para

su uso prolongado en entornos educativos. Los avances en este campo prometen mejorar la forma en que se monitorean y analizan la participación y el compromiso de los estudiantes, proporcionando a los educadores herramientas valiosas para evaluar y mejorar la dinámica de aprendizaje en el aula [4] [5] [6].

### 3.3. Rendimiento y evaluación

La literatura revisada subraya la importancia de seleccionar métricas de rendimiento apropiadas para los desafíos específicos de monitorizar la asistencia y participación de los alumnos en un entorno de aula. La precisión, la velocidad de procesamiento y la eficiencia en el uso de recursos surgen como indicadores clave para evaluar y comparar la efectividad de los modelos de visión por computadora implementados en este proyecto. La comparación de estos modelos muestra que mientras que las CNN ofrecen alta precisión en la identificación de rostros, las técnicas de estimación de pose requieren consideraciones adicionales como la configuración de la cámara para alcanzar niveles similares de precisión. Además, la eficiencia del procesamiento es crucial, ya que impacta directamente en la capacidad del sistema para operar en tiempo real, un requisito indispensable en el contexto de la asistencia y participación en aulas.

Interpretando de los resultados de la investigación nos lleva a concluir que una combinación de CNN para reconocimiento facial con una configuración optimizada de estimación de pose puede ofrecer un sistema robusto para el seguimiento de asistencia y participación. Sin embargo, la adopción de tal sistema debe balancear la precisión y la eficiencia de procesamiento con consideraciones prácticas como la facilidad de implementación y la integración con la infraestructura existente en el aula.

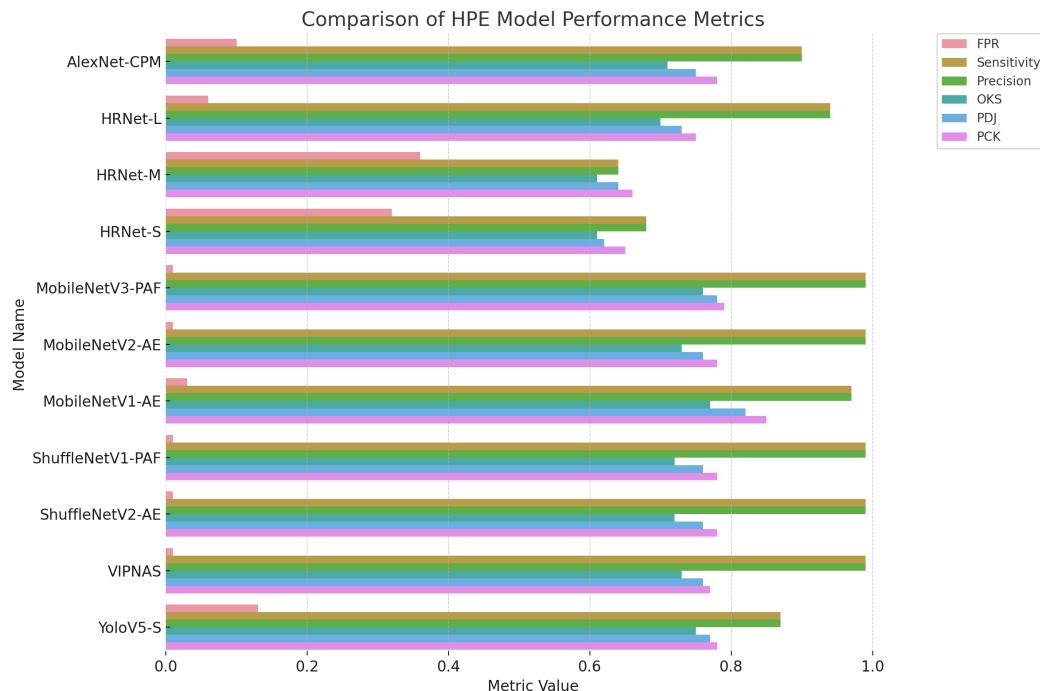


Figura 1: Pose Detection Metrics Comparison

## 4. Método Propuesto

### 4.1. Face Recognition

El mercado actual ofrece una gama de soluciones tecnológicas para el reconocimiento facial, cada una con sus propias fortalezas y enfoques especializados. OpenCV es una de las bibliotecas más utilizadas, conocida por su amplio conjunto de herramientas de procesamiento de imágenes y visión por computadora, aunque su implementación de reconocimiento facial puede requerir un mayor desarrollo a medida. Microsoft Azure Face API y Amazon Rekognition ofrecen potentes servicios basados en la nube con modelos de aprendizaje profundo altamente afinados, pero vienen con dependencias de infraestructura de nube y posibles consideraciones de costos. Google Cloud Vision API proporciona una solución robusta y de fácil acceso que incluye capacidades de aprendizaje automático avanzadas, pero al igual que los servicios de Azure y Amazon, la privacidad de los datos puede ser una preocupación debido al procesamiento de datos en la nube. Por último, la biblioteca **face-recognition** se basa en dlib y ofrece una solución enfocada en el reconocimiento facial con una gran comunidad y actualizaciones constantes.

En este contexto, la elección de la biblioteca **face-recognition** para nuestro sistema se basó en una evaluación cuidadosa de las opciones disponibles. La biblioteca **face-recognition** sobresale por su simplicidad y facilidad de uso, que permite una integración rápida y un desarrollo eficiente. Se basa en dlib, que incorpora un modelo de reconocimiento facial de vanguardia, ofreciendo una alta precisión en la identificación facial incluso bajo variaciones significativas de iluminación y orientación del rostro. Su naturaleza de código abierto y la comunidad activa proporcionan un soporte constante y aseguran que el sistema se mantenga actualizado con los avances tecnológicos. Estas ventajas, combinadas con su desempeño probado en términos de velocidad y eficiencia operativa, posicionan a **face-recognition** como la opción más adecuada para nuestro proyecto, garantizando así la consecución de nuestros objetivos y la entrega de una experiencia de usuario óptima.

El algoritmo presentado en la sección de anexos (7.1) describe el proceso de reconocimiento facial en tiempo real utilizando una cámara y la biblioteca de visión por computadora cv2 junto con la biblioteca **face\_recognition**. Inicialmente, se carga un conjunto de rostros conocidos desde una carpeta, procesando cada imagen para obtener las codificaciones faciales y asociarlas con los nombres de los individuos correspondientes. Estas codificaciones y nombres se almacenan para su uso posterior en la comparación con los rostros detectados en el video en vivo. Durante la ejecución del algoritmo, cada cuadro capturado por la cámara se procesa alternadamente para optimizar la velocidad y los recursos. El cuadro se reduce en tamaño para mejorar la velocidad de procesamiento y se convierte al espacio de color RGB, que es el formato requerido por la biblioteca **face\_recognition**. Luego, se detectan las ubicaciones de los rostros y se extraen sus codificaciones. Para cada rostro detectado, se compara su codificación con las codificaciones de los rostros conocidos para encontrar coincidencias. Si se encuentra una coincidencia, se asocia el nombre correspondiente al rostro; de lo contrario, se etiqueta como "Desconocido". El algoritmo también dibuja rectángulos alrededor de los rostros detectados y coloca el nombre identificado debajo de cada rostro. Si un nombre reconocido corresponde a un alumno conocido y presente en la sesión de Streamlit, se actualiza su estado para reflejar que ha sido visto por la cámara. Finalmente, el cuadro procesado se convierte de nuevo a BGR para su visualización y se muestra en la interfaz de usuario de Streamlit. Este proceso se repite para cada cuadro mientras la cámara esté activa, permitiendo un seguimiento continuo y en tiempo real de los rostros reconocidos durante la sesión.



