University Student Control Detection System Based on Machine Learning and Artificial Intelligence

Joseph Lopez-Carreño, BE Electronic Engineering [1], Cristhian Calvo-Lavado, BE Electronic Engineering [1], Eliseo Zarate-Perez, PhDc, Mechatronic Engineering[1]

[1]Universidad Privada del Norte (UPN), Perú, N00165247@upn.pe, N00179564@upn.pe, eliseo.zarate@upn.edu.pe

Abstract*— The objective of this study was to program a school assistant based on artificial intelligence and integrated with surveillance cameras and loudspeakers in a university classroom. This was done to complement the supervision carried out by university teachers to identify student distractions, guarantee the proper use of masks, and evaluate student behaviors in class. The virtual assistant was developed using Python to generate audio warnings through a graphical interface built in the PyCharm environment. The results demonstrated the desired functionality of the virtual assistant, its ability to meet the requirements of a university classroom, and thereby the effectiveness of the YOLO v5 network and PyCharm used for the training, execution, construction, and implementation of the system.

Keywords*— school assistant, artificial intelligence, university education, teaching assistant, student monitoring.

Digital Object Identifier (DOI):
http://dx.doi.org/10.18687/LEIRD2022.1.1.178
ISBN: 978-628-95207-3-6 ISSN: 2414-6390

Sistema de Detección de Control de Estudiantes Universitarios Basado en Machine Learning e Inteligencia Artificial

Joseph Lopez-Carreño, BE Electronic Engineering [1], Cristhian Calvo-Lavado, BE Electronic Engineering [1], Eliseo Zarate-Perez, PhDc, Mechatronic Engineering[1]
[1]Universidad Privada del Norte (UPN), Perú, N00165247@upn.pe, N00179564@upn.pe, eliseo.zarate@upn.edu.pe

***Resumen* — El objetivo de esta investigación fue programar un asistente escolar basado en inteligencia artificial integrado con la estructura de cámaras de vigilancia y altavoces en un aula universitaria. Ello se realizó con la finalidad de complementar la supervisión que realizan los docentes universitarios para identificar las distracciones de los estudiantes, garantizar el uso adecuado de la mascarilla y evaluar comportamientos en clases. El asistente virtual se desarrolló utilizando Python para generar advertencias de audio a través de una interfaz gráfica construida con el entorno PyCharm. Los resultados demostraron la funcionalidad deseada del asistente virtual y su capacidad para cumplir con los requisitos de un aula universitaria. Por lo tanto, se demostró la efectividad de la red YOLO v5 y PyCharm, utilizados para el entrenamiento, ejecución, construcción e implementación del sistema.**

***Palabras clave*— Asistente escolar, inteligencia artificial, educación universitaria, asistente de cátedra, seguimiento estudiantil.**

I. Introducción

Actualmente, numerosas herramientas informáticas y tecnológicas permiten identificar eventos importantes para la humanidad. Algunas herramientas han abierto el camino a infinitas posibilidades de desarrollo, mejorando así la calidad de vida en la sociedad en general. La aplicación de la inteligencia artificial (IA) en la educación es una de esas vías para el desarrollo [1]. De tal forma, la IA en la educación se está tornando de mucha importancia en el desarrollo de sistemas inteligentes que contribuyen significativamente con la educación [2].

El Consenso de Beijing sobre IA y Educación confirmó el compromiso asumido en la Agenda 2030 para el Desarrollo Sostenible. En particular, se reiteró el cuarto Objetivo de Desarrollo Sostenible (ODS) y sus metas para promover respuestas políticas adecuadas en el logro de la integración sistemática de la IA en la educación [3]. El consenso afirmó que la integración de la IA en la educación debería considerar las innovaciones de la enseñanza y el aprendizaje. Por lo tanto, la IA puede acelerar la creación de sistemas educativos abiertos y flexibles que faciliten oportunidades de aprendizaje permanentes, equitativas, relevantes y de alta calidad para todos.

En este contexto, las universidades y centros de enseñanza de todo el mundo han aceptado que las aplicaciones de tecnologías inteligentes basadas en IA, como tutores inteligentes, sistemas de gestión del aprendizaje y videojuegos contribuyen significativamente al sector [4]. Sin embargo, la adopción de tecnologías basadas en IA ha sido lenta en algunas instituciones debido a la falta de financiamiento y, principalmente, a la ausencia de análisis rigurosos que revelen la relevancia de dichas tecnologías para el sector educativo [5].

Además de ello, la conducta del alumno ha cambiado, al igual que la concentración en su formación presencial después del cambio inesperado por la crisis sanitaria, ambos elementos se han visto perjudicados debido a los diferentes distractores tecnológicos, como factores personales. Ello genera un gran reto para los maestros, quienes se deben valer de presentar el contenido de la clase, generando expectativa, mediante diferentes recursos y actividades para que los educandos estén atentos la mayor parte del tiempo [6]. Es por ello, se espera que los recientes despliegues tecnológicos de IA puedan contribuir para disminuir el déficit de atención académico, monitorear la actitud de los estudiantes y complementar las estrategias de enseñanza - aprendizaje [7].

