UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

Modelo para el reconocimiento de rostros mediante una red neuronal para el pase de lista

Tesis:

Para obtener el título de

Ingeniero en Computación

Presenta:

Luis René Morales Velasco

Director de tesis:

M.T.C.A. Erik Germán Ramos Pérez

Co-director:

Dr. Christian Eduardo Millán Hernández

Huajuapan de León, Oaxaca

Julio de 2023

Agradecimientos

En el presente trabajo de tesis me gustaría agradecer primeramente a mis padres, Olivia Sagrario Velasco Sánchez y René Morales Luis, quienes me impulsaron a seguir adelante con este trabajo y siempre han estado para brindarme las herramientas necesarias para poder llegar a donde estoy ahora, sin su constante apoyo no sería yo la persona que soy ahora y espero poder retribuirles todo.

A mis hermanos, porque han estado conmigo a través de todo este recorrido y espero sea yo una fuente de inspiración para que sigan adelante con sus estudios y nunca se den por vencidos.

A mi director de tesis, el M.T.C.A. Erik Germán Ramos Pérez quién desde que comencé la carrera ha estado ahí para brindarme su apoyo y conocimientos, sin él esta tesis no hubiera sido posible y espero que se enorgullezca de este logro tanto como yo. Así mismo a mi co-director de tesis, el Dr. Christian Eduardo Millán Hernández, quién siempre estuvo dispuesto a ofrecer su opinión para mejorar este trabajo.

También quiero agradecer a mis profesores a lo largo de la carrera, quiénes me impulsaron a llegar a este punto y aportaron sus conocimientos para aumentar los míos, siempre dispuestos a dar una mano cuando alguna duda surgiera.

Finalmente, pero no menos importante, a mis amigos. No solo aquellos que la carrera me ha brindado, sino también los que he conocido a lo largo de estos años, en especial a Miguel Ángel Uribe Matus y Montserrat Ramírez Hernández, su apoyo incondicional y ganas de aprender me han permitido adquirir la confianza necesaria para poder crecer académicamente.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1:
Introducción

La tecnología ha avanzado en los últimos años, especialmente en lo que se refiere a la Inteligencia Artificial, que ha proveído las herramientas como son las Redes Neuronales Artificiales (RNA) y sus diversas aplicaciones en problemas de la vida cotidiana, sus aplicaciones varían desde toma de decisiones, reconocimiento de patrones, hasta sistemas de navegación autónoma.

Las RNA han tenido diversas aplicaciones, una de las cuales es el reconocimiento y clasificación de imágenes, buscando identificar objetos, animales o personas mediante cámaras de vigilancia, de tráfico a través de imágenes o video [11], también se ha tenido impacto en aŕeas cómo la medicina, economía, agronomía, etc.

En el ámbito académico, estas tecnologías se han aplicado para apoyar a los profesores a realizar procedimientos repetitivos y monótonos que consumen tiempo de sus tareas sustantivas. Una de las principales tareas que realizan al estar al frente de un grupo es el pase de lista, para asegurarse de que todos sus alumnos se encuentren presentes. Sin embargo, debido al tiempo que esta actividad suele abarcar dentro de su horario de clases algunas veces los profesores prefieren pasarla por alto.

Los pases de lista pueden verse como actividades anticuadas y sin beneficios visibles a la formación educativa de los alumnos, sin embargo, forman parte del rol fundamental en la formación académica y de valores de ellos. Principalmente, su enfoque es el de mantener la puntualidad y asistencia por parte de los alumnos a las clases. Sin un pase de lista y las consecuencias correspondientes a faltar o llegar tarde se crea la idea de que el asistir a clases no es algo necesario y se ve reflejado entonces en un incremento del abandono escolar [13].

Algunas soluciones tecnológicas para esta tarea son el uso de aplicaciones para dispositivos móviles como TeacherKit [6], la cuál simplifica las tareas rutinarias de el aula de clase, o la propuesta de un sistema basado en un dispositivo móvil [5] para tomar la asistencia y llevar un control de manera electrónica de las asistencias, puede también utilizarse para obtener estadísticas de los alumnos, pero al final implica que el profesor utilice tiempo de su clase para nombrar a cada uno de los estudiantes durante el pase de lista; su ventaja principal es la elaboración del reporte final con el porcentaje de asistencia de cada alumno y del grupo en general. Con respecto a las clases virtuales existen extensiones o APIs que facilitan el proceso en aplicaciones como Google meet, entre estas están *Asistencia en Meet* [1] o *Pasar Lista en Meet* [12], que permiten el pase de lista durante una videollamada.

Algunas aplicaciones han propuesto métodos más sofisticados para reducir el tiempo de pase de lista o automatizarlos. Por ejemplo [2] mediante un dispositivo móvil y la tecnología GPS se implementa

un pase de lista automatizado. En [9] propuso aplicaciones basadas en tecnología RFID para automatizar el pase de lista.

Por otro lado, gracias al avance en los últimos años en el área de las RNA también se han implementado alternativas mediante el uso de estas para el reconocimiento de rostros. Lin y Li en [10] hacen uso de cámaras colocadas en un salón de clases y una toma constante de vídeo para obtener las imágenes mediante las cuales se reconocen a los alumnos.

En este trabajo se hizo uso de un detector de objetos en tiempo real para mantener una alta precisión del reconocimiento de rostros, así como la creación de dos conjuntos de datos para su entrenamiento. Las imágenes fueron tomadas bajo diferentes iluminaciones para poder crear un conjunto más diverso así como de múltiples imágenes desde distintos ángulos.

1.1 Planteamiento del problema

El uso de sistemas de control de asistencia a clases tiene como fin asegurar que los alumnos se encuentren presentes en las aulas de clases para poder mejorar su rendimiento académico. Sin embargo, la manera tradicional de llevar a cabo este proceso consume tiempo que podría ser usado para impartir la clase en sí.

Una alternativa para este proceso es su automatización para que mediante una imagen pueda realizarse la verificación de asistencia. Esto tiene como beneficio una reducción en el tiempo que conlleva el proceso, mayor facilidad para llevar un registro de las estadísticas de asistencia y así una mejor administración del tiempo del profesor.

Un factor importante a considerar es el conjunto de datos que se utilizará para el entrenamiento, la calidad de las imágenes, la cantidad de alumnos a clasificar y el ambiente en el que se llevará a cabo la captura de estas.

1.2 Justificación

En la actualidad el proceso del pase de lista no se ha actualizado en lo que respecta a las clases presenciales y mantiene el mismo formato desde su creación. En este proceso el profesor nombra a cada alumno para verificar que se encuentren presentes en el salón de clases. Es por esto que la solución que se propone busca automatizar el pase de lista mediante el uso de un modelo de red neuronal lo que tiene como beneficio una reducción en el tiempo empleado en este proceso.

Con la finalidad de contextualizar el problema, se considera el caso particular de la Universidad Tecnológica de la Mixteca (UTM), la cual cuenta con aproximadamente 1500 alumnos, y una plantilla de

250 profesores, los cuales tienen en promedio de dos a tres grupos a su cargo. Las clases están diseñadas para durar de 50 a 60 minutos. Los pases de lista pueden tomar de 5 a 10 minutos, impidiendo ocupar este tiempo para utilizarlos en sus labores sustantivas, siendo un 8.33% de las horas laborales al año.

Una forma de sustituir el pase de lista tradicional es utilizar un proceso automatizado donde a partir de una fotografía digital se reconozcan los alumnos que estén presentes en el salón de clases. Sin embargo, para poder considerar este proceso es necesario asegurar la construcción de un modelo de reconocimiento facial que sea confiable y eficaz. De esta manera, el modelo puede ser llevado a otras áreas más allá de la asistencia en el salón de clases, como pueden ser en asistencia a reuniones o para trabajadores.

1.3 Hipótesis

Es posible mantener una precisión alta en el reconocimiento de rostros a partir de imágenes mediante una red neuronal para el pase de lista.

1.4 Objetivos

1.4.1 Objetivo general

Entrenar una red neuronal artificial para el reconocimiento de distintos rostros a partir de imágenes para el pase de lista.

1.4.2 Objetivos específicos

Para lograr el objetivo general es necesario llevar a cabo las siguientes actividades:

- Investigar sobre los modelos de RNA más utilizados y con mayor precisión.
- Crear un conjunto de datos de imágenes de diferentes rostros de alumnos para el entrenamiento de un modelo de RNA.
- Seleccionar el modelo de RNA que mayor rendimiento tenga a partir del estado del arte.
- Entrenar la RNA y analizar la evaluación de su desempeño en distintos escenarios.

1.5 Metas

1. Encontrar una RNA que sea adecuada para el entrenamiento.
2. Crear un conjunto de datos con imágenes de alumnos desde diversos ángulos, con distintas vestimentas y a distintas condiciones de iluminación.

3. Obtener un modelo basado en una RNA para la identificación de rostros de distintos alumnos.
4. Analizar los resultados obtenidos obtenidos para ver su desempeño.
5. Redactar el documento de tesis con la investigación realizada.

1.6 Limitaciones

Para asegurar la alta precisión del modelo, se necesitó que el conjunto de datos incluyera imágenes de los alumnos desde distintos ángulos, iluminaciones y vestimentas, por lo cual fue necesario que durante el periodo completo de un semestre en una clase se realizará la obtención de las fotos de un grupo de asistentes, para ello fue necesario contar con el consentimiento de los alumnos, por lo que solo una parte del grupo aceptó formar este conjunto de datos.

La calidad de las imágenes que se utilizaron, para realizar tanto el entrenamiento como los experimentos y analizar cómo influye en los resultados, fueron tomadas con las cámaras de teléfonos inteligentes.

El tamaño de la muestra utilizada se limitó a dos grupos y a un número limitado de personas que autorizaron el uso de su información para realizar esta proyecto y para esto se requirió que fueran cooperativos al momento de la adquisición de los datos.

Capítulo 2: Marco Teórico

En este capítulo se describen los conceptos claves del aprendizaje computacional (ML, por sus siglas en inglés) y las Redes Neuronales Artificiales (RNA), así como conceptos relacionados a estos. Además se presentan las herramientas y tecnologías de soporte para la realización del proyecto, así como el Estado del Arte sobre las RNA y trabajos relacionados hasta el momento.

2.1 Redes Convolucionales

De acuerdo a Yann Lecun en [8], las Redes Neuronales Convolucionales (Convolutional Neural Networks o CNNs) son arquitecturas multietapas entrenables compuestas de múltiples capas, en cuyas entradas y salidas de cada etapa se obtiene un conjunto de arreglos denominados *mapas de características*. Mientras que en la salida de la red se muestran las probabilidades (entre 0 y 1) de que un objeto en la imagen pertenezca a cierta clase.

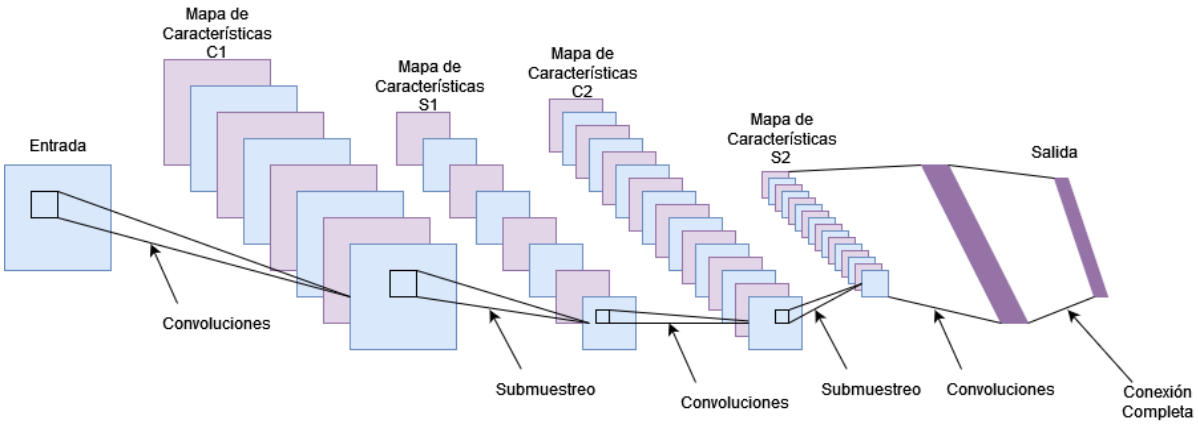


Figura 1. Una arquitectura ConvNet con dos etapas [8].

Según Lecun [8] cada etapa está compuesta por tres capas: una capa de banco de filtros, una capa de no-linearidad y una capa de agrupación de características (Figura 1), aunque según otros autores, las tres capas son: Capa de convolución, capa de agrupación o submuestreo y la capa totalmente conectada (FC por sus siglas en inglés).

De acuerdo a IBM [4], la capa convolucional es la primera en una CNN y la Totalmente Conectada es la última, aunque se pueden poseer varias capas convolucionales o de agrupación entre ellas, siendo esto lo que define la complejidad que tendrá la red neuronal. En general las primeras capas se concentran en características simples, como colores y bordes; para que a medida que la imagen es procesada a través de las demás capas, la red empiece a reconocer elementos o abstraer más información de ellos hasta que finalmente reconoce el objeto deseado. A continuación se explican algunas de estas capas [7].

2.1.1 Capa convolucional

La capa convolucional es el bloque central de una CNN y está compuesta por un grupo de filtros o *kernels.* Recibe una imagen de entrada y así genera un mapa de características en la salida. Si por ejemplo la entrada es una imagen a color, entonces se tienen 3 dimensiones en la entrada correspondientes a cada una de los canales RGB (sigla del inglés ***R**ed, **G**reen, **B**lue*; en español, 'rojo, verde, azul') en una imagen. El kernel se traslada a través de los campos de la imagen creando así pesos en un mapa de características para así irlos ajustando de mejor manera en cada iteración. A este proceso se le llama convolución.

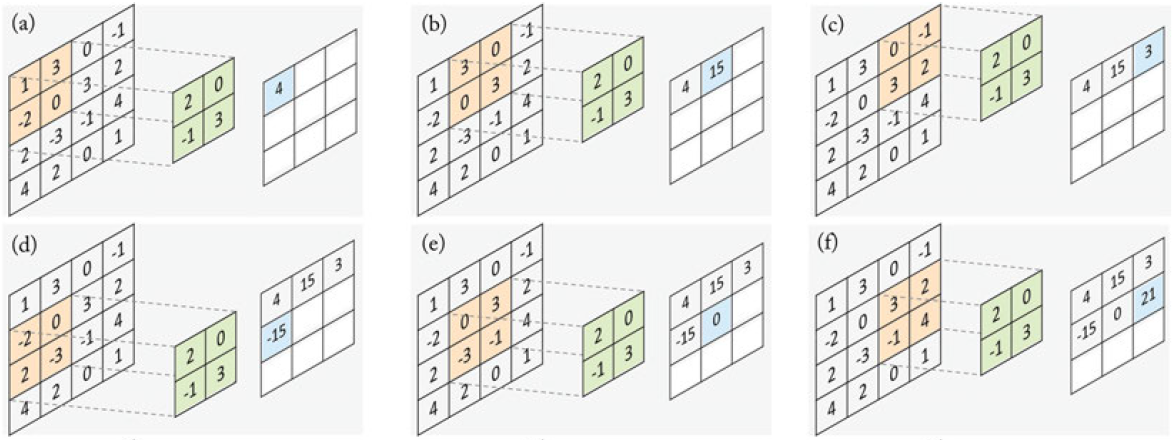


Figura 2. Imagen de una operación de convolución [7]. (a-f) Se observan los cálculos realizados a medida que el *kernel* pasa por diversas regiones de la imagen.