Figura 2: Demostración del Algoritmo de Face Recognition

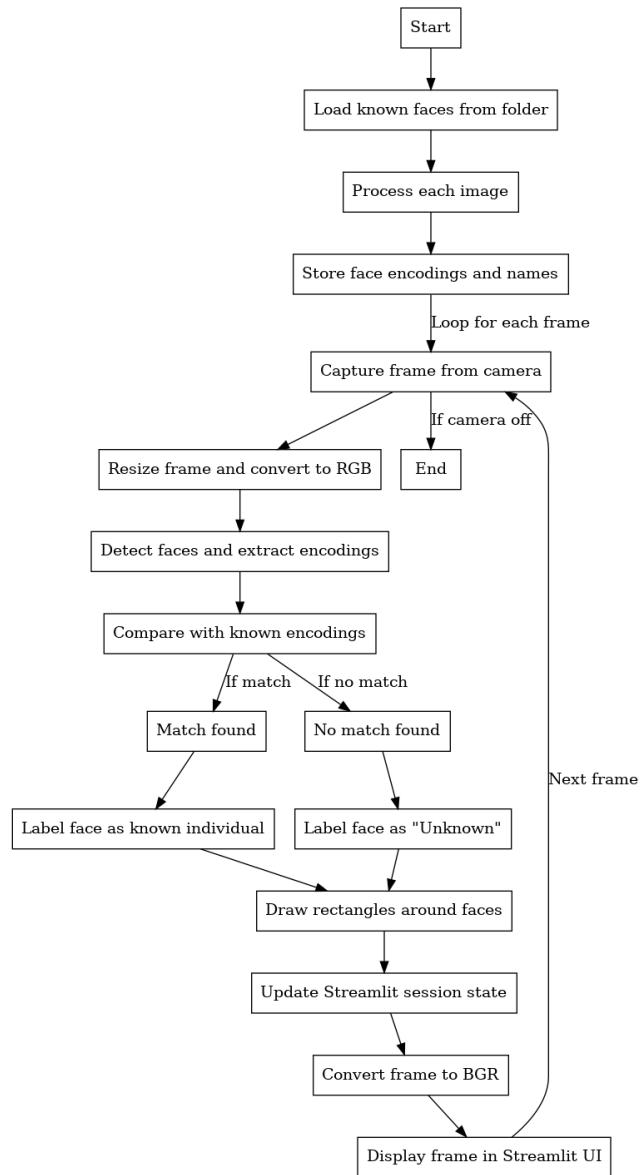


Figura 3: Flowchart Algoritmo de Face Recognition

## 4.2. Pose Detection

Dentro del campo de la visión computacional, la detección de pose humana ha experimentado avances significativos, ofreciendo diversas metodologías para su implementación. Los modelos basados en redes neuronales convolucionales, como OpenPose, ofrecen un enfoque de múltiples etapas que puede detectar con precisión puntos clave del cuerpo humano en imágenes y videos. Por otro lado, sistemas como PoseNet operan bajo un enfoque de una sola etapa, siendo más ligeros y rápidos, aunque a menudo con una precisión ligeramente menor. Estas metodologías varían en términos de complejidad computacional, precisión y capacidad de funcionar en tiempo real. Los modelos de alta resolución, como HRNet, han demostrado ser excepcionalmente precisos al mantener la información de alta resolución a lo largo de todo el proceso de la red, pero a costa de una mayor demanda de recursos computacionales.

Con la disponibilidad de estas tecnologías, se plantea un desafío en la selección de una herramienta que equilibre precisión, eficiencia y velocidad. Mientras que los modelos como HRNet ofrecen una precisión sobresaliente, pueden no ser idóneos para escenarios en tiempo real debido a sus requisitos de computación intensiva. En contraste, modelos más ligeros, aunque atractivos por su menor demanda de recursos, pueden comprometer la precisión y la granularidad de los datos de pose capturados, lo cual es crítico para nuestro objetivo de monitorizar la participación activa de los estudiantes en el aula.

En este contexto, nuestra elección se inclinó hacia el uso de YOLO (You Only Look Once) para la detección de pose. YOLO, conocido por su rapidez y eficiencia en la detección de objetos en tiempo real, ofrece una solución óptima para entornos dinámicos y en vivo, como un aula. Aunque originalmente diseñado para la detección de objetos, su adaptación al dominio de la detección de pose nos permite aprovechar su velocidad de procesamiento para identificar y rastrear las poses de los estudiantes de forma eficaz. La arquitectura de YOLO facilita un equilibrio entre velocidad y precisión, asegurando que la detección de gestos, como levantar la mano, se realice de manera fluida y confiable.

La decisión de implementar YOLO se sustenta en sus ventajas inherentes: su enfoque integrado reduce la latencia al procesar imágenes en una pasada, lo que es crucial para la captura en tiempo real de la dinámica del aula. Además, YOLO demuestra una gran resistencia a las variaciones en la iluminación y el escalamiento, manteniendo una alta tasa de reconocimiento incluso en condiciones no ideales, lo que a menudo se presenta en entornos educativos naturales. Estas características, combinadas con su robustez y su comunidad de desarrolladores activa, hacen de YOLO una elección acertada para nuestro sistema de detección de pose y seguimiento de participación estudiantil.

El algoritmo descrito en los anexos (7.2) realiza la detección de poses humanas y el reconocimiento facial en videos, utilizando el modelo YOLO (You Only Look Once) y la biblioteca de visión por computadora ‘cv2’. El proceso se inicia cargando un conjunto de rostros conocidos y codificándolos para su posterior identificación. A continuación, se analiza cada cuadro del video en busca de personas y sus poses. El algoritmo define dos funciones clave para interpretar las poses: ‘\_is\_raising\_hand’, que determina si una persona está levantando la mano basándose en la posición relativa de las manos con la nariz y los codos con el hombro, y ‘\_look4face’, que calcula la ubicación de la cara en el cuadro a partir de la posición de los hombros y la nariz. Estas funciones son cruciales para identificar momentos específicos, como cuando un alumno levanta la mano para participar.

A medida que el video se procesa, cada cuadro se examina para detectar poses y rostros. Cuando se identifica una mano levantada, el algoritmo verifica que haya pasado un tiempo mínimo desde la última identificación de participación para evitar duplicados. Luego, se extrae la región de la cara

y se envía a la función ‘recognize\_face’, que compara la cara detectada con las caras conocidas. Si se reconoce un rostro, se registra la participación del alumno correspondiente. El sistema también maneja la visualización en tiempo real, mostrando los cuadros procesados en una ventana y dibujando rectángulos alrededor de las caras detectadas. La ejecución continúa hasta que se interrumpe manualmente, y al finalizar, se liberan los recursos de video y se cierran todas las ventanas de visualización.

