

ASISTENTE DE CONCENTRACIÓN IMPULSADO POR
INTELIGENCIA ARTIFICIAL PARA ESTUDIANTES
UNIVERSITARIOS

Alumno: Guaimas Gonzalo Bartolomé
Profesor Guía: Rivera Bernsdorff Fernando Lucas
Universidad Católica de Salta
Carrera: Ingeniería en Informática

Título: “Asistente de concentración impulsado por Inteligencia Artificial para estudiantes universitarios”

Carrera: Ingeniería en informática

Profesor Guía:

Nombre: Mg. Ing. Rivera Bernsdorff, Fernando Lucas

Firma:



Tribunal Evaluador:

Nombre:

Firma:

Nombre:

Firma:

Nombre:

Firma:

Alumno:

Nombre: Guaimás, Gonzalo Bartolomé

Firma:

Fecha de Exposición: __/__/____

ÍNDICE DE CONTENIDOS

RESUMEN	9
1 INTRODUCCIÓN	10
2 PRELIMINARES	11
2.1 DEFINICIÓN DEL PROBLEMA	11
2.2 OBJETIVOS DEL PROYECTO	12
2.2.1 OBJETIVO GENERAL	12
2.2.2 OBJETIVO ESPECÍFICO	12
2.3 ALCANCE DEL PROYECTO	12
3 MARCO TEÓRICO	14
3.1 Inteligencia Artificial	14
3.2 Minería de datos	14
3.2.1 Tipos de aprendizajes en minería de datos	15
3.2.2 Algoritmos de aprendizaje supervisado	16
3.3 Aprendizaje profundo - <i>Deep learning</i>	18
3.4 Detección de objetos	19
3.4.1 Clasificación de Algoritmos de detección de objetos	20
3.4.2 Métodos de Detección de Objetos	21
3.5 YOLO (<i>You Only Look Once</i>)	23
3.5.1 Evolución de YOLO	24
3.6 CRISP-DM	25
3.7 Roboflow	27
3.7.1 Herramienta de Anotación	27
3.8 Técnica Pomodoro	27
3.9 Desarrollo Incremental	27
3.10 Node.js	28
3.11 React.js	28
3.12 Next.js	28
3.13 Nest.js	29
3.14 ONNX (<i>Open Neural Network Exchange</i>)	30
4 SOLUCIÓN PROPUESTA	31

4.1 SOLUCIÓN AL PROBLEMA	31
4.2 METODOLOGÍA	31
4.3 PLANIFICACIÓN	33
4.3.1 Estructura de Desglose del Trabajo (EDT)	33
4.3.2 Diagrama Gantt	35
4.3.3 RECURSOS	36
4.3.4 ANÁLISIS FODA	37
4.3.5 MATRIZ DE PROBABILIDAD IMPACTO	39
4.3.6 PLAN DE CONTINGENCIA	41
4.3.7 ESTUDIO DE FACTIBILIDAD	43
5 DESARROLLO DEL PROYECTO	49
5.1 Introducción	49
5.2 Etapa 1: Desarrollo del modelo	49
5.2.1 Comprensión del negocio	50
5.2.2 Comprensión de los datos	50
5.2.3 Preparación de los datos	52
5.2.4 Exportación del conjunto de datos	55
5.2.5 Entrenamiento de los modelos	56
5.2.6 Entrenamiento 1: YOLOv5 100 épocas	58
5.2.7 Entrenamiento 2: YOLOv8 100 épocas	60
5.2.8 Evaluación: YOLOv5 y YOLOv8 - 100 épocas	62
5.2.9 Entrenamiento 3: YOLOv8 100 épocas con <i>augmentation</i>	64
5.2.10 Entrenamiento 4: YOLOv8 200 épocas con menos <i>augmentation</i>	68
5.2.11 Entrenamiento 5: YOLOv8 200 épocas con poco <i>augmentation</i> y otro enfoque	70
5.2.12 Entrenamiento 6: YOLOv8 200 épocas con todas las técnicas de <i>augmentation</i> y otro enfoque	72
5.2.13 Evaluación final y selección del modelo	72
5.3 Etapa 2: Desarrollo del aplicativo web	74
5.3.1 Incompatibilidad entre Python y JavaScript	75
5.3.2 Solución incompatibilidad entre Python y JavaScript	75
5.3.3 Desarrollo sección <i>Landing page</i>	76
5.3.5 Desarrollo sección Aplicativo – Asistente de Concentración	78
5.3.6 Desarrollo lógico del reloj de simulación Pomodoro	79