Como se observa en la Figura 2 el filtro es aplicado a un área de la imagen, y se calcula el producto punto entre los píxeles de entrada y el filtro, cuyo resultado se guarda en un arreglo de salida (mapa de características). Posteriormente, el filtro recorre la imagen de izquierda a derecha y de arriba hacia abajo, repitiendo el proceso hasta que el kernel ha pasado por toda la imagen. Y una vez terminado el proceso la red aplica una transformación de Unidad Lineal Rectificada (ReLU por sus siglas en inglés) al mapa de características para introducir no-linearidad al modelo.

2.1.2 Capa de agrupación

Las capas de agrupación se encargan de la reducción dimensional de los mapas de características obtenidos en las capas convolucionales, acortando el tamaño del mapa (dependiendo del tamaño de la máscara elegida) antes de la entrada de la siguiente capa. Al igual que en la capa convolucional, la operación de agrupación recorre toda la entrada pero con la diferencia de que esta

máscara no tiene pesos predeterminados. En su lugar, el *kernel* aplica una función de agregación a los valores dentro del mapa de características, llenando así el arreglo de salida y obteniendo un nuevo mapa (ver en la Figura 3).

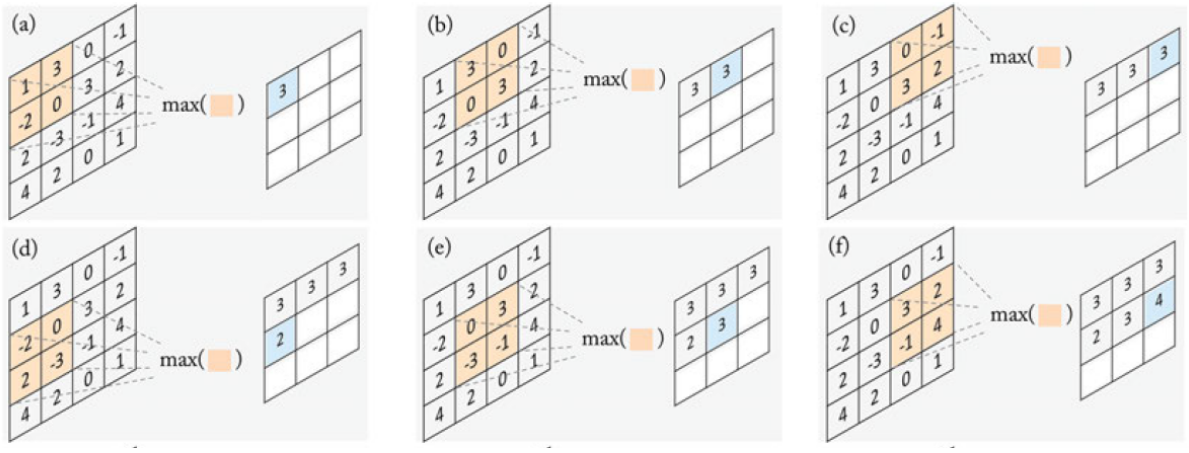


Figura 3. Imagen de una operación de agrupación [7]. (a-f) Se observan los cálculos realizados a medida que la máscara de agrupación pasa sobre el mapa de características.

Hay dos tipos principales de agrupación:

- Agrupación Máxima *(Max pooling)*: A medida que el filtro se mueve por el mapa se selecciona el píxel de mayor valor para colocarlo en el arreglo de salida.
- Agrupación Promedio *(Average pooling)*: A medida que el filtro se mueve, se calcula el promedio de los píxeles y es lo que se manda al arreglo de salida.

A pesar de la pérdida de información en esta capa, tiene muchos beneficios para la CNN, reduciendo la complejidad, mejorando la eficiencia, limitando el riesgo de desborde, que lo hace invariante a escalamientos, transformaciones o deformaciones en una imagen.

2.1.3 Capa totalmente conectada (FC)

Las capas completamente conectadas son en esencia capas de convolución con filtros de tamaño 1x1, donde en cada unidad en una capa completamente conectada está densamente conectada a todas las unidades de la capa anterior, de ahí su nombre. En general en una arquitectura típica esta capa suele ser la última para poder producir lo que es el mapa de características final, que será empleado para hacer la clasificación. Para esto la capa completamente conectada hace uso de una función de salida softmax de activación para clasificar adecuadamente las entradas, produciendo una probabilidad de 0 a 1 para cada clase, eligiendo siempre a la clase con la probabilidad más alta al momento de clasificar.

2.1.4 Parametrización de las capas.

La principal diferencia entre cada red neuronal convolucional se encuentra en la forma en que están apiladas las capas que las componen [7], pero también por la forma en que se parametrizan.

Las capas de convolución y agrupación tienen hiperparámetros, es decir, parámetros cuyo valor primero debe ser definido por el usuario antes de comenzar el entrenamiento y son estos los que definirán el tamaño de los mapas de características de salida en estas capas. Estos hiperparámetros pueden interpretarse como las posibles formas en que puede diseñarse la arquitectura de una red neuronal, y puede enfocarse dependiendo de la aplicación que vaya a dársele.

2.2 Aumento de datos

Cuando el conjunto de datos de entrenamiento no es lo suficientemente grande se pueden aplicar un conjunto de técnicas que amplían el tamaño y la calidad del conjunto de datos, con el fin de construir modelos de aprendizaje profundo más efectivos. A estas técnicas se les denomina aumento de datos e implican la introducción de pequeñas alteraciones o modificaciones en las imágenes de entrada, lo que virtualmente incrementa el tamaño del conjunto de datos de imágenes. Algunos de los algoritmos que se engloban en este concepto son: transformaciones geométricas, aumentos de espacio de color, filtros, borrado aleatorio de píxeles, aumento de espacio de características, entrenamiento adversario, redes adversarias generativas, transferencia de estilo neuronal y metaaprendizaje [14]. A continuación se mencionan algunas operaciones para el aumento de datos.

2.2.1 Inversión

La inversión implica voltear la imagen respecto a su eje X o eje Y dando lugar a una imagen "espejo" (Figura 4). En general es más común el uso de la inversión respecto al eje X en las aplicaciones que respecto al eje Y [14].

(a) (b) (c)

Figura 4. Inversión de una imagen. (a) Imagen original (b) Imagen espejo horizontal (c) Imagen espejo vertical.

2.2.2 Espaciado de color

Los datos de imágenes digitales generalmente se almacenan como una matriz dimensión (alto × ancho × canales), si es imagen a escala de grises, tendrá un solo canal, y si es imagen a color, entonces tendrá 3 canales. Un ejemplo puede ser aislar un solo canal de color, por ejemplo el rojo R, o el verde G o el azul B, simplemente apagando 2 de los 3 canales (Figura 5) . Además, también se pueden manipular los canales con el fin de aumentar o disminuir el brillo o contraste de la imagen [14].

(a) (b) (c) (d)

Figura 5. Espaciado de color. (a) Imagen original (b) Canal rojo (c) Canal verde (d) Canal azul.

También, para tener un conjunto de datos mayor, se puede hacer uso de los histogramas que describen a la imagen. Al cambiar los valores de intensidad en estos histogramas, se obtienen imágenes con cambios en su iluminación.

2.2.3 Recorte

Esta técnica está enfocada en obtener un corte central de una imagen cambiando sus dimensiones (diferentes ancho y altura), logrando obtener una sección específica (Figura 6). Esto implica reducir las medidas de la imagen original, por ejemplo, en la figura 6(a), las dimensiones de la imagen original es de 4032 x 3024 y la imagen resultante es de 1934 x 1208. Sin embargo, esto puede implicar una pérdida de los cuadros delimitadores en base al umbral escogido [14].

(a) (b)

Figura 6. Recorte de una imagen. (a) Imagen original (b) Imagen recortada.

2.2.4 Rotación

Consiste en girar la imagen hacia la derecha o hacia la izquierda en un rango de 0° a 360° (Figura 7). Un efecto importante es el posible cambio de dimensión de la imagen, por ejemplo 45°, al hacer rotar existe la posibilidad de que se pierdan las posiciones de los cuadros delimitadores [14].

(a) (b)

Figura 7. Rotación de una imagen. (a) Imagen original (b) Imagen rotada.

2.2.5 Traslación

Consiste en el desplazamiento de las imágenes en los ejes X y Y, se aplica principalmente cuando la imagen se está centrada para no perder información valiosa, es importante mencionar que al aplicar unas traslación, la imagen puede cambiar de tamaño, o en su defecto poder conservar las medidas originales, los espacios desplazados se pueden rellenar con un valor constante, normalmente 0 o 255 (Figura 8) [14].

(a) (b)

Figura 8. Traslación de una imagen. (a) Imagen original (b) Imagen trasladada.

2.3 Estado de Arte

Las CNN están compuestas por distintas estructuras, las cuales han evolucionado con el paso del tiempo en [7] se mencionan algunas de las más prominentes en el desafío ImageNet Large Scale Visual Recognition Challenge (ILSVRC) hasta el 2015.

2.3.1 LeNet-5 (1998)

LeNet-5 es una de las primeras redes convolucionales y fue creada por LeCun en 1998, esta red permitía reconocer números escritos a mano en cheques digitalizados (Cuadro 1). Está constituida por 7 capas: 3 capas convolucionales intercaladas, 2 capas de agrupación y 2 capas totalmente conectadas.

Entradas	Capas	Parámetros
227 × 227	7	60 mil

Cuadro 1. Características de LeNet-5.

2.3.2 AlexNet (2012)

AlexNet es una red a gran escala con mayor profundidad así como con un conjunto de capas convolucionales apiladas (Cuadro 2). Inspirada en LeNet consiste en 5 capas convolucionales, 3 capas de agrupación y finalmente 3 capas totalmente conectadas.

Entradas	Capas	Parámetros
227 × 227	11	60 millones

Cuadro 2. Características de AlexNet.

Todas estas características le permitieron ganar el ILSVRC 2012, AlexNet superó significativamente a todos los competidores anteriores al reducir el error de exactitud del 26% al 15.3%.

2.3.3 ZFNet (2013)

Para el ILSVRC 2013 el ganador fue una CNN denominada ZFNet la cual logró una tasa de error del 14.8%, esto se logró mediante el ajuste de hiperparámetros, reduciendo el tamaño y pasos de los filtros, siendo esto una mejora a partir de AlexNet (Cuadro 3).

Entradas	Capas
227 × 227	7

Cuadro 3. Características de ZFNet.

2.3.4 GoogleNet/Inception (2014)

GoogleNet, creada por Google, se inspiró en la CNN LeNet pero implementó un elemento denominado módulo de inicio (*inception*) el cual consistió en realizar las operaciones de las capas de manera paralela, en lugar de secuencial, generando un mapa de características multidimensional, pasando el mapa de características resultantes por varias convoluciones pequeñas para reducir el número de dimensiones (Cuadro 4).

Para el ILSVRC 2014 logró una tasa de error del 6.67% muy cercana del rendimiento a nivel humano y con esto se convirtió en la ganadora de la competencia.

Entradas	Capas	Parámetros
299 × 299	22	4 millones

Cuadro 4. Características de GoogleNet.

2.3.5 VGGNet (2014)

VGGNet es una red cuya estructura es bastante simétrica mediante el uso de 16 capas convolucionales apiladas de 3x3, intercaladas con capas de agrupación y finalmente capas totalmente

conectadas. Fue la segunda mejor CNN en el ILSVRC 2014, y fue desarrollada por Simonyan y Zisserman (Cuadro 5).

Entradas	Capas	Parámetros
227 × 227	16	138 millones

Cuadro 5. Características de VGGNet.

2.3.6 ResNet (2015)

La Red Neural Residual (ResNet) desarrollada por Kaiming He, presentó una arquitectura con saltos de dos a tres capas y realizan normalización de lotes y no linealidad entre ellas, lo cual facilita el entrenamiento de redes muy profundas (Cuadro 6). Estas conexiones de salto generan mapas de características mucho más simples de procesar y permiten un aprendizaje más estable. Fue la ganadora del ILSVRC 2015 logrando una tasa de error del 3,57 %.

Entradas	Capas	Parámetros
224 × 224	18	11 millones

Cuadro 6. Características de ResNet.

Ahora bien, una aplicación de las CNN se encuentra en detectores de objetos en tiempo real mediante los cuales se pueden clasificar y reconocer objetos o personas, a continuación se mencionan algunos de los mejores en años recientes.

2.3.7 YOLOR

Yolor [15] es una “propuesta de red unificada que busca integrar tanto el conocimiento implícito como el conocimiento explícito, y permitir que el modelo aprendido contenga una representación general de los problemas, y esta representación general a sub-representaciones adecuadas para varias tareas”.

Usando la arquitectura de Yolov4-CSP como base, se buscó integrar la idea del conocimiento implícito, de tal manera que se mezcle junto al conocimiento explícito mediante el uso de un transformador, integrando *kernels* de alineación de espacios, refinamiento de predicciones y aprendizaje de multi-tareas.

Con estas consideraciones se hace uso de vectores y matrices para la integración del aprendizaje explícito e implícito al modelo obteniendo una precisión comparable con la de Scaled-YOLOv4-P7, con una velocidad de inferencia 86% más rápida.

2.3.8 DINO: DETR

DINO: DETR es una propuesta de detector de objetos de extremo a extremo en tiempo real, mejora el rendimiento y eficacia comparado con DETR anteriores mediante el uso de transformadores para el entrenamiento de eliminación de ruido, así como un método de selección de consulta mixta para la inicialización de una caja delimitadoraa y un esquema doble de anticipación para la predicción de cajas delimitadora.

Para poder acelerar el tiempo de convergencia, DETR introduce más etiquetas de ruido en el transformador decodificador y entrenarlo para que reconstruya las etiquetas reales.

DINO alcanzó una precisión promedio de 49.4% en 12 épocas y de 51.34% en 24 en el dataset de COCO. Los resultados de las comparativas con otros detectores en tiempo real se ven en la Figura 9.

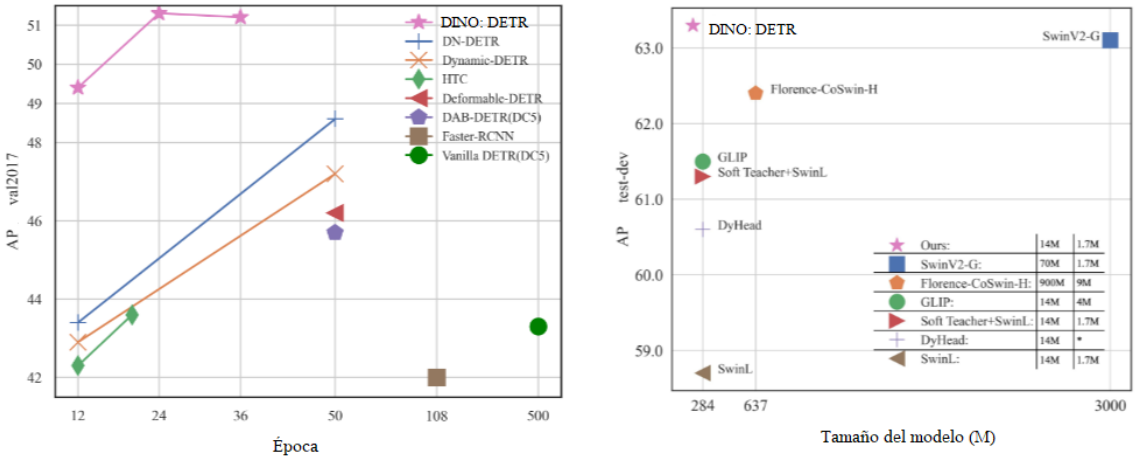


Figura 9. Comparación de DINO contra otros detectores [17].

2.3.7 YOLOv7

YOLOv7 es una propuesta novedosa basada YOLO a partir de una nueva arquitectura. Logró superar a todos los detectores de objetos conocidos tanto en velocidad como en precisión (Figura 10).