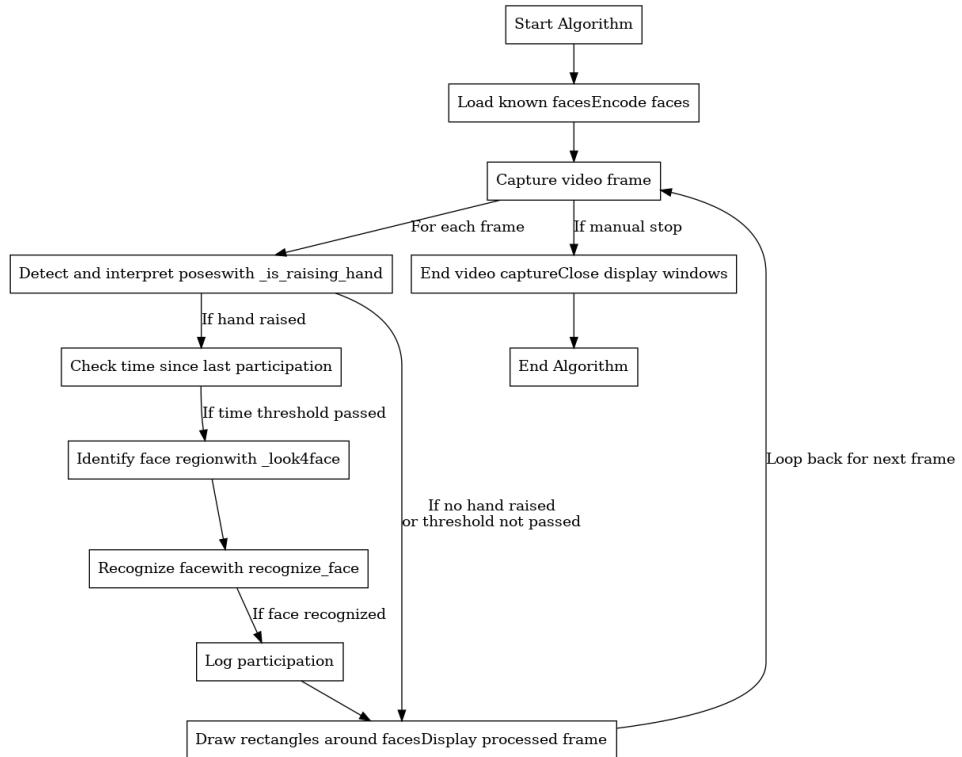


Figura 4: Flowchart algoritmo de Pose Detection

El algoritmo muestra una integración efectiva de detección de poses y reconocimiento facial para monitorear la participación en un entorno educativo, aprovechando la velocidad y eficacia de YOLO para la detección en tiempo real, y la precisión de ‘face\_recognition’ para la identificación de alumnos.

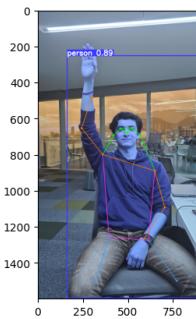


Figura 5: Demostración del algoritmo de Pose Detection

### **4.3. Interfaz Gráfica (Cómputo en la Nube)**

Cuando se trata de implementar una interfaz gráfica para webapps en la nube, el mercado ofrece diversas plataformas cada una con sus particularidades. Frameworks como Flask y Django son ampliamente reconocidos por su flexibilidad y control, permitiendo construcciones detalladas a nivel de código, pero pueden requerir una curva de aprendizaje empinada y una configuración extensa de servidores. Plataformas como Heroku y AWS Elastic Beanstalk facilitan el despliegue con infraestructuras menos complejas, aunque pueden implicar costos escalables y una configuración más involucrada para la optimización. Google App Engine proporciona una integración profunda con otros servicios de Google Cloud, pero su estructura de precios y la gestión de recursos pueden ser desafiantes para proyectos con recursos limitados.

Frente a estas opciones, Streamlit emerge como una solución excepcionalmente adecuada para la creación y el despliegue de nuestra webapp debido a su enfoque orientado a la ciencia de datos y su facilidad de uso. Streamlit permite convertir scripts de Python en aplicaciones web compatibles con una simplicidad sorprendente, lo que es ideal para la visualización de datos y la interacción con modelos de machine learning en tiempo real. La naturaleza intuitiva de Streamlit y su API de alto nivel posibilitan una rápida iteración y prototipado, cualidades esenciales para nuestro proyecto que requiere una interfaz de usuario interactiva y amigable. La decisión de utilizar Streamlit Cloud para el despliegue se basa en su integración directa con Streamlit, ofreciendo una transición fluida del desarrollo a la producción. Streamlit Cloud no solo proporciona una plataforma de despliegue sin complicaciones sino que también garantiza que las aplicaciones sean escalables y accesibles. Con Streamlit Cloud, se elimina la necesidad de gestionar infraestructuras de servidor o preocuparse por la configuración de entornos, lo que permite a nuestro equipo centrarse en mejorar la funcionalidad y la experiencia del usuario de la webapp.

Por ello, Streamlit, junto con Streamlit Cloud, ofrece una combinación poderosa de simplicidad, eficiencia y rendimiento. Esta elección metodológica nos permite presentar una interfaz gráfica centrada en el usuario y una solución de despliegue en la nube optimizada para nuestro sistema de reconocimiento facial y detección de pose, asegurando una solución robusta y confiable para la monitorización de la asistencia y la participación en el aula.

### **4.4. Base de Datos en la Nube**

Las opciones para el almacenamiento de tablas de datos en la nube son variadas, cada una con particularidades que se adaptan a diferentes necesidades y escenarios. Bases de datos relacionales como Amazon RDS o Azure SQL Database ofrecen soluciones robustas y escalables con amplio soporte para transacciones complejas, pero pueden ser excesivas para aplicaciones que requieren menor complejidad y una configuración más ágil. NoSQL databases como MongoDB Atlas proporcionan flexibilidad en el esquema y una fácil escalabilidad horizontal, aunque a veces a costa de la consistencia transaccional que caracteriza a los sistemas relacionales. Firebase de Google es otra opción popular, especialmente para aplicaciones móviles y web con necesidades de sincronización en tiempo real, aunque su modelo de precios y la gestión de datos pueden ser restrictivos a medida que la aplicación escala.

Frente a estas opciones, Deta Cloud (Deta Space) se destaca como una solución idónea para el almacenamiento de nuestras bases de datos. Deta Cloud ofrece un servicio de base de datos extremadamente simple y sin servidor, lo que elimina la necesidad de manejar la infraestructura de base de datos. Esto se alinea perfectamente con nuestra webapp, que requiere un almacenamiento de

datos eficiente y de bajo mantenimiento. La facilidad de uso de Deta Cloud y su modelo de precios accesible hacen que sea una opción atractiva para proyectos que buscan agilidad y economía, sin sacrificar la funcionalidad. La elección de Deta Cloud como nuestra solución de almacenamiento de datos se justifica por su enfoque en desarrolladores, ofreciendo una integración API sencilla que facilita la conexión con nuestra webapp de Streamlit. Además, Deta proporciona una escalabilidad automática que se adapta a nuestras necesidades de uso sin la necesidad de una gestión manual, permitiéndonos centrarnos en la mejora continua de la aplicación en lugar de en la administración de la base de datos.