5.3.7 Problema de rendimiento en la implementación del modelo	79
5.3.8 Desarrollo back-end	79
5.3.9 Arquitectura y diseño lógico final utilizado	80
6 RESULTADOS.....	82
7 FUTURAS LÍNEAS DE INVESTIGACIÓN	86
8 CONCLUSIÓN.....	87
BIBLIOGRAFÍA.....	88
ANEXOS.....	90

ÍNDICE DE TABLAS

Tabla 1: Diccionario EDT	35
Tabla 2: Matriz de probabilidad e impacto	40
Tabla 3: Costo de desarrollo del prototipo por perfiles y cantidad de horas	44
Tabla 4: Tabla de conclusiones de modelos entrenados	73

ÍNDICE DE FIGURAS

Ilustración 1: Definiciones de inteligencia artificial organizadas en cuatro categorías (Russell et al., 2004).	14
Ilustración 2: Ciclo de vida de minería de datos	26
Ilustración 3: Estructura de Desglose de Trabajo	33
Ilustración 4 Planificación del proyecto	35
Ilustración 5 Diagrama de Gantt	36
Ilustración 6: Proceso de recopilación de imágenes para crear conjunto de datos “Estado concentrado”	50
Ilustración 7: Ejemplo de captura de imagen "Estado desconcentrado"	51
Ilustración 8: Ejemplo de captura de imagen "Estado uso del teléfono"	51
Ilustración 9: Creación de proyecto en plataforma Roboflow	52
Ilustración 10: Importación del conjunto de datos en la plataforma Roboflow	53
Ilustración 11: Etiquetado estado “concentrado” en plataforma Roboflow	53
Ilustración 12: Etiquetado estado “desconcentrado” en plataforma Roboflow	54
Ilustración 13: Etiquetado estado "uso del teléfono" en plataforma Roboflow	54
Ilustración 14: Herramienta de división de conjunto de datos en Roboflow	55
Ilustración 15: Proceso de exportación del conjunto de datos en Roboflow	55
Ilustración 16: Opciones de data augmentation en Roboflow	56
Ilustración 17: Comandos para descargar conjunto de datos desde Roboflow	57
Ilustración 18: Comando iniciales para entrenar modelo con YOLO en Google Colaboratory	57
Ilustración 19: Tabla de modelos pre entrenados para utilizar con YOLOv5	58
Ilustración 20: Finalización de primer entrenamiento de modelo con YOLOv5	59
Ilustración 21: Tabla de modelos pre entrenados para utilizar con YOLOv8	61
Ilustración 22: Finalización de entrenamiento con YOLOv8 con 100 épocas	61
Ilustración 23: Comparación de estado “concentrado” entre modelos entrenados con YOLOv5 y YOLOv8	63
Ilustración 24: Comparación de estado “desconcentrado” entre modelos entrenados con YOLOv5 y YOLOv8	63
Ilustración 25: Comparación de estado "uso del teléfono" entre modelos entrenados con YOLOv5 y YOLOv8	63
Ilustración 26: Primera parte de opciones elegidas de data augmentation	64
Ilustración 27: Segunda parte de opciones elegidas de data augmentation	65
Ilustración 28: Técnicas seleccionadas para data augmentation	65
Ilustración 29: Conjunto de datos generado con técnicas de augmentation en plataforma Roboflow	66
Ilustración 30: Detección de clase "uso del teléfono" con modelo entrenado con YOLOv8 con data augmentation	67
Ilustración 31: Nuevo conjunto de datos con menos técnicas de data augmentation	68
Ilustración 32: Parada temprana en entrenamiento de modelo de 200 épocas	69
Ilustración 33: Nuevo cuadro delimitador para etiqueta “desconcentrado”	70
Ilustración 34: Nuevo cuadro delimitador para etiqueta “concentrado”	70
Ilustración 35: Conjunto de datos de dos perfiles con todas algunas técnicas de data augmentation	71