Esto se debe principalmente a su arquitectura denominada E-ELAN, esta versión no utiliza redes troncales preentrenadas de ImageNet, los modelos se entrenan usando el conjunto de datos COCO.

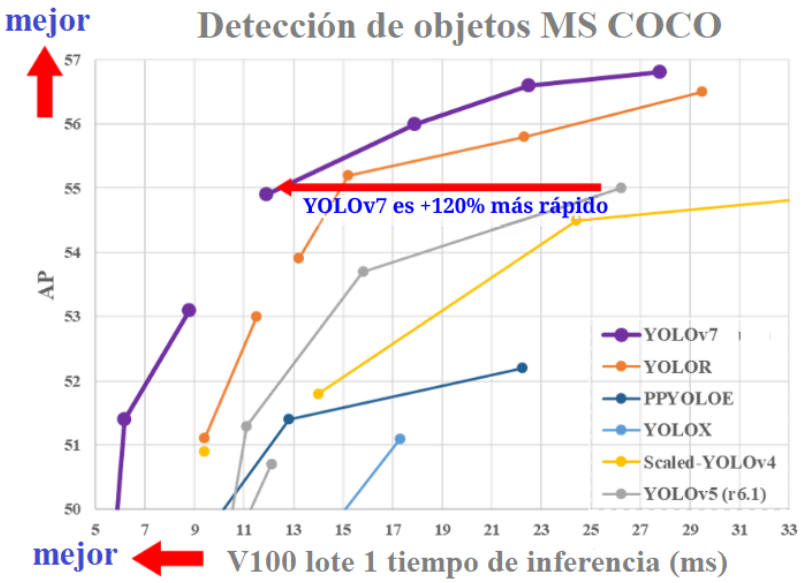


Figura 10. Comparación de YOLOv7 contra otros detectores [16].

A continuación se mencionan algunos trabajos previos realizados para tomar el pase de lista a partir de imágenes o video mediante el uso de distintos tipos de redes neuronales.

2.3.10 Student Attendance System using Face Recognition

El desarrollo del sistema propuesto en [3] tiene como objetivo lograr la digitalización del sistema tradicional de tomar asistencia.

Mediante clasificadores Haar, KNN, CNN, SVM, redes generativas antagónicas y filtros de Gabor se busca lograr el reconocimiento de rostros. Además también se crean informes de asistencia y se almacenan en formato Excel. Este sistema se puso a prueba bajo varias condiciones como la iluminación, los movimientos de la cabeza, la variación de la distancia entre el estudiante y las cámaras.

Para la implementación de éste se busca que mediante dos o más cámaras (dependiendo del tamaño del aula) se tomen fotografías constantemente a lo largo de la clase y de esta manera realizar el pase de lista.

De las pruebas realizadas, los resultados más precisos se obtuvieron mediante el uso del algoritmo KNN con un 99.27% de precisión, aunque las CNN tuvieron menos costo computacional tuvieron un porcentaje de precisión del 98.54% y las SVM fueron las de peores resultados con tan solo un 80.15% de precisión.

2.3.11 Design and Implementation of Classroom Attendance System Based on Video Face Recognition

Este artículo [10] presenta un sistema de asistencia en el aula basado en tecnología de reconocimiento facial por video. El sistema propuesto, primeramente hace uso de una cámara instalada en el aula para obtener video durante las clases. Posteriormente para el procesamiento, el video recopilado se divide en un cuadro de imágenes estáticas, de las cuales se seleccionan aquellas imágenes con rostros claros y mejor iluminación para poder hacer el reconocimiento facial y, finalmente, los resultados del reconocimiento se fusionan en caso de que haya personas repetidas para obtener la lista final.

Un factor a considerar es la influencia que puede tener la ubicación de la cámara en los resultados del reconocimiento, para solucionar esto se puede hacer uso del sistema de control de la plataforma donde se maneje el pase de lista, así se puede controlar la rotación y el enfoque de la cámara, mejorando aún más la precisión del reconocimiento.

Este sistema está dividido en dos componentes, primero la cámara para la adquisición de fotos está instalada en el aula y su función principal es recopilar videos de los estudiantes en tiempo real y transmitir los videos recopilados al servidor a través de la red para su almacenamiento y procesamiento. El segundo componente es el servidor cuya función principal es segmentar el video para obtener la imágenes del cuadro, luego segmentar cada imagen para reconocer la figura humana, así filtrar y mejorar los resultados del reconocimiento.

Capítulo 3: Desarrollo del Proyecto

En este capítulo se describe la experimentación y los resultados obtenidos. En la sección 3.1, las especificaciones del hardware y software, en la sección 3.2, se describe la creación del conjunto de datos. A partir de la sección 3.3 se presentan y describen cada uno de los módulos desarrollados para este proyecto de tesis. Finalmente, en la sección 3.4, se describe la forma en que se validan los resultados.

3.1 Especificaciones de Hardware y Software

En el Estado del Arte los entrenamientos se han realizado haciendo uso de GPUs para mayor rapidez en el proceso de entrenamiento. Por lo tanto, en el presente trabajo se utilizó una computadora con GPUs.

Para la construcción del modelo de identificación de rostros para pase de lista, se utilizó el detector de objetos Yolov7, adaptando los pesos originales del modelo base mediante un entrenamiento adicional.

Las especificaciones del uso de Yolov7 requiere la instalación particular de las siguientes dependencias y versiones:

- Python 3.10+
- matplotlib>=3.2.2
- numpy>=1.18.5
- opencv-python>=4.1.1
- Pillow>=7.1.2
- PyYAML>=5.3.1
- requests>=2.23.0
- scipy>=1.4.1
- tqdm>=4.41.0
- protobuf<4.21.3
- tensorboard>=2.4.1
- pandas>=1.1.4
- seaborn>=0.11.0

Para el uso de GPU para entrenamiento es necesario cubrir los siguientes requisitos de dependencias:

- torch==1.11.0+cu113
- torchvision==0.12.0+cu113

3.2 Creación del conjunto de datos

El conjunto de datos se compone por una variedad de imágenes de los alumnos que conformarán las clases. Estas fotos fueron tomadas a lo largo de cierto periodo de tiempo y a partir de distintos ángulos en la zona frontal del salón. Al final el conjunto de datos está compuesto por una serie de imágenes donde los participantes presentan distintas poses, vestimentas y peinados. Incluso se permitió la posibilidad de distintas condiciones de iluminación (Figura 11).

Figura 11. Imágenes tomadas para el conjunto desde distintos ángulos y con distinta ropa.

3.2.1. Adquisición de imágenes

Los pasos necesarios para la adquisición de las imágenes son:

1. Selección de los grupos que van a participar.
2. Pedir la autorización de los alumnos que participaron.
3. Toma de imágenes: En la hora de la clase, la fotografía se toma con un dispositivo móvil desde la parte frontal del aula.

3.2.2. Etiquetado de imágenes

Con las imágenes capturadas se puede llevar a cabo el etiquetado de las imágenes siguiendo los pasos:

1. Seleccionar el software de etiquetado.
2. Seleccionar el cuadro delimitador de cada alumno por cada imagen.

3.3 Entrenamiento de la red neuronal (CNN)

El módulo más importante del proceso consiste en el entrenamiento de la red mediante los conjuntos de datos creados. Este proceso hace uso de la arquitectura de Yolov7 para su realización mediante la adaptación de los pesos con los que ha sido previamente entrenada la red, esto con el fin de reducir el proceso de cómputo que implicaría entrenar la red desde cero. Para esto fue necesario seguir el esquema mostrado en la Figura 12. El cual consiste en primero importar el conjunto de imágenes al detector dividido en dos carpetas, una para el entrenamiento y otra para la verificación. Posteriormente, se ajustan los parámetros del clasificador en lo que respecta a la cantidad de clases que se requieren, así como la selección de los pesos iniciales para el entrenamiento de la red.

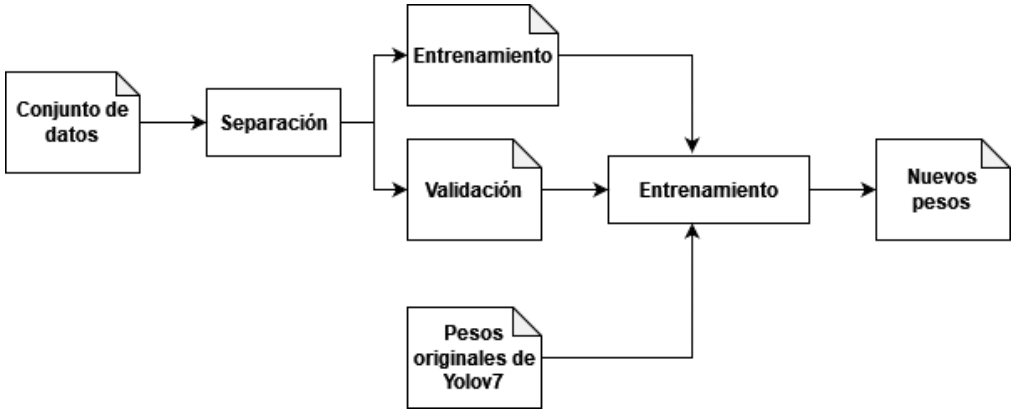


Figura 12. Entrenamiento de Yolov7

Una vez realizados los pasos descritos anteriormente se ejecuta la función de entrenamiento de yolov7, mediante un programa en python se inicia el entrenamiento con el comando en el Anexo 3.
Los parámetros de este comando son[1] :

- workers: Número máximo de trabajadores de dataloader.
- device: El número de GPU (ID) que se usará para el entrenamiento. Puede ser 0,1..n o CPU.
- batch-size: Tamaño total del lote para todas las GPU.
- epochs: Define el número de épocas para el que se entrenará el modelo.
- img: De forma predeterminada, las imágenes se redimensionan a una resolución de 640 × 640 antes de enviarse a la red. Aún así, se proporciona el tamaño de la imagen aquí.
- data: Este parámetro acepta la ruta al archivo YAML del conjunto de datos.
- hyp: Todos los modelos de la familia YOLOv7 tienen un conjunto diferente de parámetros e hiperparámetros. Estos incluyen la tasa de aprendizaje, las técnicas de aumento y también la intensidad de los aumentos, entre muchos otros hiperparámetros. Todos estos están definidos en

[1] https://github.com/WongKinYiu/yolov7 (último acceso 03/07/2023)

sus archivos de hiperparámetros (archivos YAML) en el directorio yolov7/data. Aquí, se especifica la ruta al archivo de hiperparámetros del modelo apropiado.

- cfg: Esta es la ruta al archivo de configuración del modelo que se necesita para cargar la arquitectura del modelo.
- name: Nombre del subdirectorio donde se guardarán los resultados del entrenamiento.
- weights: Esta bandera acepta la ruta al modelo preentrenado.

Una vez definidos los parámetros se procede al entrenamiento de la red. Este proceso consiste en que la imagen sea procesada por la red convolucional. En el caso de Yolov7, su arquitectura es la *Extended efficient layer aggregation networks* (E-ELAN) [16] la cual utiliza la expansión, mezcla y fusión de la cardinalidad para mejorar continuamente la capacidad de aprendizaje de la red sin destruir la ruta de gradiente original, como se muestra en la Figura 13.

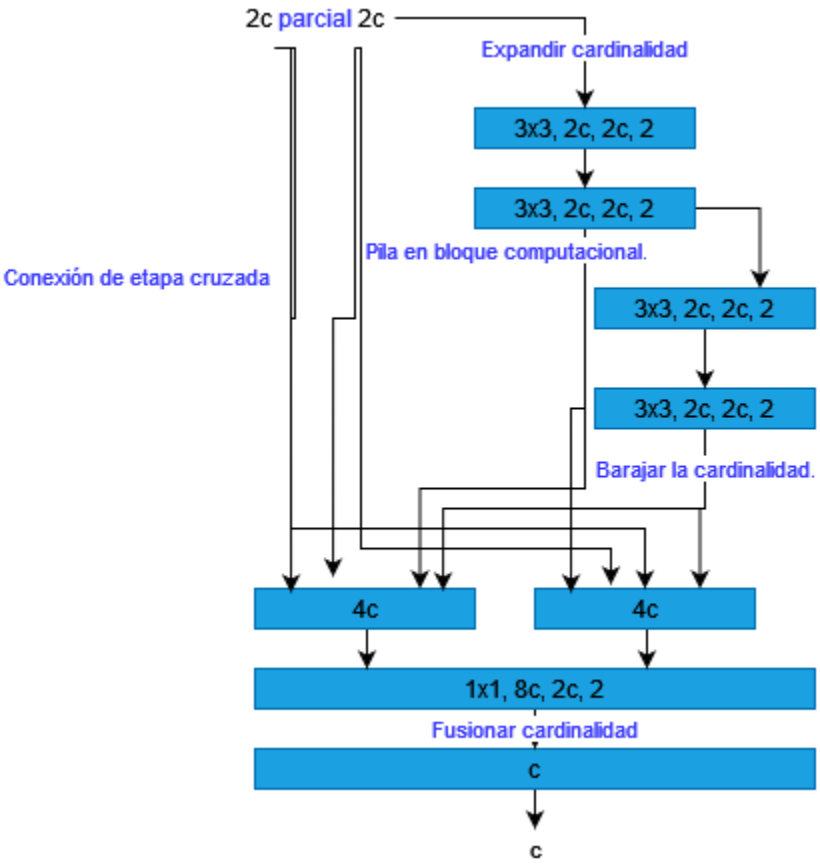


Figura 13. Arquitectura E-ELAN (Traducida y modificada de [16]).

3.4 Validación de los resultados del entrenamiento.

En esta parte se analizan los resultados que se obtienen del entrenamiento utilizando las siguientes métricas:

- Precisión (Precision),
- Pérdida (Loss),
- Exhaustividad (Recall).

Capítulo 4: Experimentación y resultados

En este capítulo se muestran los experimentos o pruebas realizadas con las imágenes de 2 grupos, los resultados obtenidos y cómo se mejoraron los resultados de los experimentos modificando el conjunto de datos utilizado.

4.1 Hardware y software utilizado

El entrenamiento y la ejecución de los experimentos se llevaron a cabo en el equipo de cómputo Dell G3 3500 cuyas características se describen en el cuadro 7.

Modelo	LENOVO LEGION Y740-171RHg
Memoria RAM	32 GB
Procesador	Intel® Core™ i7-9750H CPU @ 2.60GHz × 12
Sistema Operativo	Ubuntu 20.04 LTS de 64 bits
Tarjeta Gráfica	NVIDIA GeForce RTX 2070

Cuadro 7. Especificaciones del Hardware para el entrenamiento.

Las especificaciones de las versiones de las librerías utilizadas se pueden ver en el anexo 5.

4.2 Adquisición de las imágenes

Se eligieron dos grupos de personas para hacer los experimentos, el grupo Alumnos-UTM-902 compuesto de 6 alumnos y el Alumnos-UTM-602 de 22. Para el primer grupo se hizo una primera captura de fotografías para la creación del conjunto de datos en un período de tres meses, esto con el fin de realizar un primer entrenamiento y evaluación. Posteriormente, se adquirieron fotos por dos meses más para mejorar la calidad del conjunto de datos. Con el segundo conjunto de datos se adquirieron las fotos a lo largo de 3 meses para el entrenamiento.

Una vez capturadas las imágenes, fue necesario hacer una depuración de aquellos que no tuviesen la calidad adecuada para el entrenamiento, esto podría deberse a factores como: la calidad de la captura, rostros borrosos causados por el movimiento, poca iluminación, identificación de rostros incompletos e incluso situaciones donde alguien o algo se atravesó en la foto. Después de la selección de imágenes, se obtuvo un total de 416 imágenes para el primer grupo (Alumnos-UTM-902), las cuales fueron divididas en 404 imágenes para entrenamiento y 12 imágenes para validación.