En resumen, Deta Cloud emerge como la mejor opción para nuestro proyecto debido a su simplicidad, escalabilidad y costos predecibles. Estas características nos permiten ofrecer una solución de almacenamiento de datos que es tanto confiable como eficiente, asegurando que la información sobre asistencia y participación de los alumnos esté gestionada de forma segura y accesible.

#### **4.5. Plataforma de Almacenamiento en la Nube para Videos**

La elección de una plataforma de almacenamiento en la nube para videos es un aspecto crítico, especialmente cuando se trata de grabaciones de un entorno educativo. Servicios como Amazon S3 ofrecen soluciones escalables y duraderas para almacenamiento de datos, pero pueden resultar complejos en su configuración y en la gestión de permisos. Microsoft Azure Blob Storage proporciona almacenamiento de gran capacidad y una integración perfecta con otros servicios de Azure, aunque la implementación de medidas de seguridad y cumplimiento puede ser menos intuitiva. Por su parte, servicios como Dropbox o Vimeo ofrecen soluciones centradas en el usuario final con una interfaz amigable, pero pueden carecer de las capacidades de seguridad y personalización requeridas para manejar datos sensibles a gran escala.

Después de evaluar las opciones disponibles, Google Cloud Storage se identificó como la solución óptima para almacenar las grabaciones de nuestras clases. Google Cloud Storage sobresale por su seguridad avanzada y su enfoque en la privacidad, ofreciendo una infraestructura sólida que cumple con los estándares de ciberseguridad necesarios para proteger datos sensibles. Esto es de suma importancia en nuestro proyecto, donde la confidencialidad de las grabaciones de los estudiantes es primordial. Google Cloud Storage proporciona controles de acceso detallados, cifrado en tránsito y en reposo, y la compatibilidad con la gestión de identidades y accesos, lo que asegura que solo usuarios autorizados puedan acceder a los videos. Además, Google Cloud Storage se destaca por su escalabilidad, permitiendo que nuestra solución se ajuste a la cantidad de datos generados por las grabaciones de video sin incurrir en costos prohibitivos. Su modelo de precios basado en el uso y su red global de distribución de datos garantizan que las grabaciones se almacenen y se acceda a ellas de manera eficiente, lo que es crucial para el análisis y la revisión educativa.

La elección de Google Cloud Storage es estratégica no solo por su robustez tecnológica y flexibilidad operativa, sino también por su compromiso con la seguridad y la privacidad. Estas características nos permiten cumplir con las expectativas de ciberseguridad y proporcionar una plataforma de almacenamiento confiable y segura para las grabaciones de clases, asegurando la protección de la información de los estudiantes y la integridad del entorno educativo.

#### **4.6. ¿Big Data?**

Una consideración importante en la era actual de la tecnología es si tal sistema necesita incorporar soluciones de Big Data para su operatividad efectiva. Big Data se caracteriza por grandes

volúmenes, alta velocidad y variedad de datos, que requieren infraestructuras especializadas para su procesamiento.

En la fase actual del proyecto, el volumen de datos generado, si bien es considerable, no llega a la magnitud típicamente asociada con Big Data. Las grabaciones de video y los datos de asistencia, aunque se acumulan diariamente, están confinados a un conjunto manejable que no sobrepasa las capacidades de los sistemas de procesamiento de datos convencionales. Además, la velocidad de los datos, a pesar de ser en tiempo real, no es tan exigente como para requerir las tecnologías de procesamiento en tiempo real asociadas con flujos de Big Data. La variedad de los datos, centrada en imágenes y registros de asistencia estructurados, no implica la complejidad de fuentes heterogéneas que Big Data podría abordar. La veracidad y el valor, aunque críticos para garantizar la precisión y la utilidad de la información recopilada, no implican por sí solos la necesidad de soluciones de Big Data.

No obstante, es preciso considerar que con la expansión potencial del sistema a múltiples aulas o instituciones, donde los volúmenes de datos puedan multiplicarse rápidamente, la infraestructura de Big Data podría convertirse en una necesidad. En tal escenario, se requeriría de tecnologías capaces de manejar y analizar estos datos masivos para extraer insights y garantizar la escalabilidad del sistema. Por lo tanto, aunque en su estado actual el proyecto no se enmarca dentro de los parámetros de Big Data, la planificación para el futuro y la posible escalabilidad del sistema deben mantener una consideración abierta hacia estas tecnologías, en preparación para un crecimiento que podría llevar el proyecto a dimensiones donde Big Data se convierta en un componente integral.

## 5. Metodología

### 5.1. Estructura de la aplicación

Nuestro software se estructura en dos aplicaciones interconectadas, diseñadas para optimizar tanto la experiencia del usuario como la seguridad de los datos. La primera es una webapp alojada en la nube, accesible a través de la dirección <https://classroomassistantcloud.streamlit.app/>. Esta aplicación requiere un inicio de sesión y ofrece dos tipos de cuentas: alumno y maestro. Los alumnos tienen acceso a tres pestañas principales: una bienvenida a su grupo de clases, una lista de los miembros del grupo y un dashboard de estadísticas. Este último muestra el recuento de asistencias y participaciones por alumno y las asistencias distribuidas por días de la semana, ofreciendo también una tabla en tiempo real que se actualiza con los datos de asistencia del día en curso, gracias al reconocimiento facial ejecutado en la otra parte de la aplicación. Por otro lado, la cuenta de maestro en la webapp incorpora un formulario en la pestaña de lista de alumnos para registrar nuevos estudiantes y una pestaña adicional para el registro manual de asistencia y participación.

La segunda parte de la aplicación consiste en un ejecutable exclusivo para las computadoras de los maestros en el salón de clases. Esta decisión se tomó para mantener los datos sensibles en un dispositivo seguro y en una red confiable de la institución educativa. El ejecutable, que solo permite el acceso a cuentas de maestro, cuenta con funcionalidades similares a la webapp, pero incluye una pestaña adicional 'cámara'. Aquí, el maestro puede activar la cámara del aula y comenzar el reconocimiento facial durante los primeros 15 minutos de clase, periodo tras el cual se inicia la detección de poses para el seguimiento de participaciones. Cada rostro nuevo detectado se envía a la base de datos en la nube para registrar la asistencia, visible en tiempo real en el dashboard de la webapp. La detección de poses opera bajo una lógica similar, asociando cada participación al alumno correspondiente con un intervalo mínimo de 4 segundos entre participaciones para evitar duplicados.