Ilustración 36: Finalización del entrenamiento del conjunto de datos más completo	71
Ilustración 37: Evaluación práctica del modelo con poca augmentación y diferente etiquetado .	74
Ilustración 38: Sección principal Landing page.....	76
Ilustración 39: Sección características de la app.....	77
Ilustración 40: Sección beneficios de la app.....	77
Ilustración 41: Sección contacto	77
Ilustración 42: Asistente de concentración	78
Ilustración 43: Arquitectura del sistema	80
Ilustración 44: Reconocimiento de estado "Concentrado" en aplicativo.....	82
Ilustración 45: Respuesta de estado "Concentrado" en back-end.....	83
Ilustración 46: Reconocimiento de estado "Desconcentrado" en aplicativo	83
Ilustración 47: Respuesta de estado "Desconcentrado" en back-end.....	83
Ilustración 48: Reconocimiento de estado "Uso de teléfono" en aplicativo	84
Ilustración 49: Respuesta de estado "Uso de teléfono" en back-end.....	84

RESUMEN

En los últimos años, se observó un creciente uso de las redes sociales entre los jóvenes. Ha emergido como una actividad de ocio significativa, que conduce a una distracción en todo momento del día. Este proyecto se enfoca particularmente en las distracciones causadas por el teléfono celular en las sesiones de estudio de alumnos universitarios. Las constantes alertas de notificaciones del dispositivo celular conllevan a la revisión de las mismas todo el tiempo, esto se asemeja a un comportamiento adictivo, consumiendo e interrumpiendo la concentración del estudiante. Estos nuevos hábitos impactan en la productividad o rendimiento académico de los jóvenes estudiantes, sin lugar a duda. Este trabajo busca desarrollar una aplicación que funcione como herramienta para mejorar la concentración del estudiante en sus sesiones de estudio marcada por la técnica Pomodoro, y además, la misma podrá detectar diferentes estados de la concentración del estudiante, por lo que se hará uso de la inteligencia artificial, la cual podrá detectar, por medio de la cámara web si el individuo se encuentra concentrado, desconcentrado o usando el celular. De esta manera, se obtendrá un seguimiento de la concentración del estudiante durante sus minutos de estudio, proporcionando una retroalimentación y alertas en tiempo real. Esta información será de utilidad para el alumno de modo que regule el uso de su dispositivo celular y servirá de motivación para seguir perfeccionando su concentración y, en su consecuencia, mejorar su rendimiento académico.

1 INTRODUCCIÓN

Las tecnologías de la información y comunicación (TIC) fueron abarcando todos los aspectos de la vida, e influyen cotidianamente en las personas. Su uso, se acrecentó aún más en la pandemia del año 2020, en donde, desde niños hasta adultos, tuvieron que adaptarse al uso de las tecnologías para seguir realizando sus actividades de manera virtual. En particular, los jóvenes se han visto afectados por una distracción generada por las notificaciones de las diversas redes sociales, que se encuentran activas en sus teléfonos celulares. Estas distracciones se convirtieron en un daño a la concentración de estudiantes universitarios en sus actividades académicas, ya que pueden afectar su desempeño, dado que su productividad se ve reducida en sus sesiones de estudio (Rosa María Muñoz Franco et al., 2024).

El presente trabajo surge para responder a la necesidad de abordar este problema presente en la sociedad. Se busca desarrollar un prototipo, que funcione como una herramienta de soporte para mejorar la concentración de los estudiantes durante sus períodos de estudio, aplicando la técnica Pomodoro, además, visión artificial para el reconocimiento de los estados de concentración del estudiante frente a su computadora. Para ello, se entrenará un modelo de detección de objetos con diferentes imágenes, desde una perspectiva de la persona frente a su computadora, en estado de concentración, distracción, y usando el teléfono.

Por lo tanto, se desarrollará el marco teórico, del cual se sustenta esta aplicación de soporte, el entrenamiento del modelo de detección de estados, el aplicativo, que consumirá ese modelo, y que el usuario utilizará como herramienta de estudio, y aspectos técnicos que engloban al proyecto.

2 PRELIMINARES

2.1 DEFINICIÓN DEL PROBLEMA

Como se mencionó en la introducción de este escrito, en los últimos años, los jóvenes, desde temprana edad, ya están inmersos en el mundo de las redes sociales, mediante sus teléfonos celulares en donde están conectados con sus compañeros, amigos y familiares. Así entonces, tienen constantes notificaciones entrantes, o simplemente la proximidad con el dispositivo, hacen que pierdan la atención en cualquier momento del día, frente a otras actividades que realizan a su vez. Este proyecto, hace foco en las distracciones presentadas en los momentos de estudio de alumnos que utilizan una computadora como herramienta para leer apuntes, resúmenes, entre otros archivos digitales.