Para el segundo grupo (Alumnos-UTM-602) se manejaron primeramente 336 imágenes para entrenamiento y 12 de validación, posteriormente se aplicó aumento de datos con diferentes posiciones, rotaciones, iluminaciones, de manera que se alcanzaron a obtener 747 imágenes para entrenamiento y 12 para validación, este proceso puede verse en la Figura 14.

(a) (b)

Figura 14. Aumento de datos. (a) Imagen original (b) Imagen con brillo modificado.

4.3 Etiquetado de fotos

Una vez adquiridas suficientes imágenes en el conjunto de datos, se realiza el proceso de etiquetado. El etiquetado consiste en identificar datos sin procesar (imágenes, archivos de texto, videos, etc.) y agregar una o más etiquetas significativas e informativas para proporcionar contexto para que un modelo de aprendizaje computacional pueda aprender de ellos.

Este proceso se realizó con ayuda de una herramienta de python de nombre labelImg, (las instrucciones de instalación se pueden ver en el anexo 1). Una vez instalado, se ejecuta la herramienta (ver anexo 2).

Aparece la interfaz mostrada en la Figura 15, la cual nos permite seleccionar el directorio dónde se encuentran las imágenes a etiquetar y dónde guardar las etiquetas.

Figura 15. Interfaz de labelImg

Una vez que se carga una imagen, se puede seleccionar el área de interés (*Region of Interest* o ROI), para crear un cuadro delimitador (*bounding box* en inglés). En este caso, el área de interés es el rostro de cada persona y que para cada participante significa una clase. En la Figura 16 se observa el ROI seleccionado y la etiqueta correspondiente al nombre seleccionado. Adicionalmente, se configura la etiqueta para que se ajuste al formato de etiquetación de Yolo. Este proceso se realiza manualmente con todas las imágenes del conjunto, por lo que resultó en una labor tardada considerando el tamaño del conjunto y el cuidado en la precisión de la selección del área de interés.

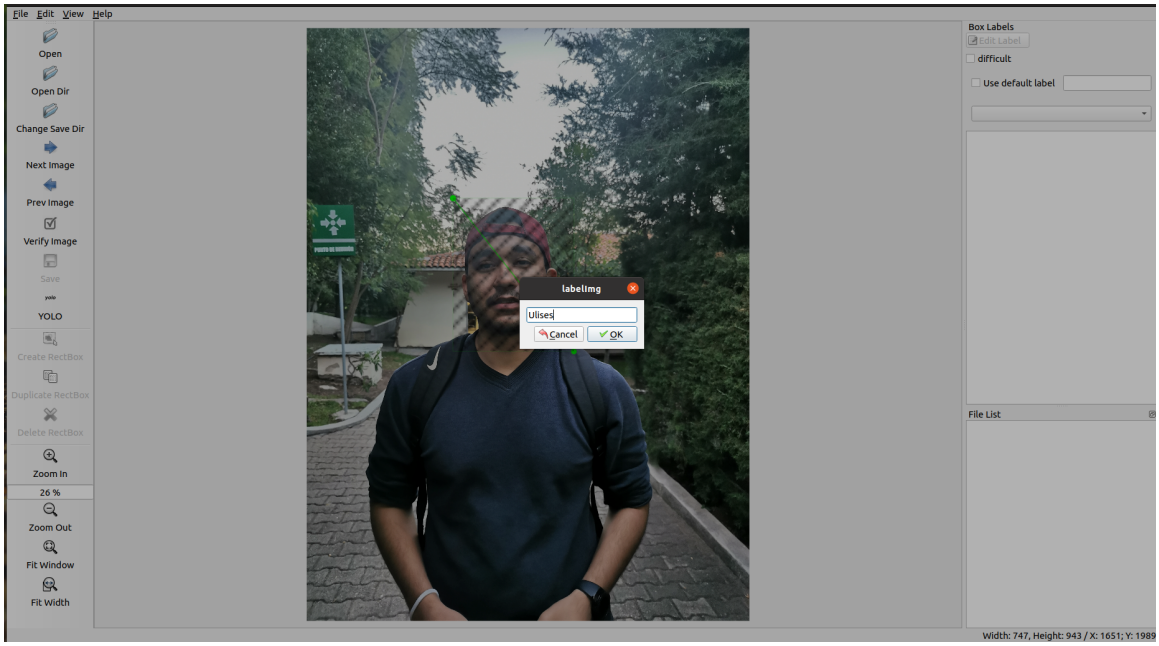


Figura 16 (a)

Figura 16 (b)

Figura 16. Etiquetamiento de imagen mediante la herramienta labelImg, en (a) se selecciona la clase y en (b) se obtiene en cuando encapsulador.

En esta fase se etiquetaron los conjuntos de datos como se muestra en el cuadro 8.

Conjunto de datos	Clases
Alumnos-UTM-902	Samuel, Matus, Ulises, Austin, Abner, Baruc
Alumnos-UTM-602	A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V

Cuadro 8. Grupos sobre los que se hicieron los experimentos.

4.4 Entrenamiento

Para el caso del conjunto de datos Alumnos-UTM-902 se establecieron los parámetros mostrados en el cuadro 9 para los experimentos 1,2,3 y 4. En el caso del conjunto de datos Alumnos-UTM-602 se tienen los parámetros en el cuadro 10.

Experimento N	Épocas	Tamaño (píxeles)	Número de imágenes entrenamiento	Número de imágenes validación
Experimento 1	100	640 x 640	256	8
Experimento 2	100	640 x 640	404	12
Experimento 3	100	640 x 640	404	12
Experimento 4	100	1824 x 1368	404	12

Cuadro 9. Argumentos para los experimentos del grupo Alumnos-UTM-902.

Experimento N	Épocas	Tamaño (píxeles)	Número de imágenes entrenamiento	Número de imágenes validación
Experimento 1	100	640 x 640	236	12
Experimento 2	100	1024 x 769	747	12

Cuadro 10. Argumentos para los experimentos del grupo Alumnos-UTM-602.

El entrenamiento se realizó con YOLO V7, para realizar la detección de objetos YOLO separa primero la imagen en N cuadrículas, cada una de estas de dimensiones SxS. Se usa cada una de estas regiones para detectar y localizar cualquier objeto que puedan contener, por cada cuadrícula así como las coordenadas del cuadro delimitador. Una vez realizado este proceso se pronostican los objetos con una etiqueta y una puntuación de probabilidad entre 0 y 1 para la presencia del objeto pronosticado.

Sin embargo, debido a que este proceso se hace en diversas cuadrículas, algunas de las predicciones se sobreponen. Para manejar esta redundancia y reducir los objetos pronosticados a aquellos realmente de interés, YOLO utiliza la supresión no-máxima para suprimir todos los cuadros delimitadores con puntajes de probabilidad más bajos y solo conservar aquellos sobre cierto umbral. Para lograr esto, YOLO realiza los siguientes pasos de forma iterativa: Primero compara los puntajes de probabilidad asociados con cada región y toma el puntaje más alto. A continuación, elimina los cuadros delimitadores con la intersección sobre unión (ISU) más grande con el cuadro delimitador de alta probabilidad elegido. Así hasta que solo queden los cuadros delimitadores finales deseados.

Para poder realizar los experimentos fue necesario usar el comando en el anexo 4, para poder ejecutar el detector con los siguientes parámetros:

- weights: Pesos para la detección.
- conf: Valor del umbral de confianza del objeto.
- img-size: Tamaño de inferencia de la imagen (píxeles).
- source: Nombre de la imagen o video en el cual se hará la detección.
- view-image: Mostrar los resultados.
- no-trace: No rastrear el modelo.
- device: El número de GPU (ID) que se usará para la detección. Puede ser 0,1…n o CPU.

4.5 Experimentación conjunto de datos Alumnos-UTM-902

Para el conjunto de datos Alumnos-UTM-902 se realizaron 4 experimentos para ver su desempeño con base a distintos entrenamientos realizados con 15 imágenes distintas, de las cuales las dos más relevantes se analizan a continuación.

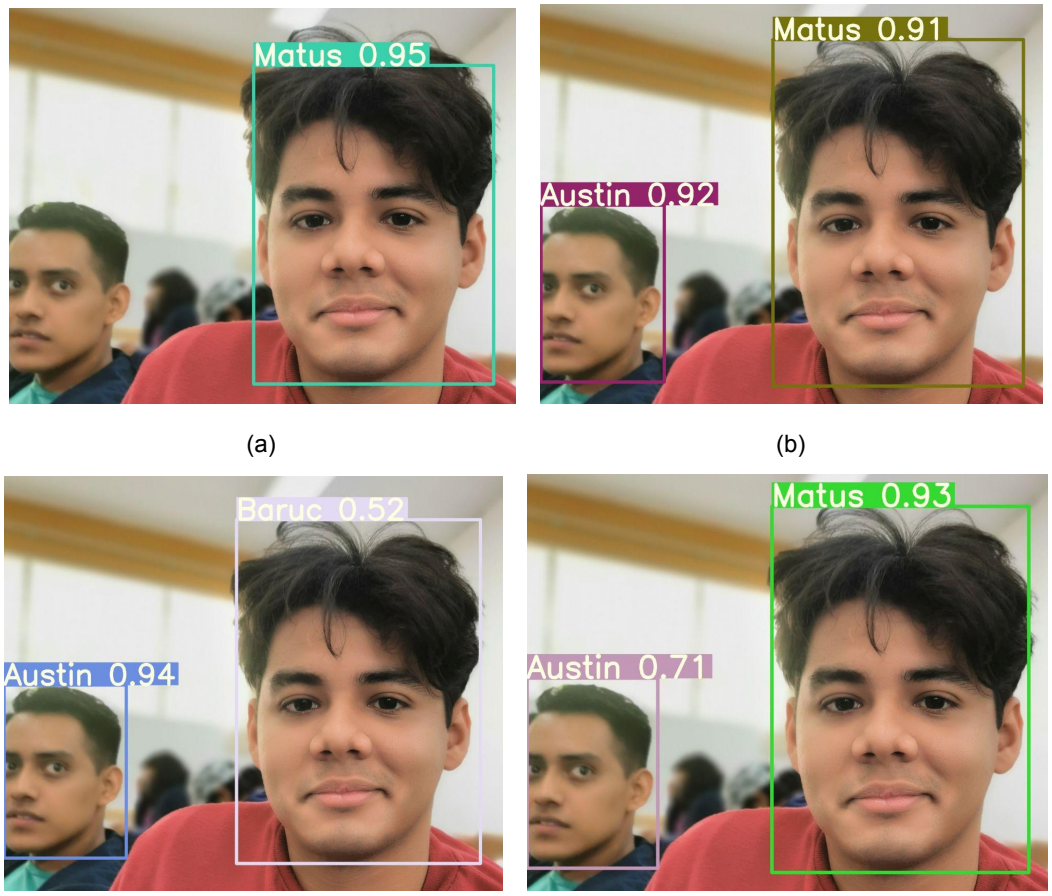
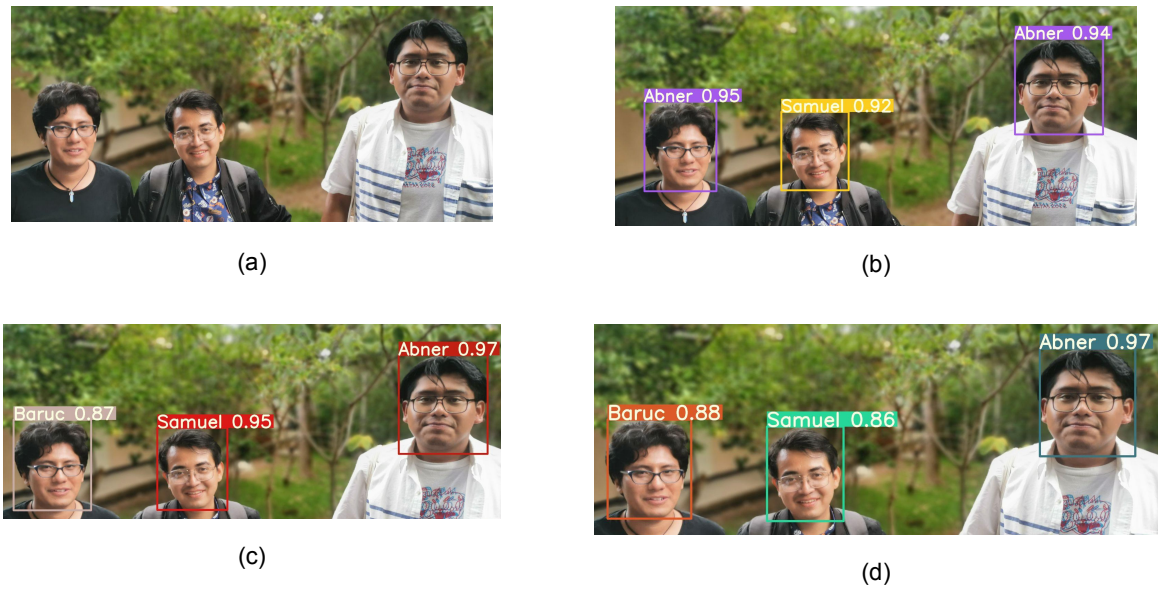


Figura 17. Imagen 1 con los experimentos realizados del grupo Alumnos-UTM-902. (a) Experimento 1 (b) Experimento 2 (c) Experimento 3 (d) Experimento 4

Como se puede observar, los resultados varían dependiendo del modelo con los que se haya realizado el experimento individual, se tiene el caso donde solo una de las personas fue etiquetada (Figura 17a), un caso en que una de las dos personas fue etiquetada erróneamente (Figura 17c) y dos casos donde ambas personas fueron etiquetadas de manera correcta (Figura 17b y d).

En otra caso tenemos una situación similar con la segunda imagen en la que para el primer experimento no se detectaron a ninguna de las 3 personas (Figura 18a), para la segunda y cuarta se etiquetó de manera correcta a las personas (Figura 18c y 18d), mientras que en el tercer experimento una de las personas fue etiquetada erróneamente (Figura 18b).



(a) (b)

(c) (d)

Figura 18. Imagen 2 con los experimentos realizados del grupo Alumnos-UTM-902. (a) Experimento 1 (b) Experimento 2 (c) Experimento 3 (d) Experimento 4

Los resultados de las imágenes pueden ser explicados mediante las gráficas de resultados obtenidas durante el entrenamiento (Figura 19). Estas gráficas corresponden a:

- **Box**: Función de pérdida del cuadro delimitador de los objetos detectados en el entrenamiento. Mide el error del bounding box de la detección frente al de las clases reales.
- **val Box**: Función de pérdida del **bounding box** de los objetos detectados en la validación.
- **Objectness**: Función de pérdida de la detección de objetos en el entrenamiento. Mide el error producido por no detectar objetos que se encontraban en las clases reales o por detectar objetos que no están etiquetados.
- **val Objectness**: Función de pérdida de la detección de objetos en la validación.
- **Classification**: Función de pérdida de la clasificación de objetos en el entrenamiento. Mide el error a la hora de clasificar los objetos detectados.
- **val Classification**: Función de pérdida de la clasificación de objetos en la validación.
- **Precision**: Precisión de la red neuronal a lo largo del entrenamiento.
- **Recall**: Recall de la red neuronal a lo largo del entrenamiento.
- **mAP@0.5**: Evolución de la Precisión media Promedio (mAP por sus siglas en inglés) para un umbral de Intersección sobre Unión del 50% a lo largo del entrenamiento.
- **mAP@0.5:0.95**: Evolución de la Precisión media Promedio para umbrales del 50% al 95% a lo largo del entrenamiento.

Donde el eje X representa las épocas de entrenamiento, y el eje Y los valores que toman las métricas.