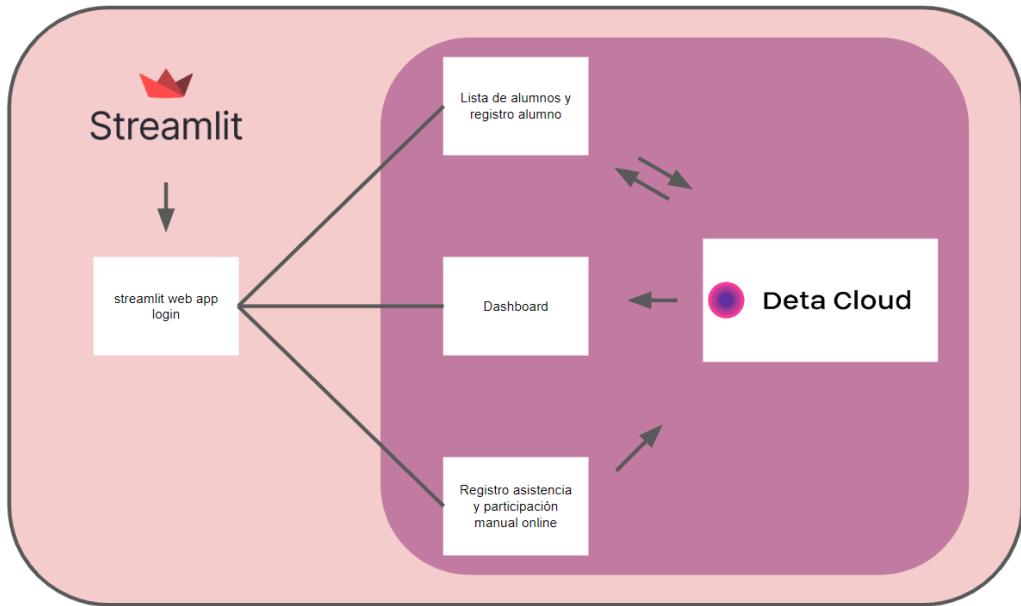


Figura 6: Estructura y componentes de la webapp

Al final de la clase, al seleccionar "terminar clase", el sistema sube automáticamente el recuento de participaciones a la base de datos en la nube y almacena el video completo de la clase en Google Cloud Storage por un periodo de 7 días. Esto ofrece una solución integral que, al finalizar la clase, provee al maestro una contabilización automática de asistencia y participaciones mediante visión computacional, garantizando así la eficiencia en la administración de la clase y la seguridad en el manejo y almacenamiento de datos sensibles.

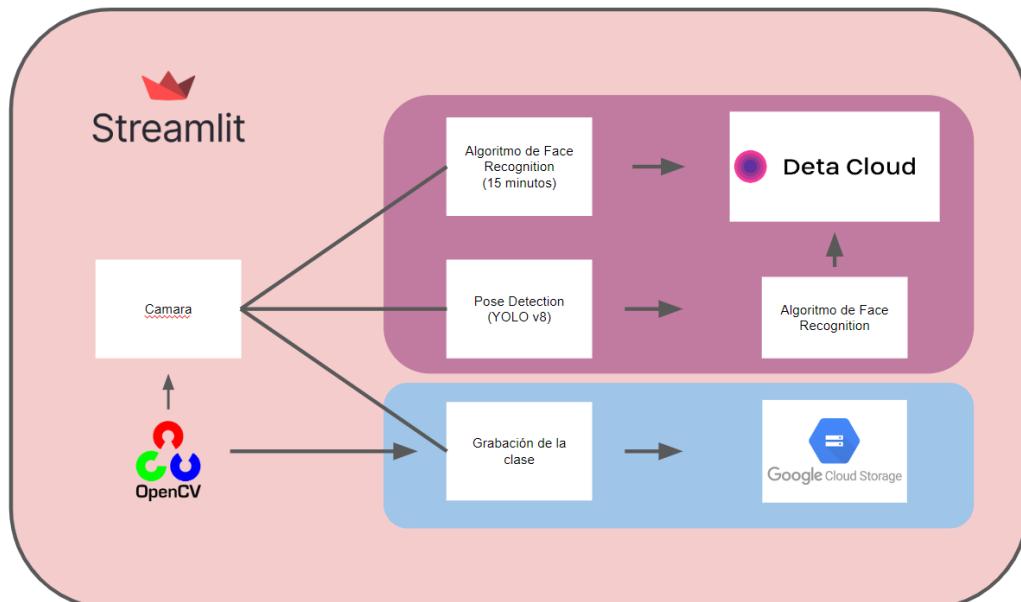


Figura 7: Estructura y componentes de la pestaña de cámara de la aplicación del maestro

## 5.2. Dataset

Para el desarrollo y la validación de nuestra aplicación de reconocimiento facial y de pose, se establecieron dos datasets esenciales: uno para el entrenamiento y otro para las pruebas. El dataset de entrenamiento se compone de una serie de fotografías que incluyen a los miembros del equipo de desarrollo y a otros individuos, seleccionados para proporcionar una amplia gama de características faciales. Este conjunto diverso es utilizado para entrenar al modelo de face\_recognition, con el objetivo de que aprenda a identificar y etiquetar con precisión las caras de los alumnos. En un escenario real de aula, este dataset de entrenamiento se transformaría en la base de datos de rostros de los estudiantes, permitiendo al modelo realizar un seguimiento efectivo de la asistencia en la clase. La



Figura 8: Ejemplos de dataset de entrenamiento para el reconocimiento facial

elección del dataset de pruebas fue crítica para garantizar la efectividad del sistema bajo condiciones operativas realistas. Se consideraron múltiples opciones, incluyendo renders generados en Unity con caras conocidas, y el uso de imágenes y videos de dominio público de aulas activas. Otra posibilidad explorada fue la creación de conjuntos de datos sintéticos mediante técnicas de aumentación de imagen para simular un mayor rango de variabilidad. No obstante, se tomó la decisión de emplear grabaciones en vivo de los miembros del equipo como el dataset de pruebas principal. Este enfoque se eligió para evaluar la aplicación en un espectro amplio de condiciones de iluminación, ángulos de cámara y entornos, proporcionando un conjunto de datos fiable y variado. Los renders de Unity

fueron descartados a pesar de su alta resolución, ya que no replican las condiciones reales de captura que se presentan en una cámara estándar de computadora de 720p.

En resumen, la utilización de grabaciones reales para las pruebas finales asegura que la aplicación sea probada en situaciones prácticas que reflejan el entorno educativo al que está destinada. Este enfoque práctico es fundamental para identificar y resolver posibles deficiencias y para validar la efectividad y la fiabilidad del sistema en condiciones reales de uso.

### **5.3. Pruebas de Face Recognition**

La fase de pruebas para el reconocimiento facial del sistema fue meticulosa, con especial atención en la iluminación y distancia. Se realizaron pruebas exhaustivas con las imágenes de cada individuo registradas en la base de datos de entrenamiento, simulando las condiciones en las que se encontrarían los alumnos en un salón de clases. Se varió tanto la intensidad lumínica como la distancia a la cámara para evaluar la precisión del reconocimiento facial en diferentes escenarios.

Respecto a la iluminación, se estableció que una iluminación óptima es crucial para el rendimiento del algoritmo. La precisión del reconocimiento facial mejora significativamente bajo una luz adecuada, mientras que situaciones de baja luminosidad aumentan la probabilidad de que el sistema clasifique erróneamente un rostro como "Desconocido". Esto subraya la importancia de contar con una iluminación suficiente en el área de trabajo para garantizar la efectividad del sistema.