En tal sentido, una aplicación basada en IA podría ayudar a identificar el comportamiento indiferente de los estudiantes universitarios en clases presenciales, posterior a la nueva normalidad. Por lo tanto, la creación de un asistente escolar para ayudar a los maestros será capaz de monitorear la conducta y el uso del equipo de bioseguridad de los estudiantes haciendo uso de una interfaz gráfica. En tal sentido, este estudio tuvo como objetivo programar un algoritmo de asistente escolar en el uso de supervisión para estudiantes universitarios. En consecuencia, el algoritmo se integró en la arquitectura del aula para identificar a los estudiantes mediante la utilización de una cámara de seguridad. La aplicación facilitó advertencias de audio a través de un altavoz, o mediante la interfaz gráfica al identificar un comportamiento inadecuado.

II. Metodología

La metodología implementada se basó en el concepto de niveles de preparación tecnológica o *Technology Readiness Levels (TRLs)*. Este se basa en validar las pruebas realizadas a cada componente a través de un laboratorio virtual o interfaz de simulación, demostrando su funcionamiento en su conjunto para determinar la capacidad de utilización [8]. En consecuencia, se utilizó el software de modelado 3D SketchUp en el desarrollo del diseño de un ambiente de aula, el cual se incorporó dentro de la estructura principal del asistente universitario.

Además, se utilizó *Google Colab* para programar y ejecutar la simulación del modelo de IA en Python. De manera similar, se logra la verificación del asistente escolar utilizando la interfaz gráfica de usuario mediante el editor de texto inteligente PyCharm. Esta herramienta se selecciona comparando las diferentes opciones de entornos avanzados de desarrollo, en el cual mediante una matriz de entrada-salida se evaluaron las características técnicas para determinar la herramienta acorde según por la norma ISO 25.000 [9].

A. Herramientas Aplicadas

1) Makesense.ai: Es una herramienta en línea gratuita orientada al etiquetado de imágenes para su respectiva detección y reconocimiento. Esta herramienta facilita la selección de los nombres de las clases y estudiantes mediante un proceso óptimo con base en el conjunto [10]. Puesto que el rendimiento del algoritmo está claramente relacionado con la calidad de las imágenes entrenadas, ello facilita la preparación de una base de datos con imágenes etiquetadas. En este programa, las imágenes fueron examinadas secuencialmente y los sujetos existentes se seleccionaron con cuadros delimitadores de acuerdo con la distribución y clase correspondiente.

2) YOLO v5: Su sigla proviene del inglés You Only Look Once (YOLO) que es un framework de detección de objetos en tiempo real de código abierto. Este nos proporciona una red neuronal convolucional; basada en Darknet, ya estructurada para la detección de objetos en imágenes o videos [10, 11]. Existen diferentes versiones de este framework, en abril del 2020 se presentó una nueva versión Yolo v5 creada por Glenn Jocher. En esta versión se encuentran 4 diferentes modelos de acuerdo a los pesos, el ancho y la profundidad de cada modelo aumentado secuencialmente: YOLO v5S, YOLO v5M, YOLO v5L, YOLO v5X [13]. Debido al excelente rendimiento en tiempo real de los algoritmos de YOLO, se utilizará el modelo de mayor precisión. De esa forma, se utiliza YOLO v5X para personalizarlo con nuestro conjunto de datos y realizar el nuevo entrenamiento y validación de la detección de imágenes de los modelos.

3) Google Colab: Es una plataforma basada en el Notebook de Jupyter que permite la escritura y ejecución de instrucciones o programas en varios idiomas en una interfaz de navegador. De la misma forma, esta herramienta permite un servicio de almacenamiento en la nube a través de Google Drive. Además, esta plataforma permite realizar funciones de aprendizaje automático, aprendizaje profundo, análisis de datos, visión artificial y otras aplicaciones centradas en GPU [14]. Considerando las funciones de este programa, se puede utilizar fácilmente para ejecutar el código del modelo YOLO v5 para la detección del comportamiento de los alumnos y equipos de protección personal. De igual forma, el modelo brinda gráficos que exponen las pérdidas de entrenamiento y el porcentaje de precisión en las métricas de rendimiento de los modelos entrenados.

4) PyCharm: Este es un entorno de desarrollo integrado (IDE) que permite la edición de texto, la depuración y la interpretación completa del código en Python. Asimismo, esta herramienta ofrece las funciones de finalización e inspección del código. De la misma forma, esta herramienta realiza la indicación de errores sobre la marcha, arreglos rápidos, refactorización de código automático y completas funcionalidades de navegación [15]. Adicionalmente, IDE cuenta con una consola de Python interactiva que es compatible con la distribución Anaconda e incluye diferentes librerías y paquetes científicos, como Matplotlib y NumPy [16]. Este software permitió la verificación y validación del modelo ejecutado en Google Colab. Es decir, se utilizó para perfeccionar las configuraciones y acciones a realizar (considerando las pruebas de video y su respuesta a la aplicación), así como la interfaz gráfica donde se observan los botones de las funciones, configuraciones y activación.

5) SketchUp: Es un programa de modelado y diseño gráfico tridimensional (3D). Este es el software principal elegido por profesionales de distintas áreas para dibujar de forma sencilla o descargar modelos [17]. Además, el software permite al usuario diseñar y generar espacios, ya que proporciona funciones específicas de dimensionamiento, visualización, simulación y renderizado en 3D. Mediante esta herramienta se simuló la estructura del asistente escolar implementado de un aula de clases para dar una perspectiva general de la integración de ambas partes.