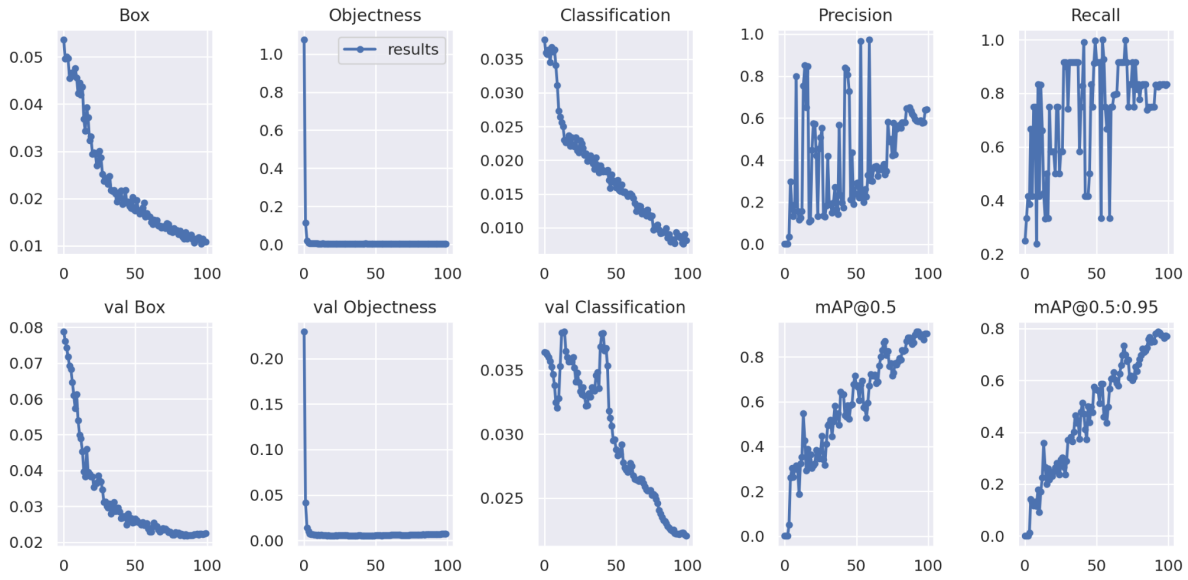


Figura 19. Resultados del entrenamiento y validación del experimento 1.

Para este proyecto el cuadro encapsulador *(bounding box)* que se muestra en las gráficas correspondientes y que está relacionado con la pérdida (*val Box*) no resulta de prioridad, sin embargo, se puede observar que en general tienden a cero, que es lo que habitualmente se busca en el caso de la pérdida. Por otro lado, la pérdida de detección de objetos (objectness) converge rápidamente a cero, lo que indica que los objetos fueron etiquetados correctamente durante el entrenamiento y no existen clases que pudieran causar ruido de fondo.

Con respecto a la pérdida de la clasificación (*classification*) en el entrenamiento resulta ser bastante buena, en constante decremento hasta llegar a una pérdida menor al 0.01, en cambio en la validación al comienzo parece mostrar una volatilidad pero logra estabilizarse hacia el final de igual manera.

Por su parte, los valores de Precisión media Promedio (mAP) convergen alrededor del 0.9 en un umbral del 50% y en un 0.78 en el umbral del 50% al 95% por ciento, lo que implica buenos resultados aunque aún no lo suficientemente buenos en lo que respecta a la detección correcta de las 6 clases de este grupo.

Para la validación del modelo se grafica la curva de Precisión-Recall (P-R, Figura 20) que en general las clases tuvieron un desempeño promedio, sin embargo la clase Abner es la que presenta la peor precisión con un deterioro mAP@0.5 de 0.448 frente al 0.904 de las 6 clases en conjunto.

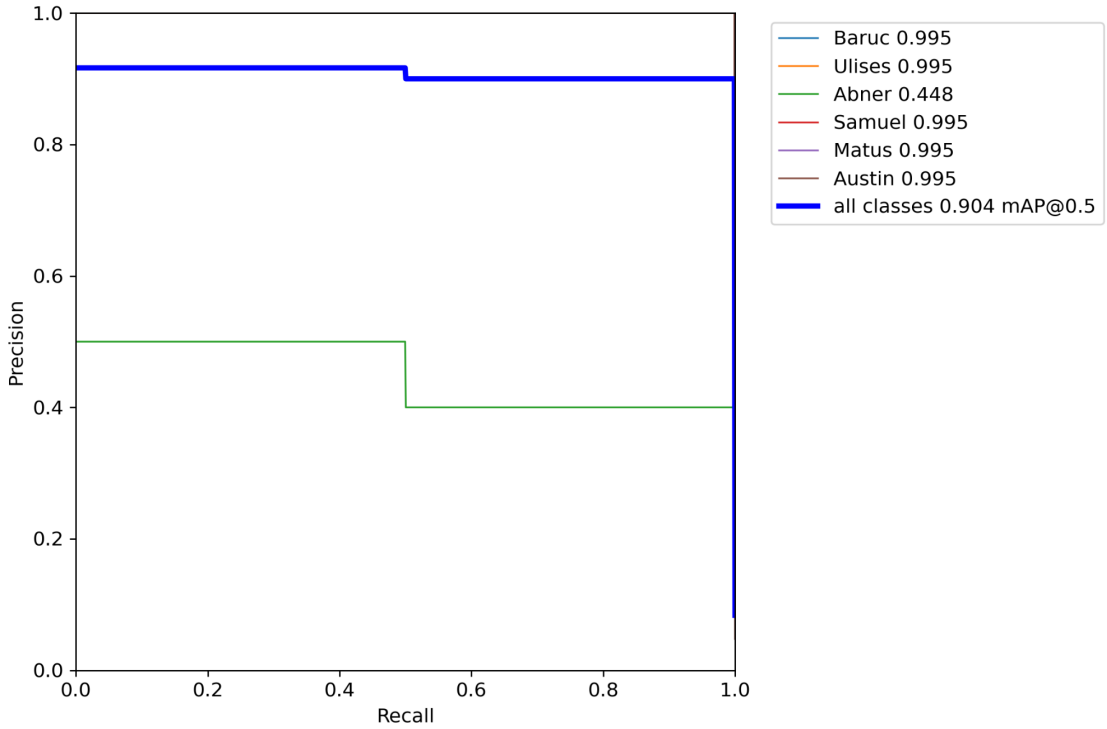


Figura 20. Curva Precisión-Recall para el experimento 1.

Una forma más clara de analizar esta información es mediante la matriz de confusión la cual es una matriz cuadrada de $C \; x \; C$, siendo C el número de clases a clasificar, aunque para el caso de imágenes complejas se agrega el ruido de fondo (Background FN). En el primer experimento se obtuvo una precisión de 64% quedando con una matriz de confusión como se muestra en la Figura 21.

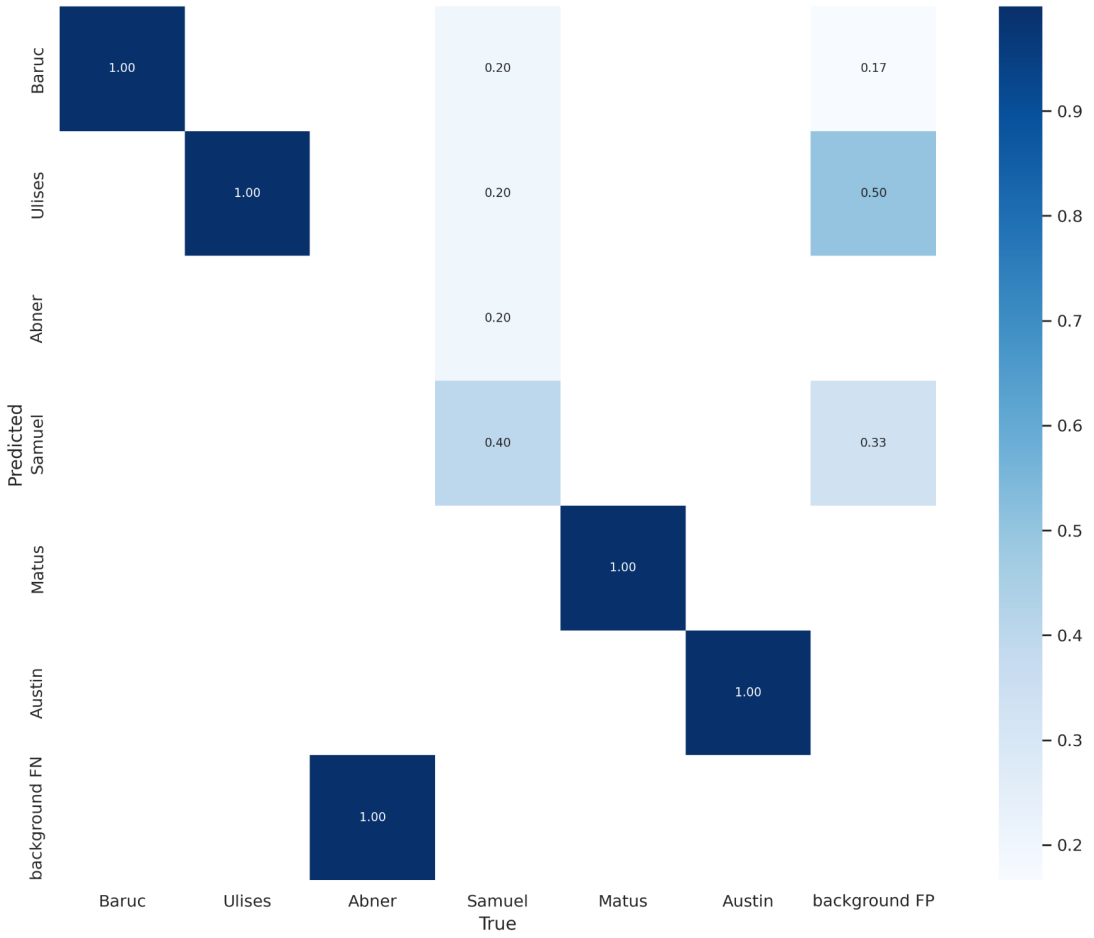


Figura 21. Matriz de confusión del modelos de probabilidad del experimento 1.

La matriz de confusión anterior indica que la clase Samuel es la clase que tiene más problemas, es decir, es la clase que más confunde y la clase Abner es la clase que no puede etiquetar, es decir, la clase que desconoce totalmente, lo cual implica resultados pobres al momento de clasificar.

Considerando los resultados anteriores, fue necesario hacer más experimentos modificando los parámetros de la red. Para esta serie de nuevos experimentos, los resultados se muestran en la Figura 22.

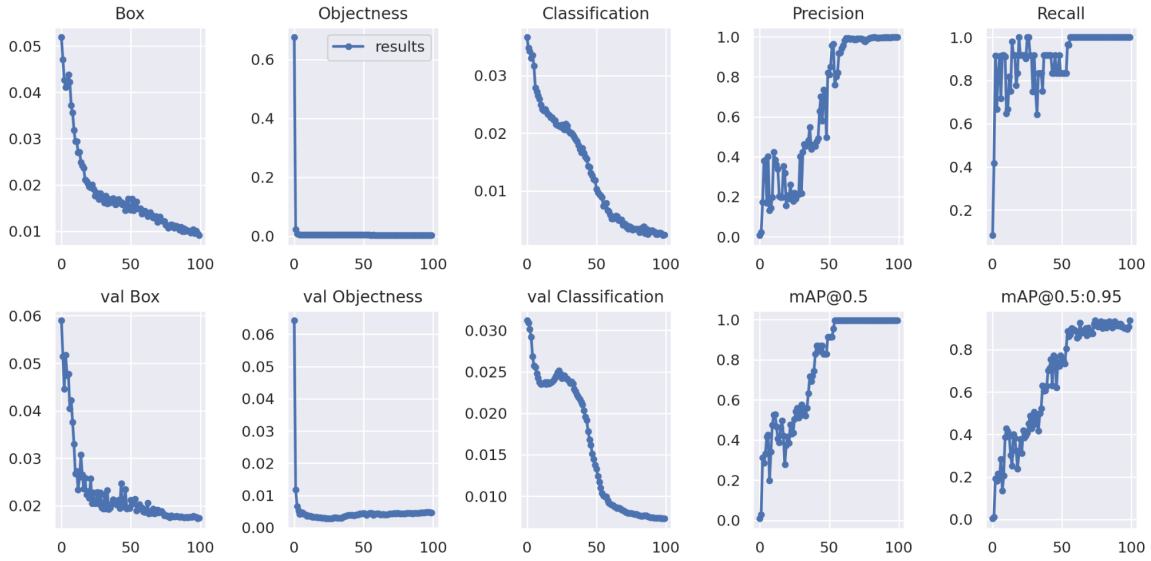


Figura 22 (a)

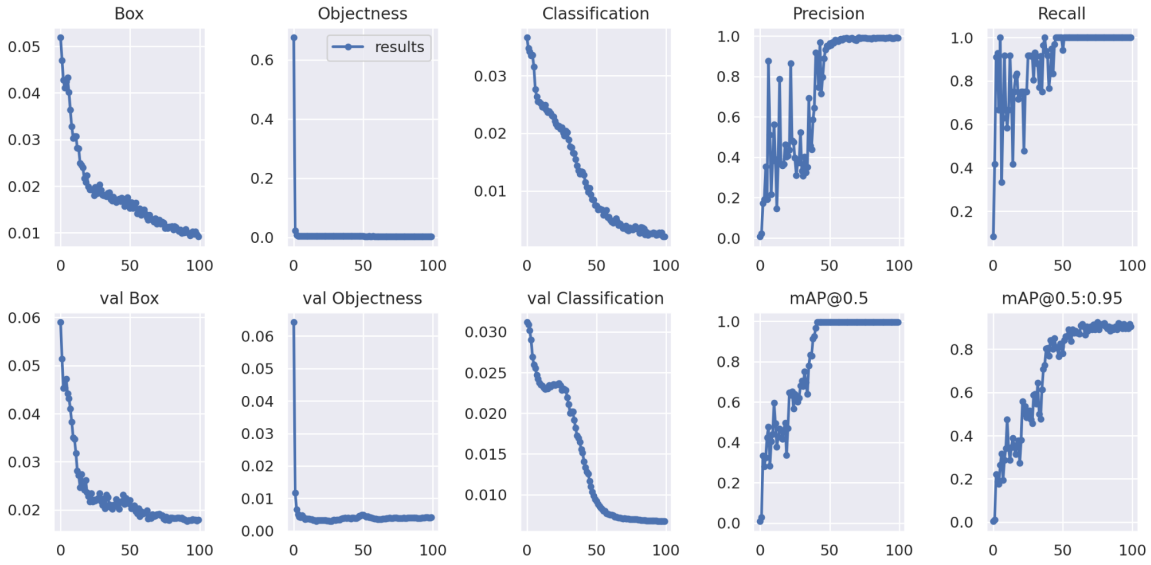


Figura 22 (b)

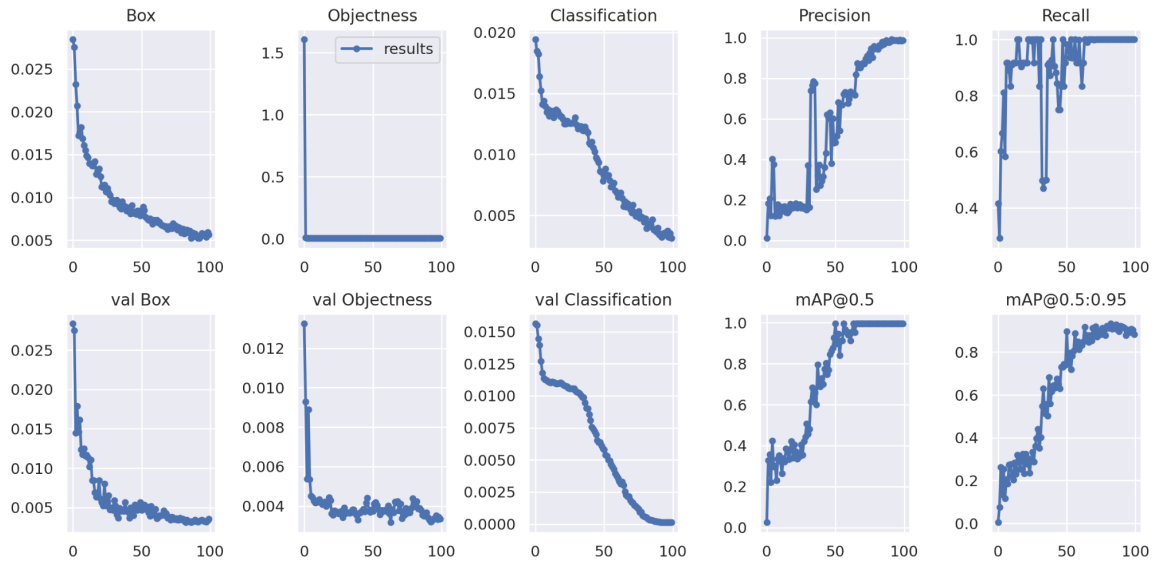


Figura 22 (c)

Figura 22. Resultados de entrenamiento y validación. (a) Experimento 2. (b) Experimento 3. (c) Experimento 4.