Se obtuvo, mediante la evaluación PISA 2022 (Programa para la Evaluación Internacional de Alumnos), que en Argentina, más del 50% de los estudiantes reconocieron que se distraen en clases, usando sus celulares, y, además, ellos afirman que no escuchan lo que dice el profesor en clases (La distracción con el celular y la indisciplina afectan el aprendizaje en Argentina - Infobae, 2024). Estos resultados llevan a pensar que, si en un aula los alumnos se distraen, aún más lo hacen en sus hogares, al momento de realizar sus actividades académicas, ya que no poseen ningún tipo de control.

Tanto es así, sobre esta problemática, que la UNESCO recomendó prohibir los celulares en las escuelas, ya que se sostiene que disminuye el rendimiento académico, y que producen más efectos negativos que positivos en las mismas aulas (Informe de la UNESCO recomienda prohibir el uso de celulares en las escuelas - Infobae, 2024).

Por otro lado, se afirma que, cuando los estudiantes se distraen, les lleva hasta veinte minutos volverse a enfocar en su actividad, interrumpiendo así la productividad. En países como China, que poseen las primeras posiciones de las evaluaciones PISA, se implementan restricciones al uso de teléfonos en adolescentes, y en países europeos, ya se están aplicando estas medidas para favorecer el rendimiento académico (Informe de la UNESCO recomienda prohibir el uso de celulares en las escuelas - Infobae, 2024).

Entonces, la raíz de este problema se encuentra en la constante revisión de notificaciones y la búsqueda incesante de actualizaciones en el celular, durante las sesiones de estudio de un alumno universitario que utiliza su computadora para leer archivos académicos. Este comportamiento se asemeja a una adicción digital, que disminuye la productividad de la persona, al gastar tiempo en estas acciones, en vez de realizar sus labores. De esta manera, sin darse cuenta, el alumno consume gran cantidad de su tiempo, revisando su celular, sin así poder avanzar con sus actividades, llevando así en su consecuencia, a un mal desempeño estudiantil.

Estos son los motivos que llevan a desarrollar este proyecto, para fortalecer la concentración del estudiante, al evitar el uso del celular y asegurarse así tener una buena productividad y en su consecuencia que esta impacte de manera positiva sobre su rendimiento académico.

2.2 OBJETIVOS DEL PROYECTO

2.2.1 OBJETIVO GENERAL

El objetivo principal del proyecto es el desarrollo de una aplicación web, que funcione como herramienta de soporte de estudio para el alumno universitario, que lo realiza a través de su computadora de escritorio o portátil. Se busca implementar la técnica Pomodoro, y, además, que el sistema utilice visión artificial para detectar los diferentes estados de concentración durante la sesión de estudio. Es por esto, por lo que también se requerirá el entrenamiento de un modelo de detección de objetos, con la ayuda de YOLOv8. Todo el sistema tendrá la finalidad de controlar, segundo a segundo, la concentración de la persona frente a la computadora durante los intervalos, y, como resultado final, obtener una retroalimentación del desempeño, para que pueda autorregularse en sus próximos intentos.

2.2.2 OBJETIVO ESPECÍFICO

Los objetivos particulares de este producto son:

- Entrenar un modelo de detección de objetos, siguiendo la metodología CRISP DM, por medio de la herramienta YOLOv8, que reconozca estados de concentración de una persona frente a una computadora, entre los cuales serían: concentrado, desconcentrado y uso del teléfono. Aquí se aplicarán conceptos de Inteligencia artificial del campo de *deep learning* y redes neuronales.
- Desarrollar la aplicación web, teniendo en cuenta la experiencia de usuario y el rendimiento de esta para múltiples dispositivos, desde los cuales se hará uso la misma. Además, implementar un reloj, que marque las sesiones de estudio de la técnica Pomodoro.
- Desarrollar aplicación de servidor, la cual tendrá la tarea de hacer uso del modelo pre-entrenado, para predecir los estados del estudiante, mediante imágenes enviadas desde la interfaz del usuario.

2.3 ALCANCE DEL PROYECTO

Se busca, en primera instancia, entrenar un modelo de detección de objetos, mediante la herramienta YOLOv8 y Roboflow, con imágenes de diferentes personas, mirando de frente a la pantalla de la computadora, otras usando el celular, y de perfil hacia la derecha. Se optaron por clasificar estos tres estados, para no complejizar el modelo, y que las predicciones sean lo más rápido posible, ya que, en la aplicación final, se consumirá el modelo cada segundo que la persona esté corriendo su sesión de estudio, por lo que el modelo debe de ser lo más liviano posible, para optimizar la aplicación, ya que esta sólo será una demostración.