B. Recopilación de datos de imágenes

El conjunto de datos utilizados en esta investigación consiste en una base de datos personalizados. Esta herramienta almacena imágenes relacionadas con comportamientos en los estudiantes como estar distraído, atento o dormido en el aula de clases. Asimismo, permite identificar diferentes rostros de estudiantes universitarios con/sin mascarillas. La recolección de imágenes se formó con base en múltiples palabras clave en diferentes recursos como: explorar en motores de búsqueda, hacer capturas de pantalla en videos y acudir a webs con bases de datos previamente preparados. Estos datos contienen ciertas características que se pretenden estudiar y están delimitados por criterios de inclusión y exclusión, tal como lo menciona [18]. En este sentido, el presente estudio expone los criterios de inclusión y exclusión.

Criterios de inclusión: Imágenes que conserven una buena cualidad, entre mediana y alta calidad; fotografías que guarden un determinado tamaño de píxeles; visualizaciones digitales preferiblemente en escala a grises.

Criterios de exclusión: Imágenes que no dispongan de buena calidad; fotografías que no guarden el tamaño deseado de píxeles; visualizaciones digitales antiguas o borrosas.

Una vez que se recolectaron las imágenes preliminares, se seleccionaron de una forma manual los datos redundantes para eliminarse debido a la superposición de los resultados al emplear múltiples motores de búsqueda, quedando una base de datos con 5500 imágenes. Posteriormente, estas imágenes fueron repartidas de acuerdo con las clases que detectarán, correspondiéndole 3300 para el Modelo de detección de comportamiento y 2200 datos para el Modelo de detección de

mascarillas (Tabla 1). Para ambos modelos, se distribuyó el 70 % de los datos para el entrenamiento y el 30 % para realizar las pruebas [19].

Cada uno de los conjuntos de datos se utilizaron para efectuarse el etiquetamiento del material utilizando la web Makesense.ai. Esta herramienta examina las imágenes secuencialmente, seleccionando el comportamiento de los sujetos existentes con cuadros delimitadores y tomando toda la porción del objeto a estudiar. Asimismo, se asignó una adecuada denominación a las etiquetas que guarden relación con las clases en el Modo Supervisión. Para ello, se emplearon tres etiquetas diferentes, "Distraído", "Dormido" y "Atento". Después de tener todas las imágenes procesadas, se exportaron sus respectivas coordenadas para entrenar el modelo del asistente escolar.

TABLA I
DISTRIBUCIÓN DE LA BASE DE DATOS

Modelo de detección de Comportamiento	
Base de Datos	**Cantidad de imágenes**
Entrenamiento (70 %)	2310
Prueba (30 %)	990
Total	**3300**
Modelo de detección de Mascarillas	
Entrenamiento (70 %)	1540
Prueba (30 %)	660
Total	**2200**

C. Programación del algoritmo del asistente escolar

El entorno interactivo de Google Colab se utilizó para la codificación, iteración y entrenamiento del modelo de detección de objetos e imágenes. De la misma forma, YOLO v5 en la programación del algoritmo (Fig. 1). Estas herramientas permitieron ajustar los parámetros del modelo, así como almacenar un sólido conjunto de datos de las imágenes previamente etiquetadas. Posteriormente, el 70 % de las imágenes se utilizaron para realizar el entrenamiento de la red YOLO v5 hasta lograr realizar la convergencia con iteraciones sucesivas. Después, se empleó el editor de texto PyCharm para verificar el modelo utilizando el 30 % restante de las imágenes.

En consecuencia, si la precisión del modelo no superaba el 80 % de precisión, se proponía un cambio en el conjunto de datos que se utilizó para entrenar la red YOLO v5 y se repetía el proceso de entrenamiento. Sin embargo, si la precisión de la variable estaba entre el 80 % y el 93 %, el modelo neuronal se reentrenaba para lograr la convergencia. Finalmente, si el resultado era favorable, se validaba el algoritmo analizando la transmisión en tiempo real de la cámara de vigilancia.

D. Materiales y tecnologías desplegadas

Los altavoces conectados se encargaban de reproducir las frases en forma de sonido mediante impulsos eléctricos enviados por la computadora [20]. Estas frases se agruparon en categorías para que los estudiantes pudieran conceptualizar y comprender su significado y utilidad [21]. Este sistema externo fue fijado a la pared y se utilizó el audio grabado en el modelo de asistente escolar como advertencia para disuadir a los estudiantes de cometer actos prohibidos por la normativa de la institución educativa.

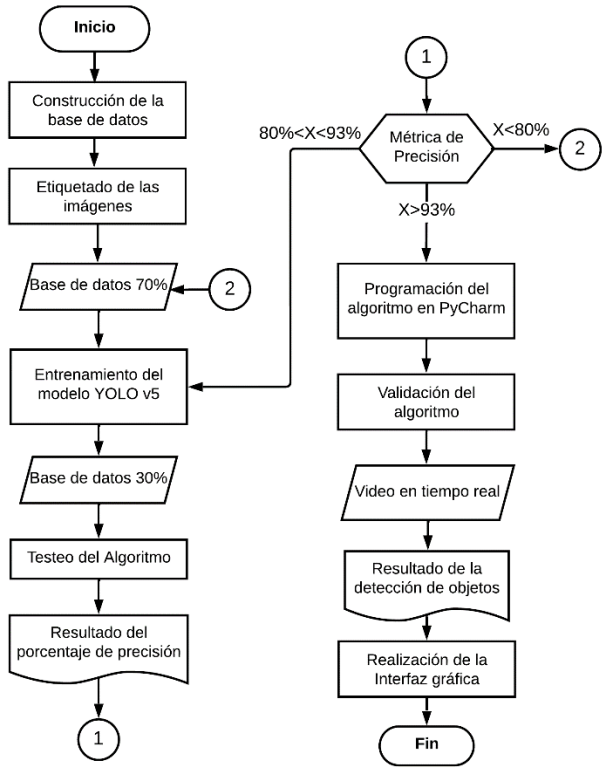

Fig. 1 Diagrama de flujo del algoritmo de detección de imágenes del asistente escolar.