En términos de distancia, los experimentos revelaron que, con una cámara estándar de 720p, el límite efectivo de reconocimiento es aproximadamente de 4 metros. Este alcance podría ampliarse con cámaras de mayor resolución y más megapíxeles, lo que indica que el hardware juega un papel crucial en la capacidad del sistema para identificar rostros a mayores distancias. Además, se ajustaron las métricas de comparación del algoritmo para mejorar la identificación. Primero, se empleó la similitud de coseno para comparar los encodings faciales, estableciendo un umbral de confianza del 85 %. Si no se logra una coincidencia, antes de clasificar el rostro como "Desconocido", el sistema intenta una comparación adicional utilizando la distancia euclíadiana entre los encodings. Solo si ninguna de estas medidas resulta en una coincidencia, el rostro se clasifica como "Desconocido".

Finalmente, se observó que en casos de individuos con rasgos muy similares, factores como la iluminación y la distancia son decisivos para que el sistema diferencie correctamente entre ellos. Estas pruebas resaltan la necesidad de considerar las condiciones ambientales para mantener la precisión del sistema, especialmente cuando se trata de diferenciar entre personas con gran parecido físico.

### **5.4. Pruebas de Pose Detection**

Las pruebas de reconocimiento de pose constituyeron un desafío significativo debido a la variedad de escenarios y combinaciones posibles que debían evaluarse. Inicialmente, se efectuaron pruebas individuales para asegurar que el sistema pudiera detectar cuando una persona levanta la mano y aplicar posteriormente el reconocimiento facial para identificar al individuo. Estas pruebas iniciales fueron superadas satisfactoriamente, incluso en condiciones de iluminación subóptimas. Se observó que, aunque el reconocimiento de pose tiene un mayor alcance que el reconocimiento facial (hasta 8 metros frente a los 4 metros del reconocimiento facial), surgían complicaciones cuando el sistema detectaba una mano alzada pero no lograba asociarla con una cara conocida. Como solución provisional, se decidió ignorar las participaciones que no se pudieran atribuir de manera clara a una cara identificada.

El desafío se intensificó al incorporar múltiples personas en el campo visual de la cámara. Uno de los problemas detectados fue la asignación errónea de participaciones cuando las muñecas de los



Figura 9: Pruebas de Reconocimiento Facial

sujetos no eran visibles en el cuadro, lo que interfería con el intervalo de 4 segundos establecido para diferenciar participaciones. Para resolver esto, se redefinió la lógica de la función `_is_raising_hand`, exigiendo que la muñeca estuviera visible por encima de la nariz y que el codo se encontrara por encima del hombro. Estos ajustes eliminaron las detecciones falsas y permitieron al sistema reconocer y asignar correctamente las participaciones reales. A medida que el número de personas en la cámara aumentaba, el modelo mantenía su rendimiento en cuanto a la detección de poses y el reconocimiento facial. No obstante, la velocidad de procesamiento disminuía, posiblemente debido a las limitaciones de la computadora utilizada para las pruebas, que no estaba equipada con una tarjeta gráfica dedicada. Se anticipa que con una computadora adecuadamente equipada, como se espera en un entorno de aula real, este problema no debería presentarse.

El éxito final de las pruebas fue la integración funcional del reconocimiento de poses y facial para hasta 5 personas simultáneamente, algunas participando y otras no. La aplicación pudo detectar e identificar con precisión quiénes y cuántas veces participaron, enviando esta información a la base de datos en la nube de Deta Cloud. Esta etapa confirmó la viabilidad operativa de nuestra solución, demostrando que la aplicación es capaz de manejar con eficacia las complejidades de un entorno de aula dinámico.

### 5.5. Repositorio e interfaz web

<https://github.com/DiegoElian02/StudentVisionClassIA>

<https://github.com/DiegoElian02/StudentVisionClassIA>

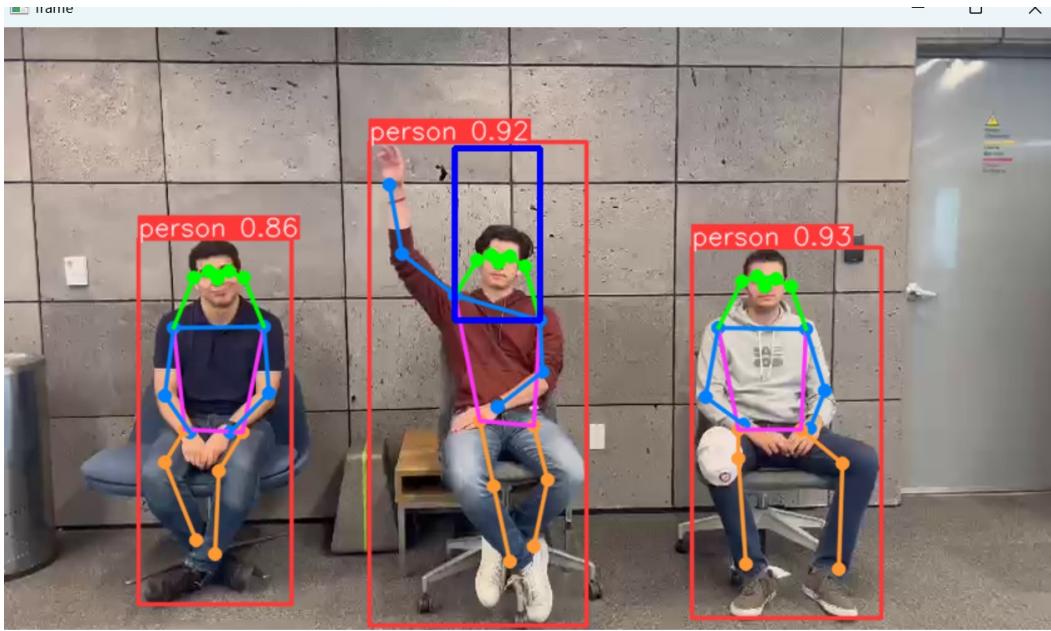


Figura 10: Pose Detection con alguien levantando la mano (se muestra Face Recognition)