Por otro lado, se desarrollará el back-end del proyecto, el cual servirá únicamente para consumir el modelo entrenado, al que, desde el front-end, recibirá los fragmentos del video, en imágenes, cada segundo, y realizará la predicción, devolviendo así los resultados. La separación del front-end y el back-end, es debido a que se busca mejorar el rendimiento, dado que el front-

end será una aplicación web, que se ejecutará en el navegador, y esta misma, por cada predicción, sufre de bloqueos debido al gran consumo de sus recursos por parte del modelo clasificador.

Además, como se mencionó anteriormente, se desarrollará la interfaz de usuario, como un aplicativo web, el cual enseñará los detalles, características, y beneficios, para que el usuario conozca el producto al cual va a acceder. Por otro lado, en la misma, se encontrará la aplicación con un reloj, que marque el tiempo de las sesiones, como así también gráficos, que muestran en tiempo real el comportamiento de la persona. Se tendrá en cuenta la experiencia de usuario, haciéndola intuitiva y rápida, con tecnologías de desarrollo modernas.

Resulta de necesidad aclarar que la aplicación será sólo una demostración, la cual se ejecutará de manera local¹. Se utilizará la metodología CRISP-DM, para el entrenamiento del modelo, y la metodología de desarrollo de software iterativa e incremental.

¹ Correr la aplicación directamente en el equipo del usuario, sin necesidad de utilizar un servidor o infraestructura en la nube, permitiendo probar y desarrollar el software en un entorno controlado

3 MARCO TEÓRICO

Para lograr una comprensión de la solución propuesta, se detalla a continuación conceptos fundamentales que se aplican en el desarrollo del proyecto, como ser tecnologías, metodologías, definiciones, y procedimientos técnicos. Cada uno de los elementos es el sustento teórico de la solución tecnológica a la problemática, como así también, ayudan a garantizar el cumplimiento de los objetivos obteniendo un proyecto de calidad.

3.1 Inteligencia Artificial

El campo de la inteligencia artificial no sólo intenta comprender, sino que también intenta construir entidades inteligentes. Abarca grandes subcampos, que va desde el propósito general, aprendizaje y percepción, a otras como demostraciones de teoremas matemáticos y diagnóstico de enfermedades. La IA sintetiza y automatiza tareas intelectuales, por lo que, es muy relevante para cualquier ámbito de la actividad intelectual humana. En la siguiente Ilustración 1, se presentan diferentes definiciones de inteligencia artificial, que el autor extrajo de otros. Las superiores, se refieren a procesos mentales y al razonamiento, y las de la parte inferior, a la conducta del humano. Las definiciones del lado izquierdo miden el éxito en términos de fidelidad, en cómo actúan los humanos, y las de la derecha toman como referencia el concepto de inteligencia, que es la racionalidad (Russell et al., 2004).

Sistemas que piensan como humanos	Sistemas que piensan racionalmente
«El nuevo y excitante esfuerzo de hacer que los computadores piensen... máquinas con mentes, en el más amplio sentido literal». (Haugeland, 1985)	«El estudio de las facultades mentales mediante el uso de modelos computacionales». (Charniak y McDermott, 1985)
«[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje...» (Bellman, 1978)	«El estudio de los cálculos que hacen posible percibir, razonar y actuar». (Winston, 1992)
Sistemas que actúan como humanos	Sistemas que actúan racionalmente
«El arte de desarrollar máquinas con capacidad para realizar funciones que cuando son realizadas por personas requieren de inteligencia». (Kurzweil, 1990)	«La Inteligencia Computacional es el estudio del diseño de agentes inteligentes». (Poole <i>et al.</i> , 1998)
«El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor». (Rich y Knight, 1991)	«IA... está relacionada con conductas inteligentes en artefactos». (Nilsson, 1998)

Ilustración 1: Definiciones de inteligencia artificial organizadas en cuatro categorías (Russell et al., 2004).

3.2 Minería de datos

En la actualidad, estamos sobrecargados de datos, gran cantidad de ellos alrededor del mundo crecen día a día. Este crecimiento, se debe a que todo el mundo se encuentra utilizando dispositivos electrónicos, como computadoras y celulares, y hacen que los datos se almacenen en todo momento y más fácil que nunca.