La cámara de seguridad se encargaba de registrar imágenes de cualquier evento que pudiera haber ocurrido dentro o fuera de un espacio. También es responsable de monitorear las actividades de visitantes y mantener un registro audiovisual de los eventos [22]. Finalmente, la computadora jugó un papel fundamental en la arquitectura de red del asistente escolar al funcionar como un intermediario entre la interfaz gráfica y el hardware, compuesto por la cámara de seguridad fija y los parlantes [23]. La aplicación instalada en el ordenador respondía a las incidencias captadas por la cámara, enviaba mensajes correspondientes a los parlantes y brindaba explicaciones al docente a través de la interfaz gráfica (Fig. 2).

E. Matriz de selección del software

Para la elección del entorno de desarrollo integrado se realizó una búsqueda para identificar el software especializado en el entorno de codificación de Python que mejor se adaptaba a los requisitos de rendimiento del sistema de asistencia escolar. Se elaboró y segmentó un cuadro comparativo con base a la calidad del software y calidad en uso basado en la norma ISO 25 000. Esta norma se relaciona con el proceso de evaluación de la calidad del software [9] y contempla 13 características con 41 sub-características. En consecuencia, se considera el factor económico, facilidad de uso, operabilidad, programación, entre

otros. Además, se consideró el valor de 5 como el puntaje máximo que se puede otorgar a cada sub-característica. Por lo tanto, el valor máximo que se puede obtener en la evaluación de desempeño es de 205 puntos.

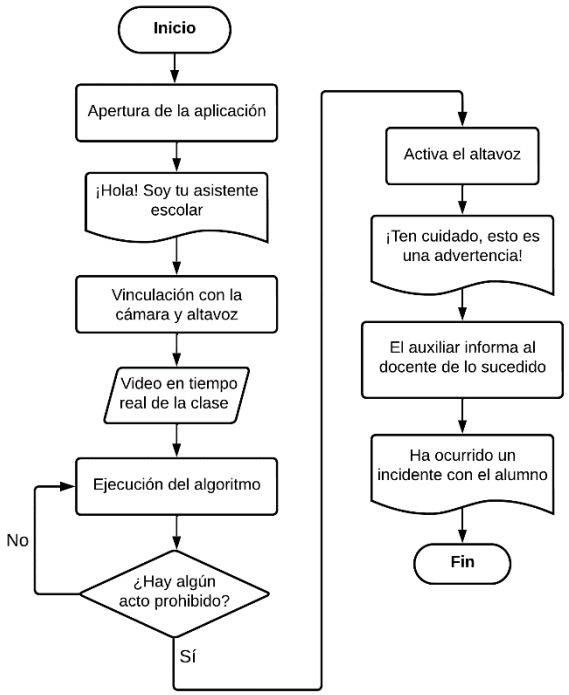

Fig. 2 Diagrama de flujo del proceso del asistente escolar integrado al aula de clase.

III. Resultados y Discusiones

A. *Diseño de la interfaz gráfica*

Se tuvo en consideración el diseño de una interfaz gráfica de usuario (Fig. 3) sencilla y atractiva en la que el docente pueda visualizar las funciones a elegir. Uno de los botones se empleó para activar cualquiera de las funciones previamente seleccionadas que dispone el algoritmo del asistente escolar desarrollado, en el cual aparece inmediatamente la transmisión en tiempo real de la cámara de vigilancia con su sistema de detección. Esa interfaz busca identificar en cuadros delimitadores la presencia de personas. Un botón adicional es para terminar la ejecución de la función, finalizando la transmisión de la cámara.

De esa forma, se tuvo en cuenta un botón adicional; Opciones, ubicado en la ventana principal de la interfaz gráfica, el cual permite abrir un pequeño panel de control simple e interactivo (Fig. 4). Donde se observan algunas configuraciones generales como el cambio de fondo e Idioma, en los cuales se tienen diferentes elecciones mediante un menú desplegable. También se observan botones del tipo *toggle* (conmutación), donde se utiliza la activación de la música del tipo instrumental para acompañar y permitir la concentración en la clase. La búsqueda de actualización mostrará un enlace al ser activada, por donde se podrán descargar las últimas actualizaciones del asistente escolar. Al ser activada la opción "Sugerencias al iniciar" se mostrará una serie de pasos básicos para el uso correcto del asistente virtual al abrir esta aplicación. Finalmente, en la zona baja se contará con una sección donde se mostrará la versión instalada y los datos de contacto acerca de los autores.