Para las gráficas de pérdida (box, objectness, classification) los resultados son muy similares entre sí y con el experimento original, se puede entonces inferir que la detección de las *bounding boxes* fue en general bastante buena, así como la pérdida para la detección de objetos con los tres experimentos tendiendo a 0. Con respecto a los valores de Precisión media Promedio (mAP), los tres experimentos convergieron a 1.0 con un umbral del 50%, y hacia un 0.95 con un umbral entre el 50-95%, siendo el experimento 3 (Figura 22c) la que mostró menor convergencia en este caso cercano a 0.9.

En la validación del modelo mediante la curva P-R (Figura 23) se observa claramente una mejora con respecto al experimento 1, sin embargo, los 3 experimentos mostraron resultados bastante similares con un mAP@0.5 de 0.996. Lo que implica un buen entrenamiento derivado del incremento del conjunto de datos.

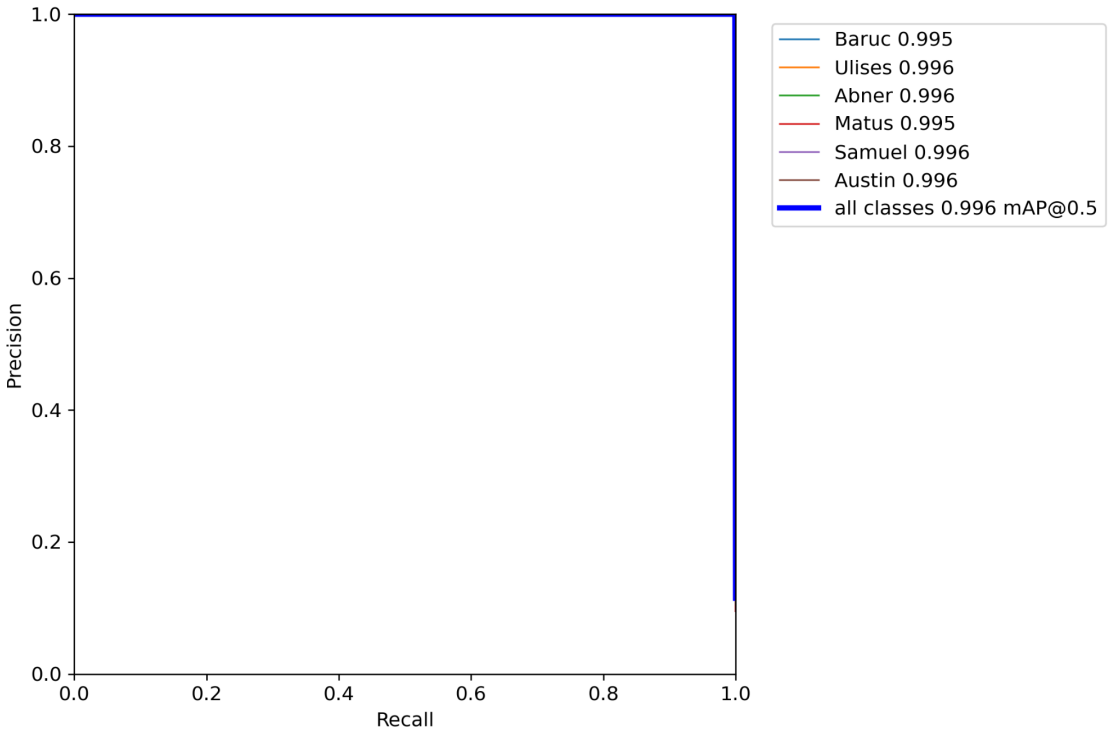


Figura 23 (a)

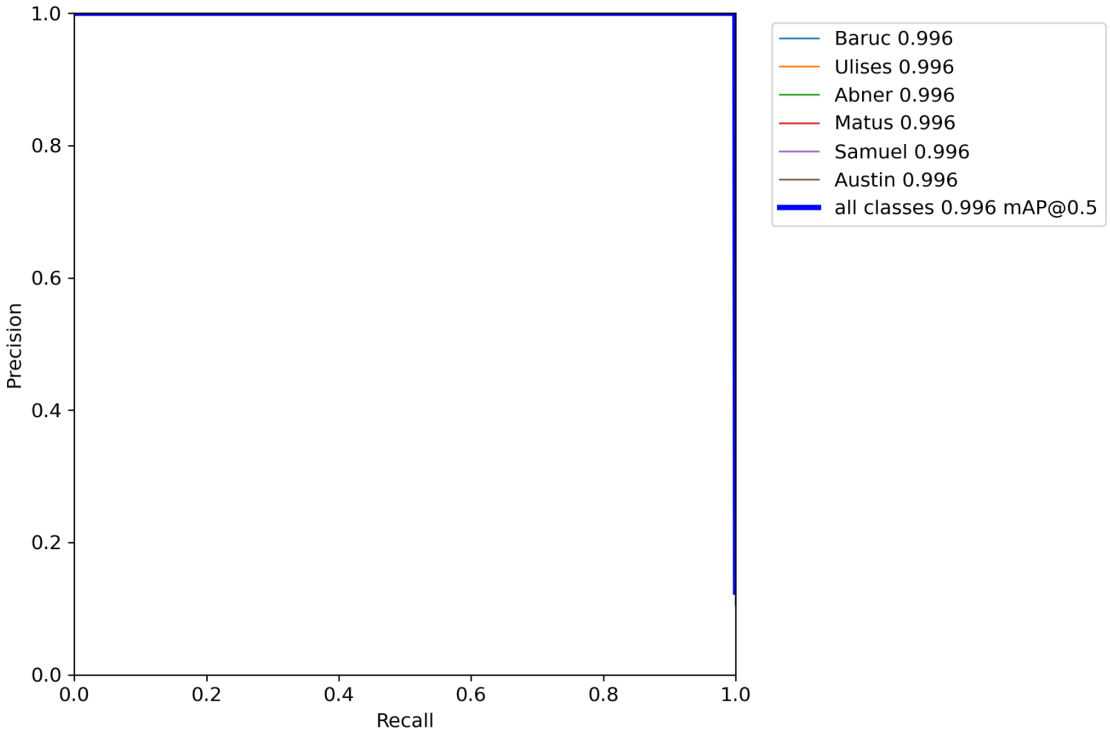


Figura 23 (b)

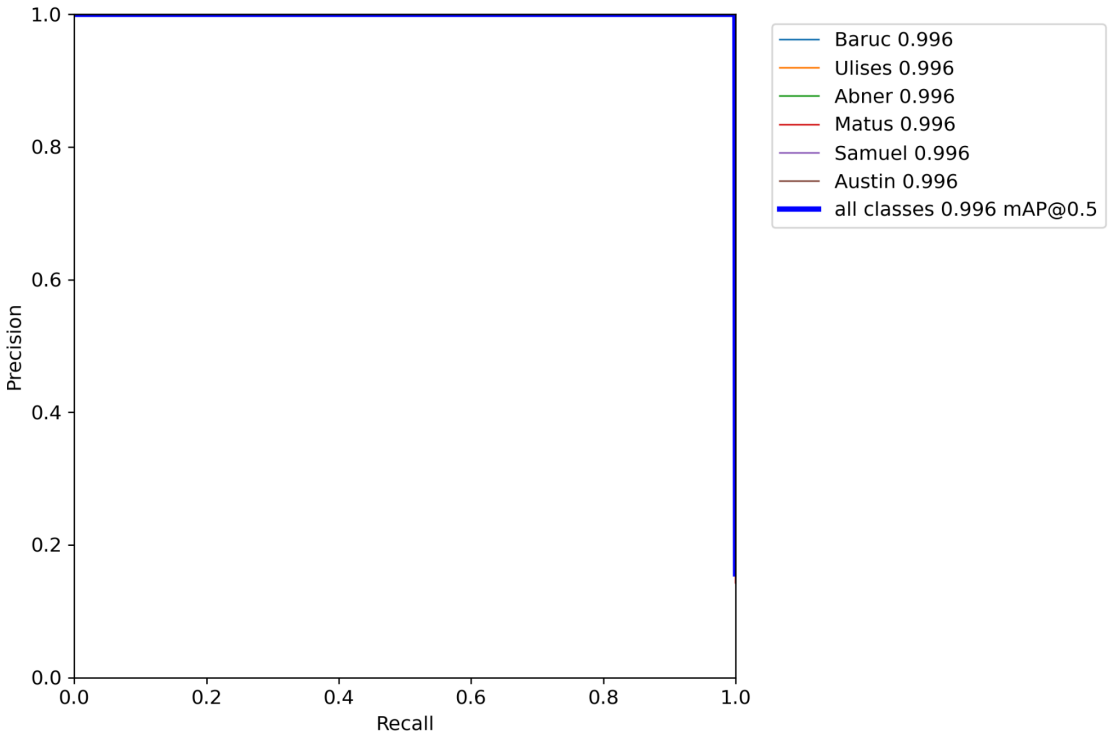


Figura 23 (c)

Figura 23. Curvas de Precisión-Recall. (a) Experimento 2. (b) Experimento 3. (c) Experimento 4.

Mediante la matriz de confusión se puede observar que se tienen resultados casi perfectos, que aunque podrían considerarse como posibles situaciones de sobre-ajuste, viendo las gráficas anteriores se observa que no es el caso. Clases como Matus o Austin mantuvieron su buena precisión, mientras que la clase Samuel y Abner logran mejorar sus resultados con respecto al primer experimento, con una pequeña excepción en el experimento 3 (Figura 24b) donde la clase Abner aún presentó un pequeño error al confundirse con ruido de fondo.

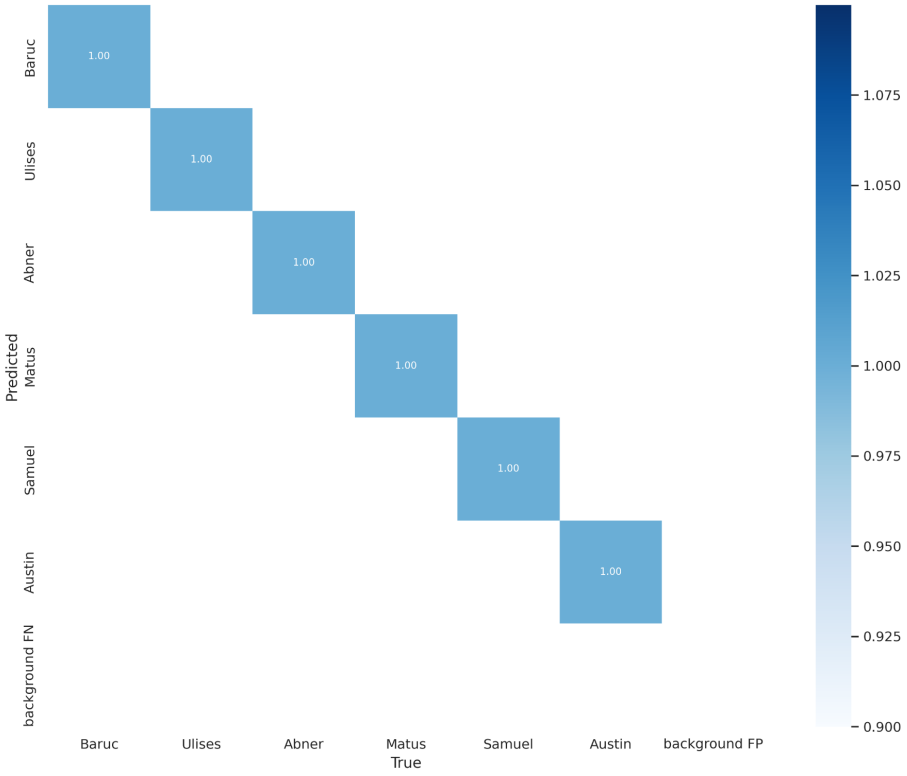

Baruc
Ulises
Abner
Matus
Samuel
Austin
background FN
Predicted
1.00
1.00
1.00
1.00
1.00
1.00
Baruc
Ulises
Abner
Matus
Samuel
Austin
background FP
True
1.075
1.050
1.025
1.000
0.975
0.950
0.925
0.900

Figura 24 (a)

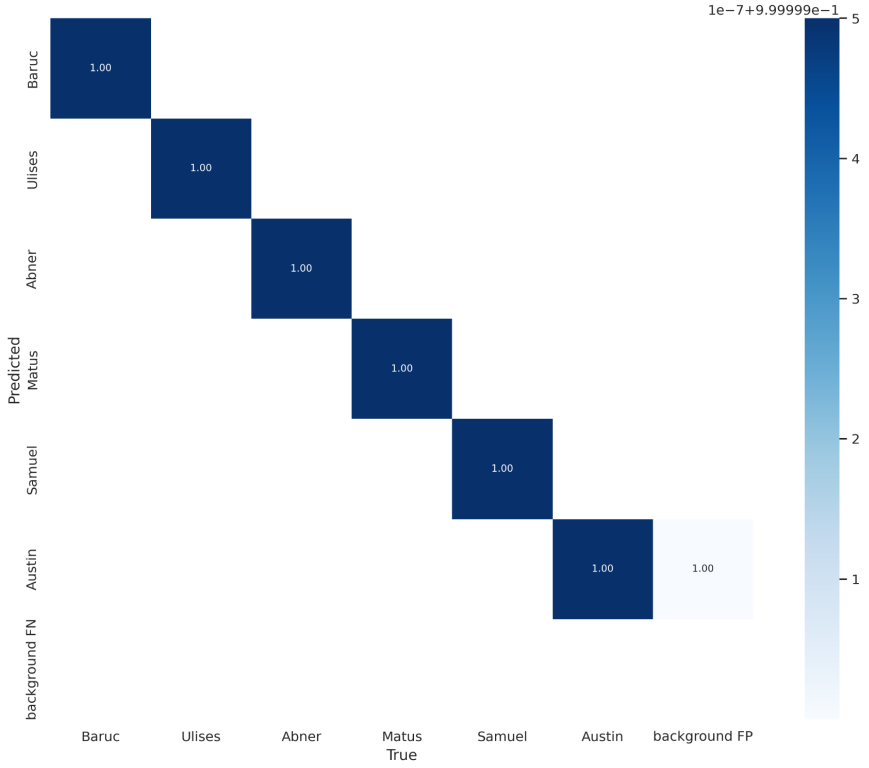

1e−7+9.99999e−1
5
4
3
2
1
Predicted
Baruc
Ulises
Abner
Matus
Samuel
Austin
background FN
1.00
1.00
1.00
1.00
1.00
1.00
1.00
Baruc
Ulises
Abner
Matus
Samuel
Austin
background FP
True