En la minería de datos, la información se almacena de forma electrónica, y la búsqueda es automatizada o al menos asistida por computadoras. Esto no es algo completamente nuevo, ya que economistas, estadísticos, pronosticadores e ingenieros de comunicación han trabajado durante mucho tiempo con la idea de que los patrones en los datos pueden ser encontrados, identificados, validados y utilizados para hacer predicciones de manera automática. Lo novedoso, es el enorme incremento en las oportunidades para descubrir patrones en los datos. El crecimiento sin control de las bases de datos en los últimos años, para actividades cotidianas como las preferencias de los clientes, ha puesto a la minería de datos en la vanguardia de las nuevas tecnologías de negocios. Se estima, que la cantidad de datos almacenados en las bases de datos del mundo se duplica cada 20 meses, y aunque justificar esta cifra de manera precisa podría ser complicado, todos podemos entender cualitativamente el rápido ritmo de crecimiento. A medida que aumenta el volumen de datos, y las máquinas capaces de realizar estas búsquedas se vuelven comunes, también aumentan las oportunidades para la minería de datos. Con el mundo generando cada vez más datos y volviéndose más complejo, la minería de datos se convierte en nuestra única esperanza para revelar patrones ocultos. Los datos analizados de manera inteligente, son un recurso muy valioso, que puede llevar a nuevos descubrimientos y, en el ámbito comercial, a ventajas competitivas (Witten Ian et al., 2011).

La minería de datos, se centra en resolver problemas mediante el análisis de grandes volúmenes de datos ya sea que provengan de datos estructurados (hojas de cálculo, bases de datos relacionales), no estructurados (imágenes, videos, audios) y semiestructurados (formato XML, JSON). Por ejemplo, para abordar la lealtad de los clientes en un mercado competitivo, se pueden analizar las elecciones, y perfiles de los clientes para identificar patrones de comportamiento. Esto permite identificar a los clientes, que probablemente cambien de producto y dirigirles tratamientos especiales. También se pueden usar estas técnicas para ofrecer nuevos servicios a clientes actuales. En la economía actual, los datos son esenciales para el crecimiento empresarial, si se pueden extraer eficientemente. La minería de datos es el proceso, generalmente semiautomático, de descubrir patrones significativos, en grandes cantidades de datos para obtener por ejemplo, ventajas económicas (Witten Ian et al., 2011).

3.2.1 Tipos de aprendizajes en minería de datos

En el campo de la minería de datos es fundamental distinguir los distintos tipos de aprendizajes que permiten a los modelos procesar y analizar grandes volúmenes de información. Estos métodos se distinguen entre aprendizaje supervisado y no supervisado, cada uno con características y aplicaciones específicas que determinan cómo el modelo adquiere y utiliza el conocimiento para hacer predicciones identificando patrones.

Aprendizaje supervisado consta de aprender en función de patrones de entradas con sus respectivas salidas. Por ejemplo, un agente aprende de reglas de condiciones para frenar un vehículo, entonces, a partir de este aprendizaje genera una salida para frenar o no frenar. En este caso, que se usarán imágenes como datos se lo podría ejemplificar de la siguiente manera: A partir de una imagen enseñamos a nuestro agente a identificar si existe o no un auto, de esta forma luego nos puede decir si en una imagen existe o no (Russell et al., 2004).

Aprendizaje no supervisado consiste en aprender por medio de patrones de entrada en los que no se identifican las salidas de estos. Por ejemplo, un taxista puede predecir, a medida que transcurre el tiempo y aumente su experiencia, los días de tráfico buenos o malos sin que tenga ejemplos etiquetados de ellos (Russell et al., 2004).

Aprendizaje por refuerzo El aprendizaje por refuerzo, es el más general de las tres categorías. En lugar de que un agente reciba instrucciones específicas de un profesor, el agente debe aprender a partir del refuerzo. Por ejemplo, la falta de propina al final de un viaje, o una gran factura por un accidente proporciona al agente indicaciones de que su comportamiento no es adecuado. El aprendizaje por refuerzo típicamente incluye el subproblema de aprender cómo se comporta el entorno (Russell et al., 2004).