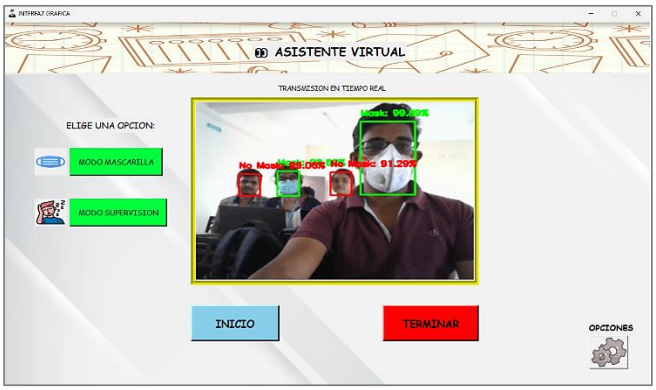

Fig. 3 Interfaz gráfica activa, cuando inicia la detección.

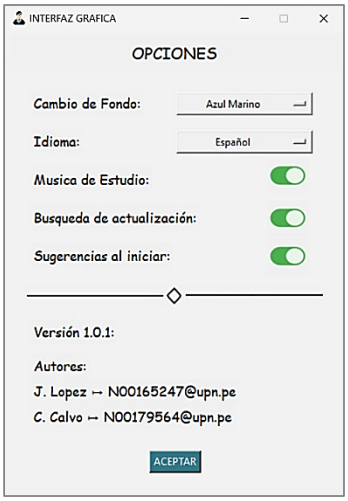

Fig. 4 Ventana opciones de la interfaz gráfica.

B. *Visualización del sistema con SketchUp*

Para desarrollar la visualización del aula de clase universitaria integrada con la estructura del asistente escolar se utilizó un software de modelamiento paramétrico en 3D basado en caras llamado SketchUp, versión 2022, como se detalla a continuación.

1) Distribución de los dispositivos: Como se puede observar en la Fig. 5, el sistema se compone de dos altavoces y una cámara de seguridad que están instalados en la zona superior de la pizarra, obteniendo una mejor vista de los hechos. En esta ubicación la cámara de seguridad tiene una iluminación adecuada y la libertad de realizar funciones de movimiento en diferentes ángulos en el eje horizontal o vertical. Ello le permite realizar mejor el seguimiento del comportamiento de los estudiantes en tiempo real. Así también, los altavoces están situados idealmente para direccionar la potencia del audio,

asegurando de que el mensaje llegue sin distorsión a la audiencia de la clase.

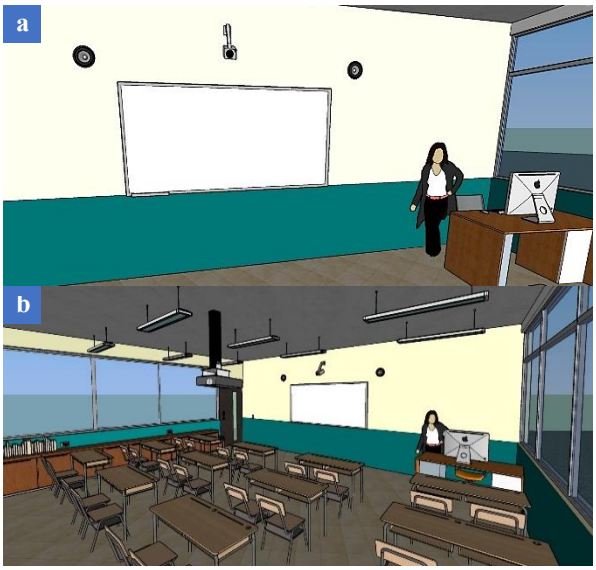

Fig. 5 Visualización del sistema. (a) Ubicación de los dispositivos. (b) Proyección del asistente escolar integrado al aula.

C. *Código de programación*

TABLA II
CÓDIGO DE PROGRAMACIÓN EN PYCHARM

Codificador PyCharm

#Importamos Líbrerias
import torch # Librería para Deep Learning
import cv2 # Librería para Visión Artificial
import numpy as np #Líbreria para manejar vectores y matrices
#Lectura del Modelo
model= torch.hub.load('ultralytics/yolov5','custom',
path='C:/Users/USUARIO/Desktop/Video/model/Mask2.pt')
#Realizamos la Videocaptura
cap= cv2.VideoCapture(0) #Búsqueda de la camara a utilizar
#Empezamos
while True: #Inicialización del bucle infinito
#Realizamos lectura de la videocaptura
ret, frame= cap.read() #Captura del video cuadro a cuadro
#Realizamos detección
detect=model(frame)
#Mostramos FPS
cv2.imshow('Deteccion de Mascarilla',
np.squeeze(detect.render()))
#Leer el teclado
t=cv2.waitKey(5)
if t== 27:
break #Rompe el bucle infinito
cap.release() #Libera la transmisión de video
cv2.destroyAllWindows() #Cerrar todas las ventanas

La Tabla II informa el código de programación de nuestro algoritmo propuesto realizado en el software PyCharm. Este describe el funcionamiento del sistema inteligente y se encarga de supervisar continuamente a los alumnos. Además, supervisa el comportamiento de los alumnos y verifica el uso de equipos de protección individual por parte de los alumnos en clase.

D. *Métricas de rendimiento*

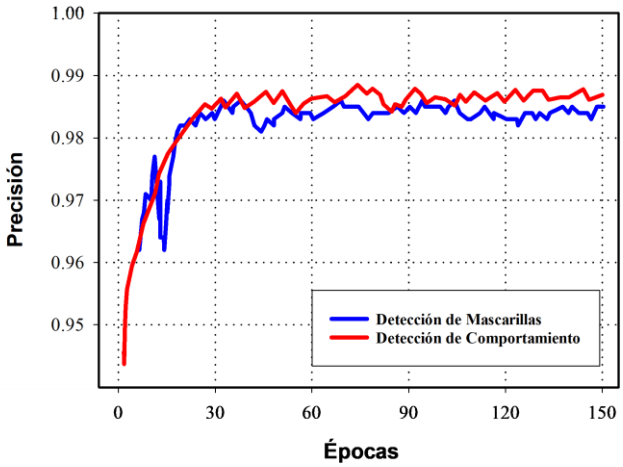

Fig. 6 Desempeño de los modelos de detección de mascarillas y comportamiento de estudiantes.