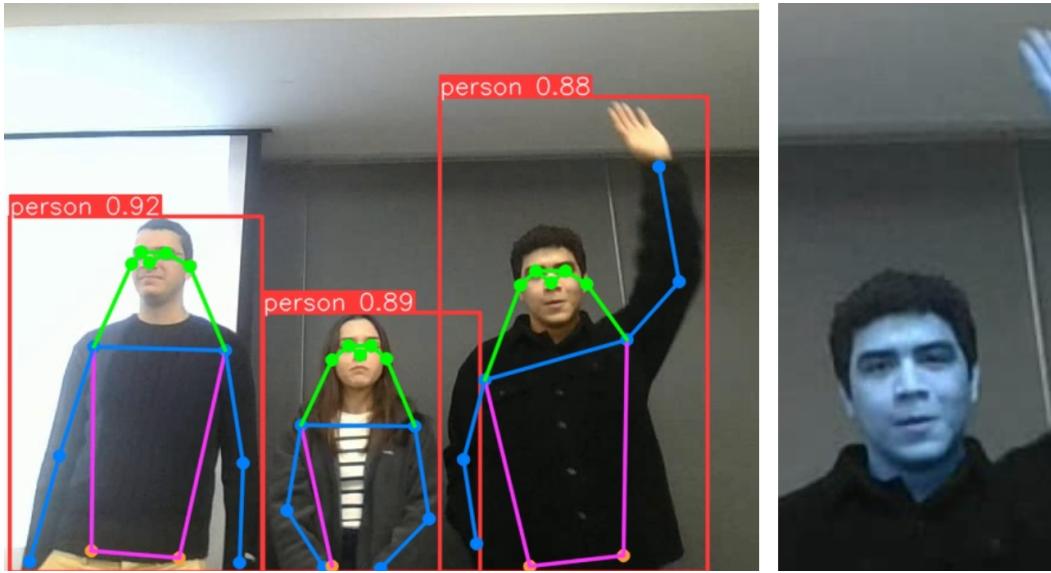


Figura 11: Pose Detection y recorte de cara de quien levanta la mano

## 6. Conclusión

El proyecto ha culminado con la creación exitosa de un sistema automatizado para la toma de asistencia y el conteo de participaciones en entornos educativos, empleando tecnologías avanzadas de reconocimiento facial y de pose. Hemos demostrado que, mediante el uso de modelos de inteligencia artificial como YOLO y herramientas de visión computacional, es posible mejorar significativamente la eficiencia y precisión en la gestión de aulas.

Nuestro sistema no solo facilita la labor administrativa de los docentes, sino que también proporciona datos valiosos que pueden contribuir a la mejora de la dinámica de clase. Con la capacidad de

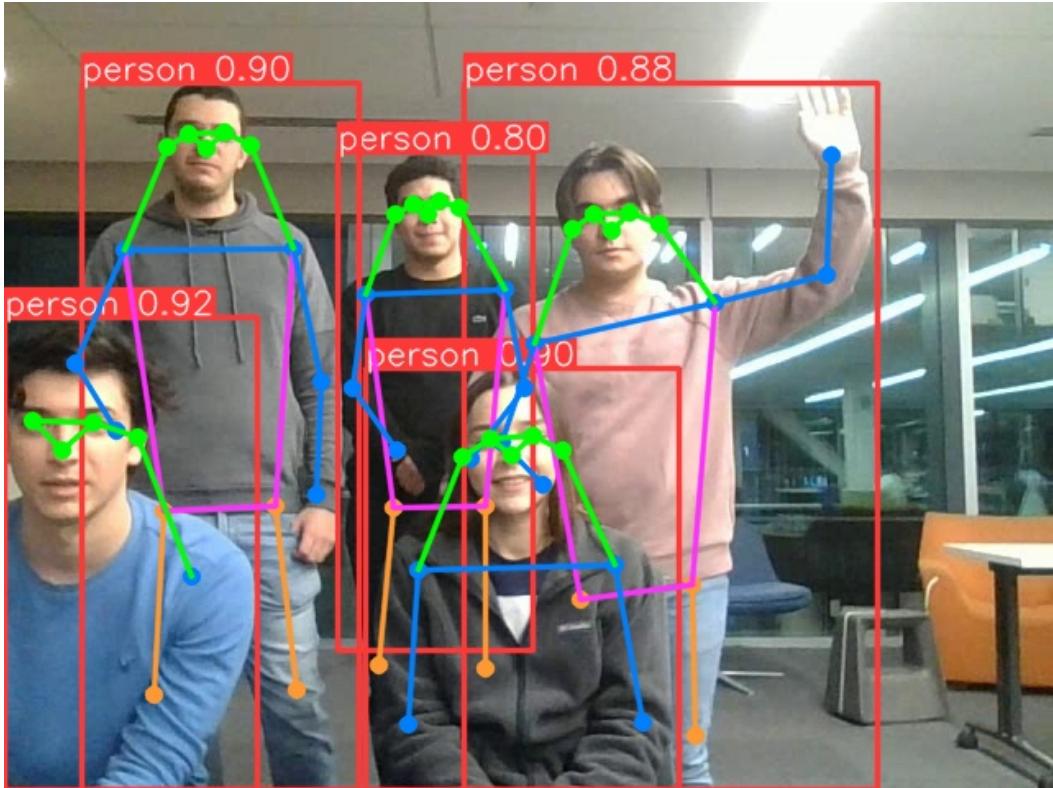


Figura 12: Pose Detection con 5 personas

register asistencias y participaciones en tiempo real, los educadores pueden obtener una perspectiva más clara de la implicación de los estudiantes y adaptar sus métodos de enseñanza para fomentar una mayor interacción y compromiso. A pesar de los avances realizados, se reconocen áreas de mejora. La precisión del reconocimiento facial disminuye con la iluminación subóptima y la distancia, lo que sugiere la necesidad de una iluminación adecuada en el aula y posiblemente el uso de cámaras de mayor resolución. Del mismo modo, aunque el reconocimiento de poses es efectivo hasta cierto punto, su rendimiento puede verse comprometido cuando se introduce un mayor número de participantes o cuando se enfrenta a limitaciones computacionales.

Para el futuro, se plantea la posibilidad de integrar hardware más potente y desarrollar algoritmos aún más robustos que puedan operar con una mayor variedad de condiciones ambientales y bajo restricciones técnicas más amplias. La adopción de este tipo de sistemas por parte de las instituciones educativas podría revolucionar la forma en que se llevan a cabo las tareas administrativas, permitiendo a los educadores centrarse más en la enseñanza y menos en la burocracia.

## 7. Anexos

### 7.1. Algoritmo de Reconocimiento Facial

```

known_face_encodings = []
known_face_names = []

for person in list_files_in_folder('known_faces'):
    
```

```
face = face_recognition.load_image_file(f"known_faces/{person}")
face_face_encoding = face_recognition.face_encodings(face)[0]

known_face_encodings.append(face_face_encoding)
known_face_names.append(person[:-4])

face_locations = []
face_encodings = []
face_names = []
process_this_frame = True
iter = 0
reconocidos = set()

f = fetch_alumnos()
alumnos = [alumno['nombre'] for alumno in f]
asistencia = {alumno: {"asistio": False, "fecha": None,
                      "participaciones": 0} for alumno in alumnos}

for alumno in reconocidos:
    if alumno in asistencia:
        st.session_state[alumno] = True

with col1:
    st.write("Camara en tiempo real con detección:")
    run = st.checkbox('Run Webcam')
    FRAME_WINDOW = st.empty()
    camera = cv2.VideoCapture(0)

    while run:
        _, frame = camera.read()
        if process_this_frame:
            small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
            rgb_small_frame = small_frame
            face_locations = face_recognition.face_locations(
                rgb_small_frame)
            face_encodings = face_recognition.face_encodings(
                rgb_small_frame, face_locations)
            face_names = []
            for face_encoding in face_encodings:
                matches = face_recognition.compare_faces(
                    known_face_encodings, face_encoding)
                name = "Unknown"
```

```

        face_distances = face_recognition.face_distance
                        (known_face_encodings, face_encoding)
        best_match_index = np.argmin(face_distances)
        if matches[best_match_index]:
            name = known_face_names[best_match_index]
            face_names.append(name)
        for (top, right, bottom, left), name in zip(
            face_locations, face_names):
            top *= 4
            right *= 4
            bottom *= 4
            left *= 4
            cv2.rectangle(frame, (left, top), (right,
                                              bottom), (0, 0, 255), 2)
            cv2.rectangle(frame, (left, bottom - 35), (
                right, bottom), (0, 0, 255), cv2.FILLED)
            font = cv2.FONT_HERSHEY_DUPLEX
            cv2.putText(frame, name, (left + 6, bottom - 6),
                        font, 1.0, (255, 255, 255), 1)
        for name in face_names:
            if name != "Unknown" and name in st.
                session_state:
                    st.session_state[name] = True
    # process_this_frame = iter == 0
    # iter = (iter + 1) \% 2
    process_this_frame = True
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    FRAME_WINDOW.image(frame)