Dentro del aprendizaje supervisado existen dos tipos de tareas:

Clasificación: usa un algoritmo para asignar con precisión los datos de prueba en categorías específicas. Dentro de un conjunto de datos, reconoce las entidades e intenta sacar conclusiones de cómo deben de ser etiquetadas o definidas esas entidades (¿Qué es el aprendizaje supervisado? | IBM, 2024). Las tareas de clasificación pueden ser:

- **Detección de Spam:** Un modelo clasifica correos electrónicos como "spam" o "no spam" en función de características como el contenido del mensaje, el remitente, y el asunto, basándose en ejemplos previamente etiquetados.
- **Reconocimiento de imágenes:** El modelo identifica y clasifica objetos en imágenes, como por ejemplo "perro", "gato" o "persona". Cada imagen de entrenamiento está etiquetada con la clase correspondiente para guiar el aprendizaje.
- **Clasificación de sentimientos:** El modelo puede analizar un texto de reseñas de productos o publicaciones en redes sociales para clasificarlas en "positiva" o "negativa", basándose en un conjunto de datos etiquetados previamente.

Regresión: se usa para comprender la relación entre variables dependientes e independientes. Habitualmente se la usa para hacer proyecciones como los ingresos por ventas dentro de una empresa (¿Qué es el aprendizaje supervisado? | IBM, 2024). Por ejemplo:

- **Predicción de Precios:** Un modelo puede estimar el precio de una vivienda basándose en variables como tamaño, ubicación, número de habitaciones, entre otras.
- **Pronóstico de Ventas:** En el comercio, un modelo de regresión puede predecir el número de ventas futuras en función a datos históricos de ventas, estacionalidad y campañas de marketing. La salida es un número esperado de ventas.
- **Estimación de consumo energético:** Un modelo de regresión estima el consumo de energía de un edificio en función de variables como temperatura externa, número de ocupantes y hora del día. Puede ser útil para optimizar uso de recursos.

3.2.2 Algoritmos de aprendizaje supervisado

Estos algoritmos son fundamentales para tareas de clasificación y regresión donde el modelo se entrena con un conjunto de datos etiquetados. Este tipo de aprendizaje permite que el

modelo identifique patrones y relaciones en los datos para hacer predicciones precisas en nuevos casos. Cada uno de los métodos que se nombran a continuación poseen ventajas y desventajas específicas en cuanto a precisión y velocidad.

Redes Neuronales

Las redes neuronales, utilizadas principalmente en algoritmos de deep learning, imitan la estructura del cerebro humano con capas de nodos interconectados. Cada nodo recibe entradas, aplica ponderaciones y un sesgo, y genera una salida. Si la salida supera un umbral específico, el nodo se activa, pasando datos a la siguiente capa. Las redes neuronales ajustan sus parámetros basándose en la función de pérdida utilizando el descenso de gradiente, buscando minimizar la función de costes para mejorar la precisión del modelo.

Naive Bayes

El algoritmo Naive Bayes clasifica datos, basándose en el Teorema de Bayes, asumiendo que las características son independientes entre sí. Este método se divide en tres tipos: Naive Bayes multinomial, Bernoulli y gaussiano. Es especialmente útil en la clasificación de textos, detección de spam y sistemas de recomendación, ya que considera que la presencia de una característica no influye en otra.

Regresión Lineal

La regresión lineal busca identificar la relación entre una variable dependiente y una o más variables independientes, utilizando esta relación para hacer predicciones. Puede ser simple (una variable independiente) o múltiple (varias variables independientes). La línea de mejor ajuste se calcula mediante el método de mínimos cuadrados y se representa como una línea recta en un gráfico.

Máquinas de Vectores de Soporte (SVM)

Se usan tanto para clasificación como para regresión, aunque se emplean principalmente en clasificación. El algoritmo construye un hiperplano que maximiza la distancia entre diferentes clases de puntos de datos, separando las clases en lados opuestos del plano.

K Vecino Más Cercano (KNN)

El algoritmo KNN clasifica puntos de datos basándose en su proximidad a otros datos, asumiendo que los datos similares están cerca unos de otros. Calcula la distancia entre puntos de datos (generalmente usando la distancia euclidiana) y asigna una categoría basándose en la categoría más frecuente entre sus vecinos. Es popular por su facilidad de uso y bajo tiempo de cálculo, aunque su eficiencia disminuye con conjuntos de datos grandes.

Random Forest

Random Forest es un algoritmo flexible utilizado para clasificación y regresión. Consiste en una colección de árboles de decisión no correlacionados que se combinan para reducir la varianza y mejorar la precisión de las predicciones.