En esta sección se presentan los resultados de las métricas de rendimiento basados en el modelo de detección de comportamiento e identificación de equipos de protección. La Fig. 6 muestra los resultados del desempeño de precisión basándose en los modelos evaluados a lo largo de las épocas. Es decir, una época es un ciclo de entrenamiento cuando el algoritmo analiza todas las muestras de entrenamiento. Además, se realizó un ajuste de parámetros de entrenamiento para ambos modelos, donde se indicó 150 épocas, un tamaño de Batch de 16 y se utilizó el modelo más robusto YoloV5x. Con todo ello, se puede verificar mediante la gráfica que los modelos de detección de comportamiento de los estudiantes y detección de equipos de protección (mascarillas) demuestran que (partir de la época 90) hay un desempeño en la precisión superior a 0.98. Ello queda reflejado en porcentajes a una certeza superior al 98 % indicando un buen desempeño en los modelos propuestos.

En la Fig. 7 se presenta los resultados del rendimiento del parámetro pérdidas con base a los modelos definidos. El modelo de detección de comportamiento muestra ambas curvas de pérdidas relacionadas con los datos del entrenamiento, que representa a los errores frente a información conocida; como los datos de validación, que representa a los errores frente a información nueva. Se observa que se inicia con una pérdida importante de 0.060 y que posteriormente con la validación del modelo se va reduciendo de forma logarítmica el error de la detección a medida que se realizan más iteraciones de las épocas. Por medio del conjunto de datos necesario se evita tanto el "*overfitting*"; es decir, un sobre entrenamiento. De la misma

forma, se puede evitar el "*underflitting*", es decir un entrenamiento insuficiente. Logrando un entrenamiento apropiado que alrededor de las 120 épocas; el modelo tiende a estabilizar su rendimiento de detección, como se muestra en la Fig. 7 (a).

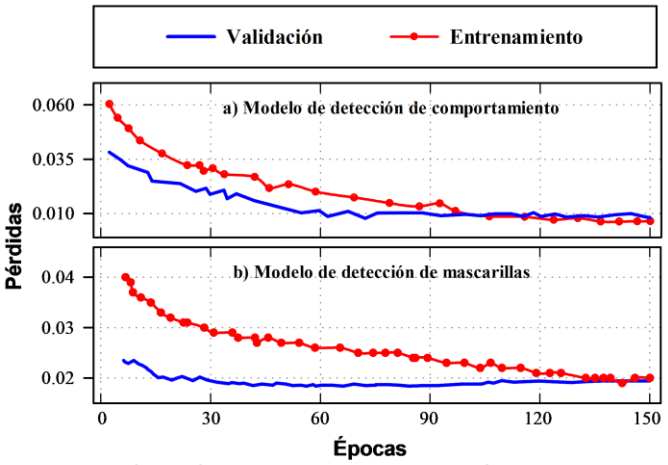

Fig. 7 La métrica Pérdidas de los modelos de detección de mascarillas y comportamiento de alumnos

El modelo de detección de mascarilla muestra de igual manera ambas curvas de pérdidas tanto de los datos del entrenamiento, como los datos de validación. Se observa que la curva de entrenamiento inicia con una pérdida significativa de 0.04 y por medio de la selección de una correcta base de datos se evita el "*overfitting*" y el "*underflitting*, obteniendo un entrenamiento apropiado que convergen con el valor de la curva de validación alrededor de las 120 épocas. Con ello se obtiene una pérdida de 0.02, ligeramente más alta en comparación con el modelo de detección de comportamiento de alumnos, expuesto en la Fig. 7 (b). Esta diferencia se debe a que ambos modelos son evaluados con una cantidad diferente de conjunto de datos, tanto en el entrenamiento como en la validación. De tal forma, siendo más robusta la base de datos destinado a la detección de conductas prohibidas y consiguiendo que dicha curva disminuya logarítmicamente.

E. Matriz de elección del IDE según ISO 25.000

La Tabla III enumera las características que influyeron en la elección del software. Según los requisitos de los usuarios, PyCharm resultó ser el entorno de desarrollo óptimo. PyCharm tuvo un 87,5 % de confiabilidad, lo que indica que tiene características superiores en comparación con los demás entornos mencionados en la matriz.

Los resultados indican que el sistema inteligente basado en visión artificial generó respuestas correctas porque tuvo una selección óptima en las secciones de entrenamiento y pruebas realizadas. Esto confirma los hallazgos de [24] de que al aumentar el número de sesiones de entrenamiento y la cantidad de datos, los parámetros de la red pueden optimizarse continuamente para mejorar la precisión de la clasificación.

Además, la capacidad de detección del sistema inteligente demostró que se obtuvieron resultados precisos al implementar el algoritmo en Python usando PyCharm como IDE. Un estudio previo [25] confirma que el IDE utilizado para programar este entorno de programación es específico del lenguaje Python.