Figura 24 (b)

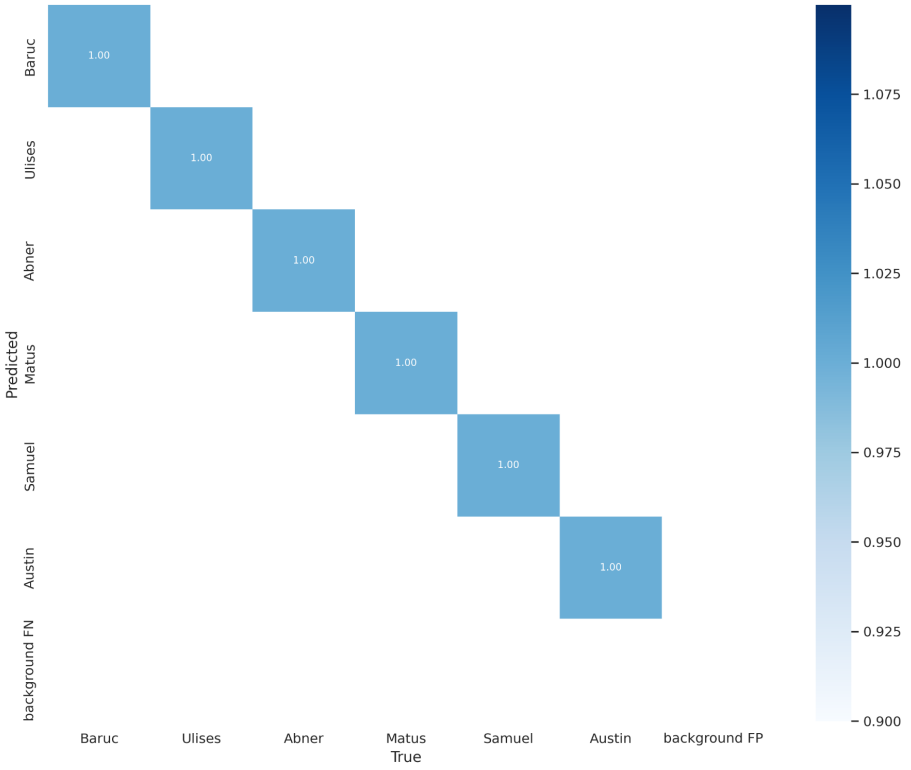


Figura 24 (c)

Figura 24. Matrices de confusión de probabilidades del modelo del conjunto de datos Alumnos-UTM-902. (a) Experimento 2. (b) Experimento 3. (c) Experimento 4.

4.6 Experimentación conjunto de datos Alumnos-UTM-602

Para el conjunto de datos Alumnos-UTM-602 se realizaron 2 experimentos para ver su desempeño en base a distintos entrenamientos realizados con 9 imágenes distintas, de las cuales las dos más relevantes se analizan a continuación denominadas como Imagen 1 e Imagen 2 respectivamente.

Figura 25 (a)

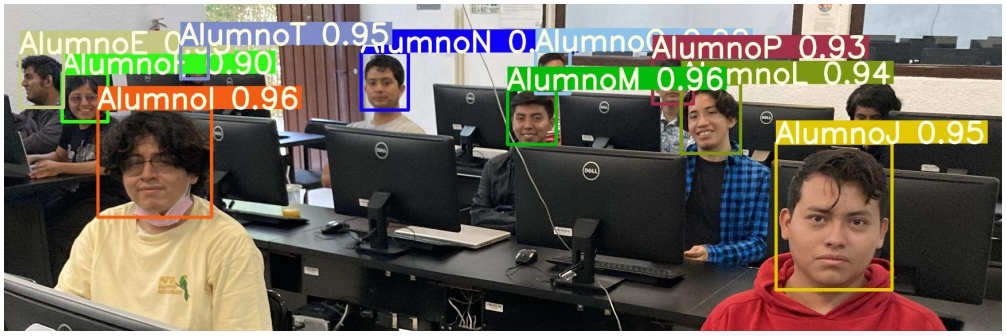
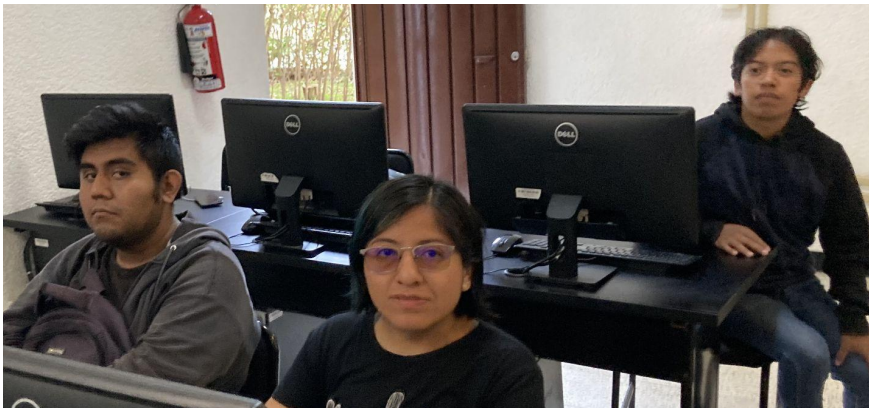


Figura 25 (b)

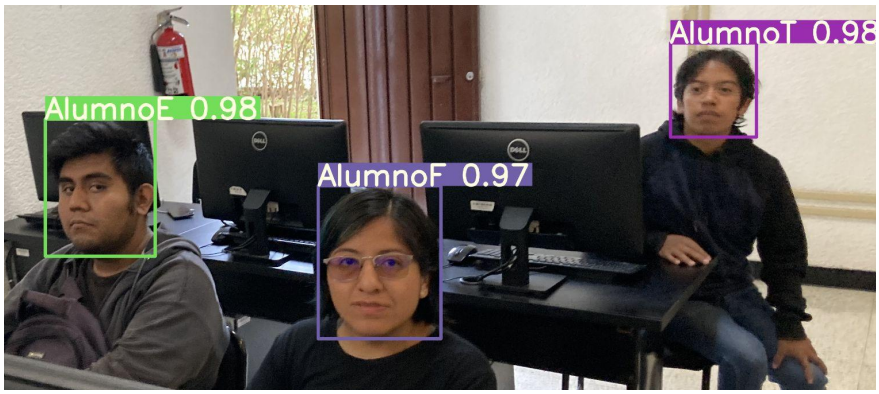
Figura 25. Imagen 1 con los experimentos realizados del grupo Alumnos-UTM-602 (a) Experimento 1. (b) Experimento 2.

En el caso de la imagen 1 , al igual que con el grupo Alumnos-UTM-902, los resultados varían con base a los pesos del entrenamiento. Se obtuvo el caso donde ninguna de las personas fue etiquetada (Figura 25a) mientras que en otro caso casi todos fueron etiquetados adecuadamente, con excepción de una persona (Figura 25b).

En otro caso probado tenemos una situación similar al de la imagen 2 en la que para la primera prueba no se detectaron a ninguna de las 3 personas (Figura 26a), mientras que en la segunda prueba las 3 personas fueron etiquetadas correctamente (Figura 26b).



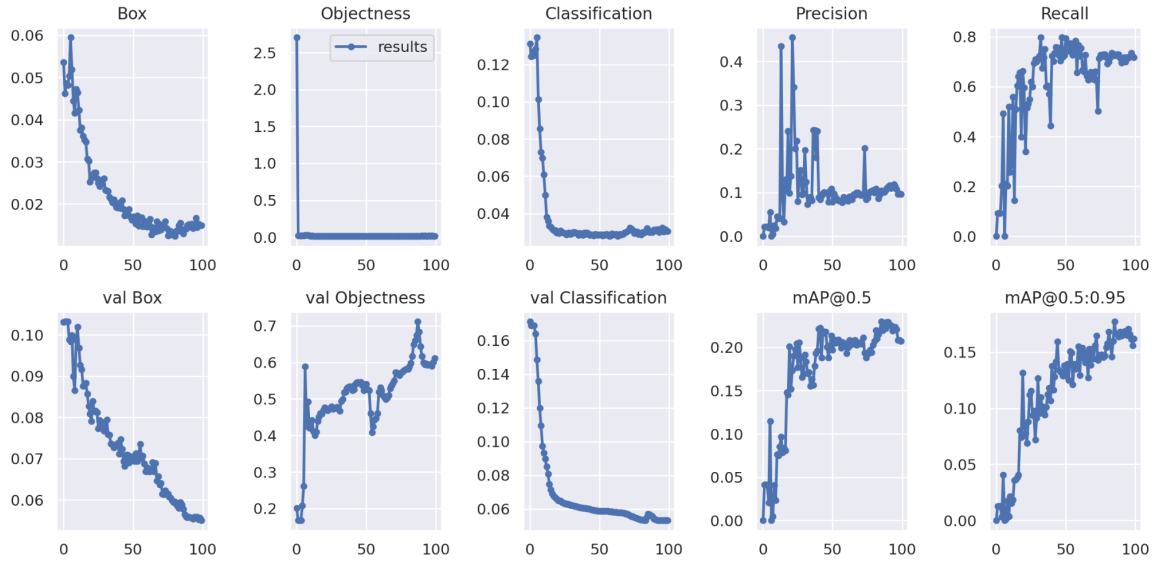
(a)



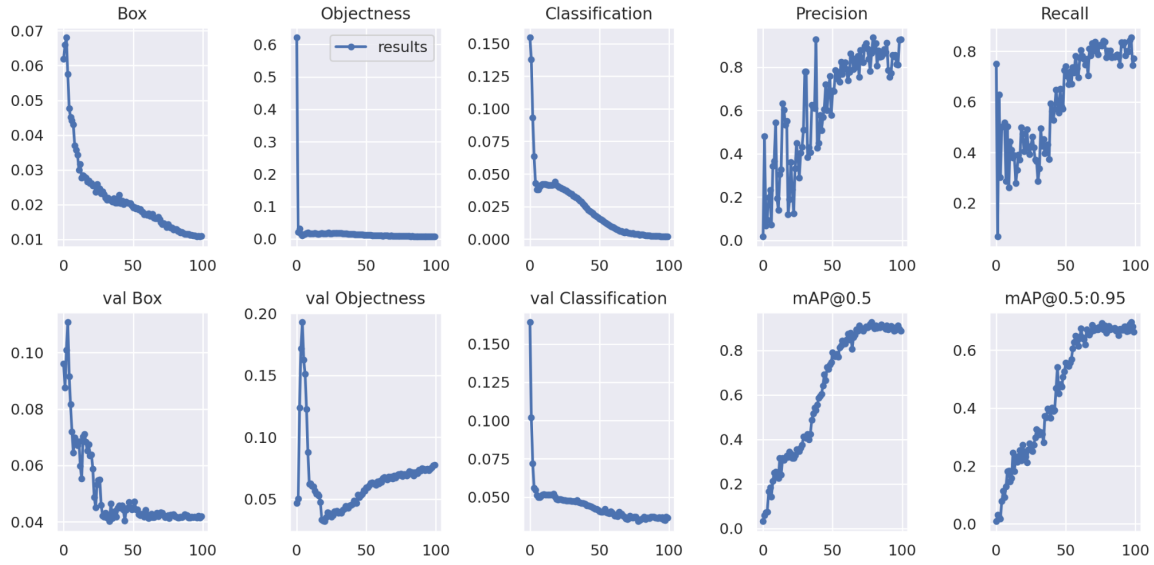
(b)

Figura 26. Imagen 2 con los experimentos realizados del grupo Alumnos-UTM-602. (a) Experimento 1. (b) Experimento 2.

Estos resultados al igual que los del conjunto de datos Alumnos-UTM-902 pueden explicarse mediante las gráficas obtenidas durante el entrenamiento (Figura 27). Los aspectos a considerar son principalmente dos: la cantidad de clases para este grupo, siendo 22 a comparación de las 6 del primer grupo, y el tamaño del conjunto de datos original.



(a)



(b)

Figura 27. Resultados de entrenamiento y validación del grupo Alumnos-UTM-602.

Como se mencionó previamente, la pérdida en la detección del *bounding box* no es un factor relevante en esta investigación. Por otro lado se puede observar la diferencia que hay en la pérdida de la detección de objetos (objectness), principalmente en el caso de la validación (val Objectness), donde para el caso de la experimento 1 se tiene una pérdida del 0.6 y en aumento, a diferencia del experimento dos que llega a un 0.07. Esto indica que el experimento 2 tuvo menores errores al detectar los objetos.

Mientras que en la pérdida de la clasificación (*classification*) ambos experimentos mostraron converger a 0.

En lo que respecta a la parte de mAP@0.5 se puede ver un mayor contraste con respecto al grupo Alumnos-UTM-902, donde en el primer experimento resultó con un valor de 0.9, en este caso apenas y sobrepaso un 0.2, mientras que para el umbral del 50-95% solo alcanzó el valor de 0.16. En cambio una vez aumentado el conjunto de datos de entrenamiento los resultados son muy diferentes, con un mAP@0.5 de 0.9 y un mAP@0.5:0.95 de 0.7, sigue sin ser el más óptimo pero supera con creces los del primer experimento.

En la parte de validación se ve el desempeño de manera más clara, analizando la curva P-R (Figura 28), se puede observar la precisión deficiente que hubo con prácticamente todas las clases siendo pocas las excepciones y teniendo un mAP@0.5 de 0.207. En contraste, una vez aumentado el conjunto de datos la mayoría de las clases muestran precisión promedio arriba del 88.7%, con pocas clases teniendo una menor precisión, pero aún así mucho mejor que en el primer experimento.

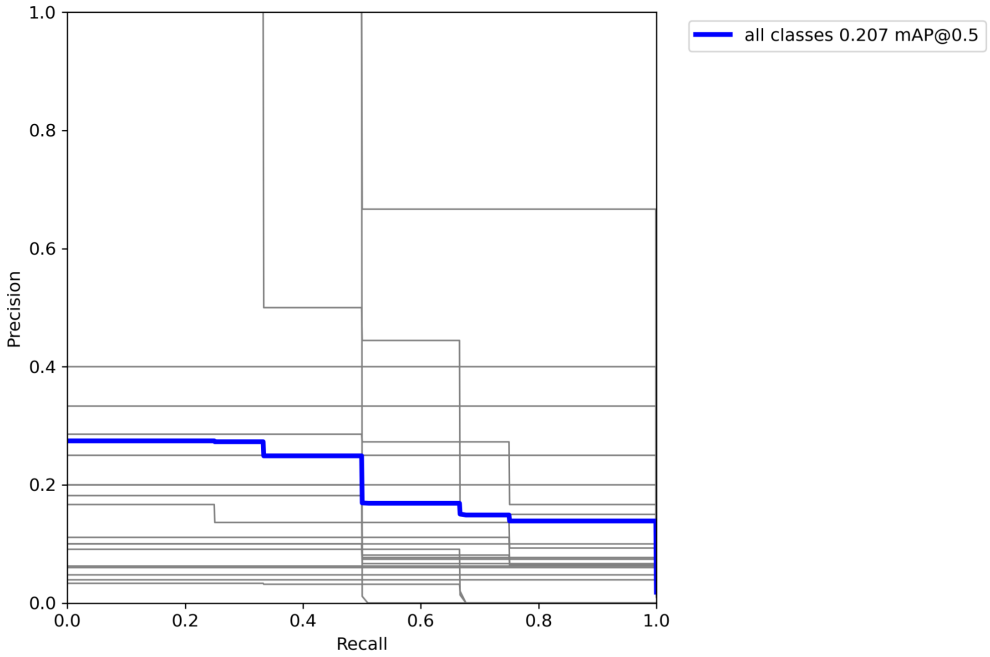


Figura 28 (a)

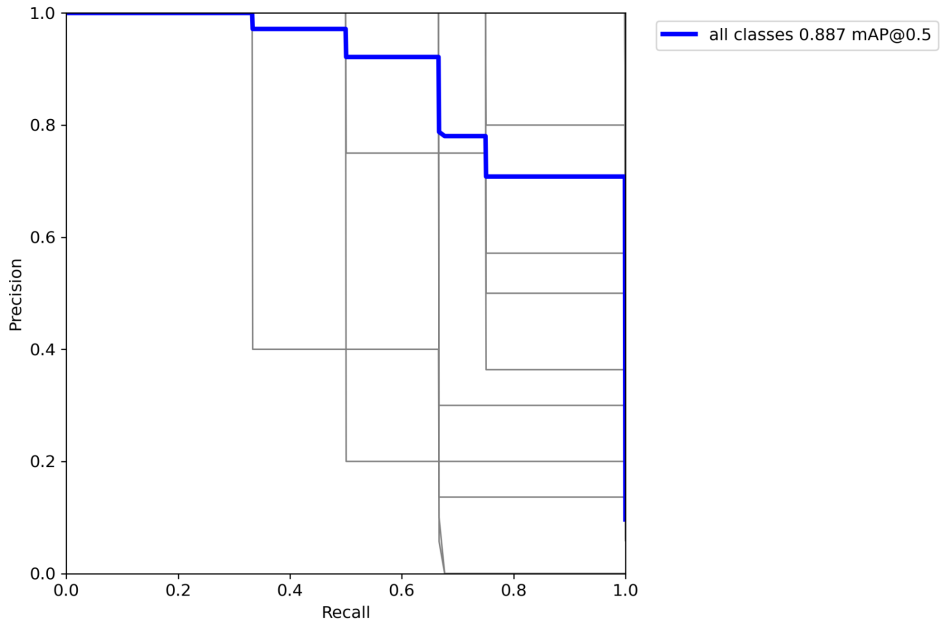


Figura 28 (b)

Figura 28. Curvas de Precisión-Recall del grupo Alumnos-UTM-602.