```

## 7.2. Algoritmo de Detección de Pose

```

def _is_raising_hand(keypoints):
    nose = keypoints[0]
    left_hand = keypoints[9]
    right_hand = keypoints[10]
    right_elbow = keypoints[8]
    left_elbow = keypoints[7]
    right_shoulder = keypoints[6]
    left_shoulder = keypoints[5]

    if(
        (left_hand[1] < nose[1] and left_hand[1] > 0 and
         left_elbow[1] < left_shoulder[1]) or
        (right_hand[1] < nose[1] and right_hand[1] > 0 and
         right_elbow[1] < right_shoulder[1])
    )

```

```

    ):

        return True
    return False

def _look4face(keypoints):
    left_shoulder = keypoints[5]
    right_shoulder = keypoints[6]
    nose = keypoints[0]

    down = int(max(left_shoulder[1], right_shoulder[1]))
    left = int(left_shoulder[0])
    right = int(right_shoulder[0])
    up = int(down + 3*(nose[1] - down))

    return (left, up), (right, down)

def _list_files_in_folder(folder_path):
    file_names = []
    for file in os.listdir(folder_path):
        if os.path.isfile(os.path.join(folder_path, file)):
            file_names.append(file)
    return file_names

def recognize_face(frame, know_faces_dir = r'..\Face
Recognition\known_faces'):
    known_face_encodings = []
    known_face_names = []

    for person in _list_files_in_folder(know_faces_dir):
        face = face_recognition.load_image_file(know_faces_dir
+ f"/{person}")
        face_face_encoding = face_recognition.face_encodings(
            face)[0]

        known_face_encodings.append(face_face_encoding)
        known_face_names.append(person[:-4])

    face_locations = []
    face_encodings = []
    face_names = []

    # Find all the faces and face encodings in the current
    # frame of video
    face_locations = face_recognition.face_locations(frame)

```

```

face_encodings = face_recognition.face_encodings(frame,
    face_locations)

face_names = []
for face_encoding in face_encodings:
    # See if the face is a match for the known face(s)
    matches = face_recognition.compare_faces(
        known_face_encodings, face_encoding)
    name = "Unknown"

    # Or instead, use the known face with the smallest
    # distance to the new face
    face_distances = face_recognition.face_distance(
        known_face_encodings, face_encoding)
    best_match_index = np.argmin(face_distances)
    if matches[best_match_index]:
        name = known_face_names[best_match_index]

    face_names.append(name)

if face_names:
    return face_names[0]
else:
    return 'Unknown'

model = YOLO('yolov8n-pose.pt')
video = r'C:\Users\elias\Dropbox\Carrera\7mo Semestre\Blocke2\
    StudentVisionClassIA\PoseDetection\media\VideoRapido.mp4'
vid = cv2.VideoCapture(video)

fps = 30/4
delta_time_part = 5
time = 0
last_recog_time = -delta_time_part

def count_participations(video_path:str, known_faces_dir:str,
    fps:float= 30/4, delta_time_part:float = 5):

    participation_dict = {name[:-4]:0 for name in
        _list_files_in_folder(known_faces_dir)}

    model = YOLO('yolov8n-pose.pt')
    video = video_path
    vid = cv2.VideoCapture(video)
    counter = 0

```

```

time = 0
last_recog_time = -delta_time_part / 2

while vid.isOpened():
    ret , frame = vid.read()
    if not ret:
        break

    results = model.predict(frame, conf=0.65, show=False)
    no_show_frame = frame.copy()

    if results[0]: # Verificar si hay detecciones
        for person in results[0]: # Recorrer la lista de
            objetos (Personas) detectados

            keypoints = person.keypoints.xy.cpu().numpy()
            [0]

            if(_is_raising_hand(keypoints) and time -
               last_recog_time > delta_time_part):
                # last_recog_time = time

            left_up, right_down = _look4face(keypoints)
            face = no_show_frame[left_up[1]:right_down
            [1], right_down[0]:left_up[0]]

            # Convert the cropped face to RGB
            try:
                face = cv2.cvtColor(face, cv2.
                    COLOR_RGB2BGR)
                cv2.imwrite(f'Face{counter}.jpg', face)
                box_frame = results[0].plot(boxes=True)
                cv2.imwrite(f'Frame{counter}.jpg',
                           box_frame)
                counter += 1

                face_name = _recognize_face(face,
                                            known_faces_dir)

                print(face_name)

                if face_name != 'Unknown':
                    last_recog_time = time
                    participation_dict[face_name] += 1
            except Exception as e:

```

```

        print(f"Paso alguien. Error: {e}")

# cv2.imshow('frame', frame)

if cv2.waitKey(1) == ord('q'):
    break

time += 1/fps

vid.release()
cv2.destroyAllWindows()

return participation_dict

part_dict = count_participations(video_path = r'output.mp4',
                                  known_faces_dir=r'known_faces',
                                  fps=30/4,
                                  delta_time_part=6)
part_df = pd.DataFrame(list(part_dict.items()), columns=[
    'Nombre', 'Participaciones'])
st.table(part_df[part_df['Participaciones'] > 0])

```

## Referencias

- [1] V. Authors, “Student attendance with face recognition (lbph or cnn): Systematic ...” *ScienceDirect*, 2023. [Online]. Available: [www.sciencedirect.com](http://www.sciencedirect.com)
- [2] B. Roy, I. Hossen, M. G. Rashed, and D. Das, *Automated Student Attendance Monitoring System Using Face Recognition*, 02 2021.
- [3] V. Authors, “Efficient attendance management system based on face recognition,” *Springer*, 2023. [Online]. Available: [link.springer.com](http://link.springer.com)
- [4] ——, “Performance benchmark of deep learning human pose estimation,” *Springer*, 2023. [Online]. Available: [link.springer.com](http://link.springer.com)
- [5] T. Xu, J. Guo, M. Hong, and L. Bai, “A fast teacher pose estimation framework base on kernelized correlation filter and spatial transformed high-resolution network,” *Procedia Computer Science*, vol. 174, pp. 393–399, 2020, 2019 International Conference on Identification, Information and Knowledge in the Internet of Things. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877050920316252>
- [6] F. C. Lin, H. H. Ngo, C. R. Dow, K. H. Lam, and H. L. Le, “Student behavior recognition system for the classroom environment based on skeleton pose estimation and person detection,” *Sensors (Basel, Switzerland)*, vol. 21, no. 16, p. 5314, 2021. [Online]. Available: <https://doi.org/10.3390/s21165314>