TABLA III
CARACTERÍSTICAS PARA LA ELECCIÓN DEL SOFTWARE

Calidad del software					
Características	**Sub características**	**Spyder**	**PyCharm**	**Atom**	**Vs. Code**
Funcionalidad	Operabilidad	4	5	4	4
	Corrección	3	4	3	3
	Idoneidad	4	5	4	4
Rendimiento	Comportamiento en el tiempo	4	4	4	4
	Optimización de recursos	3	5	4	4
Compatibilidad	Coexistencia	4	4	0	4
	Interoperabilidad	4	4	4	4
Usabilidad	Inteligibilidad	4	5	4	4
	Aprendizaje	5	4	4	5
	Operabilidad	4	5	4	4
	Responsabilidad	4	4	3	4
	Atractividad	4	4	3	5
	Accesibilidad	3	5	3	4
Fiabilidad	Madurez	3	4	3	3
	Disponibilidad	4	5	4	4
	Tolerancia a fallos	3	4	3	3
	Capacidad de recuperación	4	4	4	4
Seguridad	Confidencialidad	4	4	4	4
	Integridad	4	4	4	4
	Disponibilidad	4	4	4	4
	Autenticación	4	4	4	4
	Protección a errores de usuario	4	5	4	4
Mantenibilidad	Modularidad	4	4	3	4
	Reusabilidad	3	4	3	3
	Analizable	4	4	3	4
	Reemplazable	4	4	4	4
	Capacidad de ser probado	4	5	4	4
Portabilidad	Adaptabilidad	4	5	4	4
	Facilidad de instalación	4	5	3	4
	Intercambiabilidad	4	4	3	4
Calidad en uso					
Característica	**Sub característica**	**Spyder**	**PyCharm**	**Atom**	**vs Code**
Efectividad	Agilidad	3	4	3	4
Productividad	Optimización	4	5	4	4
Seguridad	Riesgo de daño económico	4	4	4	4
	Riesgo de salud	4	4	4	4
	Riesgo ambiental	4	5	4	4
Satisfacción	Cumplimiento del propósito	4	5	4	5
	Confianza	4	5	4	4
	Calidad de la experiencia	4	4	3	4
	Confort	4	5	4	4
Contexto de	Flexibilidad	3	5	4	4

uso	Cumplimiento de contextos de uso	4	5	4	4
TOTAL		**157**	**182**	**147**	**163**
Nota:	Las puntuaciones de desempeño están orientadas a la funcionabilidad del software en evaluación, la manipulación y creación de archivos de texto plano, así como poseer librerías y plugins descargables.				

IV. CONCLUSIONES

En este estudio, se programó e integró un asistente escolar basado en inteligencia artificial con las cámaras de vigilancia y los altavoces colocados en el aula de clase universitaria. Los resultados sugirieron que un asistente escolar para docentes universitarios puede complementarse con IA, lo que fomenta el uso de IA en la educación superior peruana. Además, se puede afirmar que la supervisión de los estudiantes en cuanto al comportamiento y uso correcto de los equipos de bioseguridad se realizó satisfactoriamente.

La interfaz gráfica de usuario estuvo correctamente operable utilizando las funcionalidades propuestas de acuerdo con las necesidades sanitarias de un aula universitaria. Por lo tanto, se concluye que la red YOLO v5 facilita un entrenamiento eficiente y una ejecución precisa con PyCharm. Estas herramientas facilitan el desarrollo robusto de una interfaz gráfica. Además de ello, al ser herramientas de código abierto se pueden utilizar en diferentes aplicaciones, siendo compatibles con múltiples bibliotecas.

AGRADECIMIENTOS

A la Universidad Privada del Norte (UPN) por el apoyo en parte del trabajo de investigación.

REFERENCIAS

[1] R. Darío and M. Padilla, "La llegada de la inteligencia artificial a la educación," *Rev. Investig. en Tecnol. la Inf.*, vol. 7, no. 14, pp. 260–270, 2019, doi: 10.36825/RITI.07.14.022.

[2] C. González, "Sistemas inteligentes en la Educación: Una revisión de las líneas de investigación," *Reli. - Rev. Electrónica Investig. y Evaluación Educ.*, vol. 10, no. 1, 2004, doi: 10.7203/RELIEVE.10.1.4329.

[3] L. Chamba *et al*., "Revisión Sistemática de Literatura: Estado de la Cuestión de la Enseñanza y Aprendizaje de la Inteligencia Artificial en Escuelas y Colegios," *osf.io*, 2022, Accessed: Sep. 02, 2022. [Online]. Available: https://osf.io/4t6zc/download

[4] E. Sanchez and M. Lama, "Monografía: Técnicas de la Inteligencia Artificial aplicadas a la educación," *Rev. Iberoam. Intel. Artif.*, vol. 11, no. 33, pp. 7–12, 2007, Accessed: Sep. 02, 2022. [Online]. Available: https://www.redalyc.org/pdf/925/92503302.pdf

[5] Y. Ocaña, L. Valenzuela, and L. Garro, "Inteligencia artificial y sus implicaciones en la educación superior," *Propósitos y Represent.*, vol. 7, no. 2, pp. 536–568, Jan. 2019, doi: 10.20511/PYR2019.V7N2.274.

[6] A. Salazar-Moran, F. Toala-Bozada, M. Ayón-Lucio, and Y. Solís-Barreto, "La deficiente atención académica de los estudiantes en las clases online en nivelación," *Polo del Conoc.*, vol. 7, no. 8, pp. 625–640, 2022, doi: 10.23857/pc.v7i8.