Por último y para que queden más claros los resultados se analizaron las matrices de confusión de ambas pruebas (Figura 29). La primera matriz corresponde a la primera prueba y muestra justamente la poca precisión que se alcanzó, siendo que ninguna clase fue etiquetada correctamente y más bien fueron detectadas como ruido de fondo.

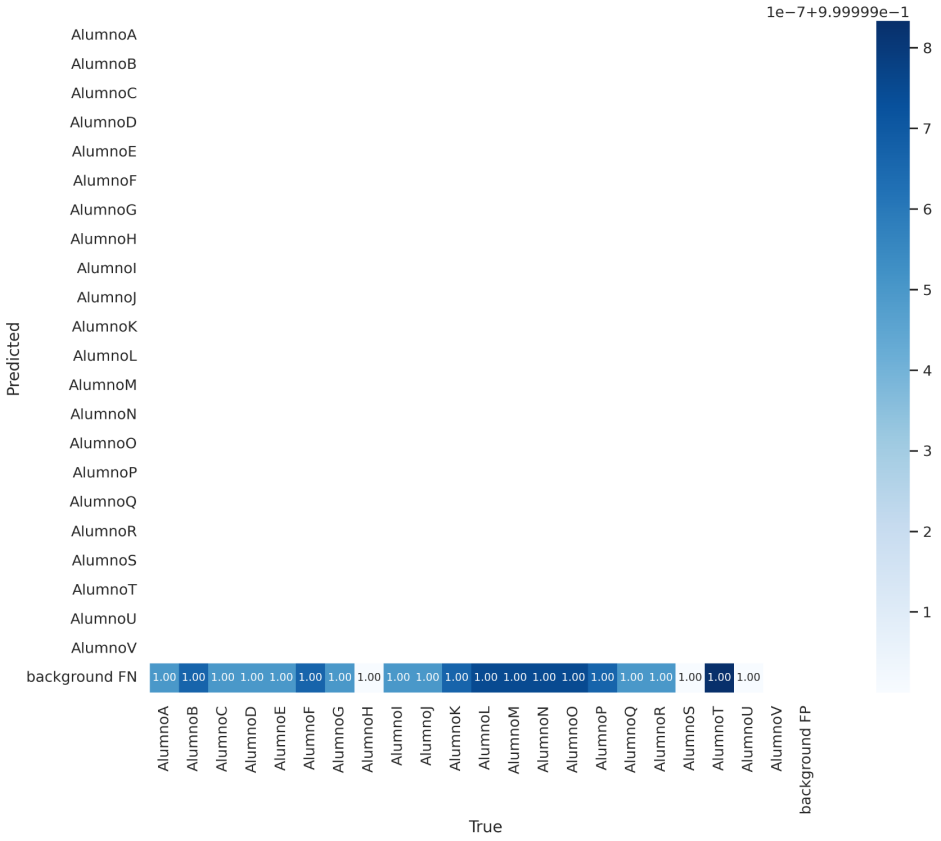

1e−7+9.99999e−1
AlumnoA
AlumnoB
AlumnoC
AlumnoD
AlumnoE
AlumnoF
AlumnoG
AlumnoH
AlumnoI
AlumnoJ
AlumnoK
AlumnoL
AlumnoM
AlumnoN
AlumnoO
AlumnoP
AlumnoQ
AlumnoR
AlumnoS
AlumnoT
AlumnoU
AlumnoV
background FN
Predicted
1.00
AlumnoA
AlumnoB
AlumnoC
AlumnoD
AlumnoE
AlumnoF
AlumnoG
AlumnoH
AlumnoI
AlumnoJ
AlumnoK
AlumnoL
AlumnoM
AlumnoN
AlumnoO
AlumnoP
AlumnoQ
AlumnoR
AlumnoS
AlumnoT
AlumnoU
AlumnoV
background FP
True
8
7
6
5
4
3
2
1

Figura 29 (a)

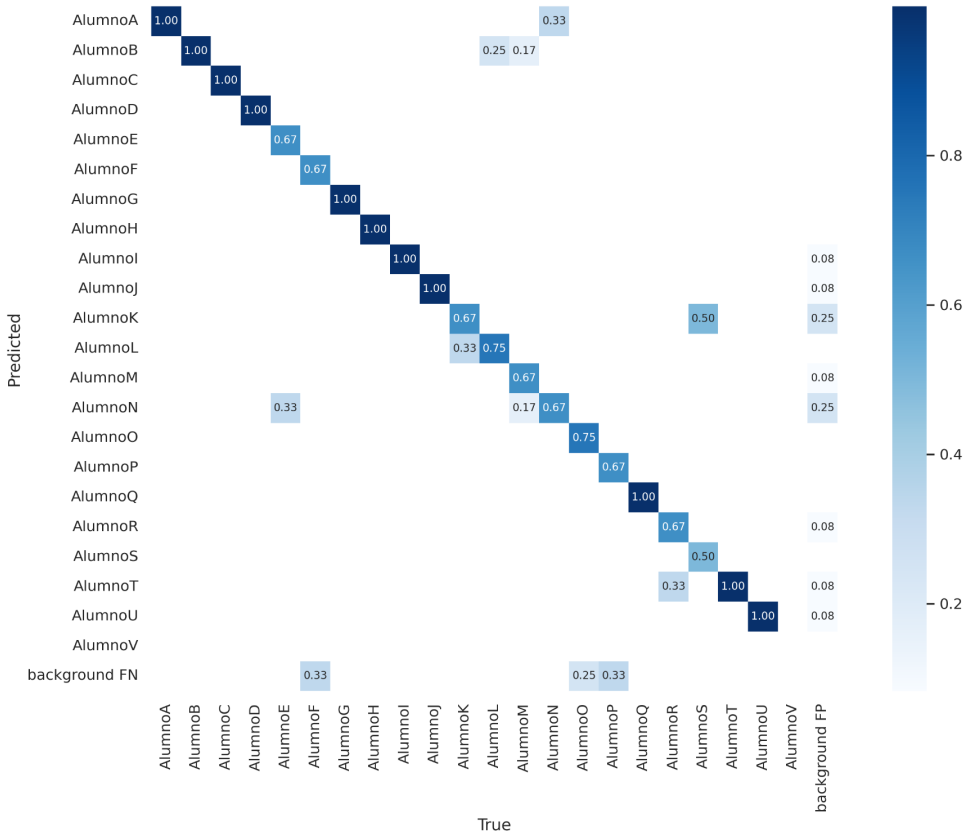


Figura 29 (b)

Figura 29. Matrices de confusión del grupo Alumnos-UTM-602.

Por otro lado, en el caso de la segunda prueba se tiene que la mayoría de las clases fue correctamente etiquetada (Figura 29b). Siendo que hubo aún cierta confusión entre algunas de estas clases como es el caso de los Alumnos E,F, K, L, entre otros, lo cual se debe a la distancia de las personas en las imágenes del conjunto y las pocas veces que aparecen comparados con las otras clases o también considerar semejanzas físicas entre las personas que complican la diferenciación entre las clases.

Capítulo 5: Conclusiones y trabajo a futuro

Con base en el análisis de los resultados se observó el desempeño de una red neuronal convolucional para poder reconocer distintos rostros a través de imágenes, tomando en cuenta distintos factores como son la cantidad de imágenes con las que se elaboró el conjunto de datos, la calidad de éstas y bajo qué condiciones se tomaron. Entonces, con base a los experimentos realizados se concluye que efectivamente se puede conseguir una precisión de más del 88.7% en el caso de la identificación de rostros mediante una red neuronal para el grupo Alumnos-UTM-602, y mayor al 90% para el grupo Alumnos-UTM-902. Sin embargo, la calidad de esta precisión está sujeta en gran medida a la calidad del conjunto de datos para el entrenamiento, principalmente entre mayor cantidad sea la cantidad de clases (alumnos) a identificar mayor cantidad de fotos requerirá la red para obtener mejores resultados, es por esto que técnicas de aumento de datos mostraron ser un factor importante en la mejora de los conjuntos para el entrenamiento.

En general, con los resultados obtenidos se concluye que es posible el uso de redes convolucionales para el reconocimiento de rostros mediante imágenes con resultados razonablemente altos, aunque es necesario considerar la cantidad de clases para así crear un conjunto de datos que esté refinado para obtener buenos resultados en el entrenamiento de esta red.

5.1 Aportaciones

Uno de los logros importantes de este trabajo de investigación es la adquisición y etiquetado de las imágenes de dos conjuntos de datos (Alumnos-UTM-902, Alumnos-UTM-602).

La comprobación del uso del aumento de datos para mejorar el modelo del segundo conjunto de datos (Alumnos-UTM-602).

La configuración adecuada de los parámetros de Yolov7 para lograr una alta precisión en la automatización del pase de lista mediante reconocimiento de rostros.

5.2. Trabajo futuro

Así pues, debido a los resultados obtenidos y sus potenciales aplicaciones, se deja como trabajo a futuro una implementación de un pase de lista mediante una aplicación móvil, lo cual requerirá de no solo el desarrollo de la aplicación, si no también de una metodología para la captura de las imágenes, de tal manera que se mantenga cierta calidad en ellas para el entrenamiento. Por otra parte, de implementarse la aplicación es importante establecer una metodología para asegurarse de que los resultados sigan manteniendo la precisión, lo que implica la intervención del profesor para asegurarse que los alumnos sean cooperativos con el sistema para la toma de imágenes, y que además se mantengan las imágenes tomadas en caso de que se soliciten aclaraciones en la asistencia mediante un reporte.

Bibliografía.

- [1] *Asistencia en Meet*. (s. f.). Chrome Web Store. https://chrome.google.com/webstore/detail/meet-attendance/nenibigflkdikhamlnekfppbganmojlg?hl=es
- [2] Buddhiwant, A., Bharkshe, M., Bansod, R., & Chandekar, M. (2017). Smart Attendance Application. *International Journal of Engineering and Management Research*, *7*(2), 221-224. http://www.indianjournals.com/ijor.aspx?target=ijor:ijemr&volume=7&issue=2&article=041
- [3] Dev, S., & Patnaik, T. (2020). Student Attendance System using Face Recognition. 2020 International Conference on Smart Electronics and Communication (ICOSEC). doi:10.1109/icosec49089.2020.9215441
- [4]*IBM*. (s. f.). *What are Convolutional Neural Networks?* https://www.ibm.com/topics/convolutional-neural-networks
- [5] Islam, M. M., Hasan, M. K., Billah, M. M., & Uddin, M. M. (2017). Development of smartphone-based student attendance system. 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC). doi:10.1109/r10-htc.2017.8288945
- [6] ITWORX EDU FZ LLC (2022). *TeacherKit* (2.22.0) [Aplicación móvil]. Google Play. https://play.google.com/store/apps/details?id=com.teacherkit.app&hl=es&gl=US
- [7] Khan, S., Rahmani, H., Shah, S. A. A., & Bennamoun, M. (2018). A Guide to Convolutional Neural Networks for Computer Vision. *Synthesis lectures on computer vision*, *8*(1), 1-207. https://doi.org/10.2200/s00822ed1v01y201712cov015
- [8] LeCun, Y., Kavukcuoglu, K., & Farabet, C. (2010). Convolutional networks and applications in vision. *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*.
- [9] Lim, T. S., Sim, S. C., & Mansor, M. M. (2009). RFID based attendance system. 2009 IEEE Symposium on Industrial Electronics & Applications. doi:10.1109/isiea.2009.5356360

- [10] Lin, Z., & Li, Y. (2019). Design and Implementation of Classroom Attendance System Based on Video Face Recognition. 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS). doi:10.1109/icitbs.2019.00101
- [11] Murphy, K. P. (2022). *Probabilistic Machine Learning*. MIT Press
- [12] *Pasar Lista en Meet*. (s. f.). Chrome Web Store. https://chrome.google.com/webstore/detail/pasar-lista-en-meet/fcikoaljfkfdofbhmfeonmpkmmagnokd?hl=es-419
- [13] Saccone, M. (2020). La asistencia a clases de los estudiantes en la educación media superior. Aportes desde una investigación etnográfica en la Ciudad de México. *Revista latinoamericana de estudios educativos*, *50*(2), 55-88. https://doi.org/10.48102/rlee.2020.50.2.59
- [14] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, *6*(1), 1-48.
- [15] Wang, C., Yeh, I., & Liao, H. M. (2021e). You Only Learn One Representation: Unified Network for Multiple Tasks. *arXiv (Cornell University)*. http://export.arxiv.org/pdf/2105.04206
- [16] Wang, C., Bochkovskiy, A., & Liao, H. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2207.02696
- [17] Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M., & Shum, H. (2022). DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2203.03605

Anexos

Anexo 1. Comando para la instalación de labelImg

```
pip install labelImg
```

Anexo 2. Comando para la ejecución de labelImg

```
labelImg
```

Anexo 3. Comando para entrenamiento de yolov7

```
python train.py --workers 1 --device 0 --batch-size 2 --epochs 100 --img 640 640 --data data/custom_data.yaml --hyp data/hyp.scrath.custom.yaml --cfg cfg/training/yolov7-custom.yaml --name yolov7-custom --weights yolov7.pt
```

Anexo 4. Comando para detección de yolov7

```
python detect.py --weights best.pt --conf 0.5 --img-size 640 --source img03.jpg --view-img --no-trace --device cpu
```

Anexo 5. Especificaciones de software utilizadas en este proyecto.

- Python 3.9.16
- matplotlib 3.7.1
- numpy 1.23.3
- opencv-python 4.7.0.72
- Pillow 9.5.0
- PyYAML 6.0
- requests 2.31.0
- scipy 1.10.1
- tqdm 4.65.0
- protobuf 4.21.2
- tensorboard 2.13.0
- pandas 2.0.2
- seaborn0.12.2
- torch 1.11.0+cu113
- torchvision 0.12.0+cu113