[7] I. Jara and J. M. Ochoa, "Usos y efectos de la inteligencia artificial en educación," *Banco Interam. Desarro. (Grupo BID)*, pp. 3–27, May 2020, doi: 10.18235/0002380.

[8] J. Ollivier, P. Martínez, and I. Domínguez, "Madurez tecnológica e innovación en empresas mexicanas," *Investig. Adm.*, vol. 50, no. 128, pp. 0–23, 2021, [Online]. Available: https://www.redalyc.org/articulo.oa?

[9] B. Schiller, T. Brogt, J. P. M. Schuler, G. Strobel, and S. Eicker, "Identifying Quality Factors for Self-Tracking Solutions: A Systematic Literature Review," *Hawaii Int. Conf. Syst. Sci.*, vol. 2020-Janua, pp. 3690–3699, 2020, doi: 10.24251/HICSS.2020.452.

[10] E. Guney and C. Bayilmis, "An Implementation of Traffic Signs and Road Objects Detection Using Faster R-CNN," *SAUCIS*, vol. 5, no. 2, 2021, doi: 10.35377/saucis.05.02.1073355.

[11] Y. Ma and S. Zhang, "Feature Selection Module for CNN Based Object Detector," *IEEE Access*, vol. 9, pp. 69456–69466, 2021, doi: 10.1109/ACCESS.2021.3073565.

[12] D. Wang and D. He, "Channel pruned YOLO V5s-based deep learning approach for rapid and accurate apple fruitlet detection before fruit thinning," *Biosyst. Eng.*, vol. 210, pp. 271–281, 2021, doi: 10.1016/j.biosystemseng.2021.08.015.

[13] K. Liu, "STBi-YOLO: A Real-Time Object Detection Method for Lung Nodule Recognition," *IEEE Access*, vol. 10, no. June, pp. 75385–75394, 2022, doi: 10.1109/ACCESS.2022.3192034.

[14] P. Gujjar, P. Kumar, and N. Chiplunkar, "Image classification and prediction using transfer learning in colab notebook," *Glob. Transitions Proc.*, vol. 2, no. 2, pp. 382–385, 2021, doi: 10.1016/j.gltp.2021.08.068.

[15] F. Nakhle and A. L. Harfouche, "Ready, Steady, Go AI: A practical tutorial on fundamentals of artificial intelligence and its applications in phenomics image analysis," *Patterns*, vol. 2, no. 9, p. 100323, 2021, doi: 10.1016/j.patter.2021.100323.

[16] S. Saabith, T. Vinothraj, and M. Fareez, "A review on Python libraries and Ides for Data Science," *Int. J. Res. Eng. Sci.*, vol. 9, no. 11, pp. 36–53, 2021, [Online]. Available: https://www.ijres.org/papers/Volume-9/Issue-11/Ser-2/G09113653.pdf

[17] J. De La Torre, J. L. Saorín, C. Carbonell, D. Del Castillo, and M. Contero, "Modelado 3D como herramienta educacional para el desarrollo de competencias de los nuevos grados de Bellas Artes," *Arte, Individuo y Soc.*, vol. 24, no. 2, pp. 179–193, 2012, doi: 10.5209/rev-ARIS.2012.v24.n2.39025.

[18] S. Salas and Y. Yang, "Artificial intelligence applications in Latin American higher education: a systematic review," *Int. J. Educ. Technol. High. Educ.*, vol. 19, no. 1, 2022, doi: 10.1186/s41239-022-00326-w.

[19] J. Roshan and V. Akhil, "SPlit: An Optimal Method for Data Splitting," *Technometrics*, vol. 64, no. 2, pp. 166–176, 2022, doi: 10.1080/00401706.2021.1921037.

[20] F. Vacas, "¿Internet sin pantallas? Altavoces conectados como dispositivo de acceso," *Rev. la Asoc. Española Investig. la Comun.*, vol. 6, no. 12, pp. 302–326, 2019, doi: 10.24137/raeic.6.12.14.

[21] C. Lecuona, N. Jerez, M. Padrón, and V. González, "Diseño e implementación de un prototipo de comunicador para dispositivos móviles orientado a personas con diversidad funcional," pp. 65–70, 2012.

[22] R. Evangelio, "Videovigilancia en las aulas universitarias y protección de la vida privada. Consideraciones sobre la STEDH de 28 de noviembre de 2017 (caso Antović)," *Derecho Priv. y Const.*, vol. 2017, no. 33, pp. 79–116, Nov. 2018, doi: 10.18042/cepc/dpc.33.03.

[23] V. Tintín, J. Caiza, and F. Caicedo, "Arquitectura de redes de información. Principios y conceptos," *Dominio las Ciencias*, vol. 4, no. 2, p. 103, 2018, doi: 10.23857/dc.v4i2.780.

[24] Z. Liu, L. Sun, and Q. Zhang, "Algoritmo de clasificación y reconocimiento de imágenes de alta similitud basado en una red neuronal convolucional," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–10, Apr. 2022, doi: 10.1155/2022/2836486.

[25] Y. Wang *et al*., "Construction and Application of Indoor Video Surveillance System Based on Human Activity Recognition," *MATEC Web Conf.*, vol. 232, pp. 1–6, 2018, doi: 10.1051/matecconf/201823204024.