3.3 Aprendizaje profundo - *Deep learning*

Deep learning, es una rama del *machine learning* que consiste en una red neuronal. Estas redes neuronales buscan imitar el funcionamiento del cerebro humano permitiéndoles "aprender" a partir de grandes volúmenes de datos. Si bien una red neuronal con una sola capa puede hacer predicciones aproximadas, las capas ocultas adicionales mejoran y perfeccionan la precisión. Es la base de muchos servicios y aplicaciones de inteligencia artificial que facilitan la automatización, ejecutando tareas analíticas y físicas sin necesidad de intervención humana. Esta tecnología sustenta numerosos productos y servicios que usamos a diario (como asistentes digitales, controles de TV activados por voz y la detección de fraudes con tarjetas de crédito), así como tecnologías emergentes (como los vehículos autónomos) (¿Qué es Deep Learning? | IBM, 2024).

Comparación entre Deep learning y Machine learning

Deep learning se diferencia del machine learning clásico en su mejor capacidad para manejar grandes volúmenes de datos no estructurados, como imágenes y textos. Los algoritmos de machine learning utilizan datos estructurados y etiquetados para hacer predicciones, lo que implica definir características específicas durante la entrada de datos y organizarlos en tablas. Esto no significa que no puedan usar datos no estructurados, pero generalmente requieren un procesamiento previo para organizarlos en un formato estructurado.

Deep learning reduce la necesidad de procesamiento previo de datos que requiere el machine learning. Estos algoritmos pueden procesar datos no estructurados, como texto e imágenes, y automatizan la extracción de características, disminuyendo la dependencia de expertos humanos. Por ejemplo, si tenemos un conjunto de fotos de diversas mascotas y queremos categorizarlas como "gato", "perro", "hámster", etc., los algoritmos de deep learning pueden identificar qué características (como las orejas) son cruciales para diferenciar a un animal de otro. En machine learning, esta jerarquía de características debe ser definida manualmente por un experto. (¿Qué es Deep Learning? | IBM, 2024).

¿Cómo funcionan las redes neuronales?

Las redes neuronales de deep learning, también conocidas como redes neuronales artificiales, buscan imitar el cerebro humano combinando entradas de datos, ponderaciones y sesgos. Estos componentes colaboran para reconocer, clasificar y describir con precisión los objetos en los datos.

Las redes neuronales profundas están compuestas por múltiples capas de nodos interconectados, cada una construida sobre la anterior para refinar y optimizar la predicción o clasificación. Esta serie de cálculos a través de la red se conoce como propagación hacia adelante.

Las capas de entrada y salida en una red neuronal profunda se llaman capas visibles. La capa de entrada es donde el modelo de deep learning recibe los datos para su procesamiento, y la capa de salida es donde se realiza la predicción o clasificación final.

Un proceso adicional llamado propagación inversa utiliza algoritmos, como el descenso de gradiente, para calcular los errores en las predicciones y ajustar las ponderaciones y sesgos de la función, moviéndose hacia atrás a través de las capas para entrenar el modelo. La propagación hacia adelante y la propagación inversa permiten a la red neuronal hacer predicciones y corregir errores de manera efectiva. Con el tiempo, el algoritmo mejora su precisión.

Lo anterior describe el tipo más simple de red neuronal profunda en términos sencillos. Sin embargo, los algoritmos de deep learning son extremadamente complejos y existen diferentes tipos de redes neuronales diseñadas para resolver problemas o manejar conjuntos de datos específicos. Por ejemplo:

- **Redes Neuronales Convolucionales (CNN):** Principalmente utilizadas en aplicaciones de clasificación de imágenes y visión por computadora, estas redes pueden detectar características y patrones en una imagen, permitiendo tareas como la detección y el reconocimiento de objetos. En 2015, una CNN superó por primera vez a un humano en un desafío de reconocimiento de objetos.
- **Redes Neuronales Recurrentes (RNN):** Comúnmente empleadas en aplicaciones de procesamiento de lenguaje natural y reconocimiento de voz, estas redes aprovechan datos de series temporales o secuenciales (¿Qué es Deep Learning? | IBM, 2024).

3.4 Detección de objetos

La detección de objetos es una técnica, que integra la clasificación de imágenes, con la localización precisa de objetos, ofreciendo una comprensión detallada de la imagen. Tradicionalmente, se utilizaban métodos de extracción manual de características junto con arquitecturas simples para la detección de objetos. Con el crecimiento de las herramientas de deep learning, se han superado numerosas limitaciones de las técnicas tradicionales, gracias a la capacidad de aprender características semánticas y profundas. Este campo ha atraído la atención de numerosos investigadores recientemente. La detección de objetos incluye dos operaciones: localización y clasificación de objetos (Aziz et al., 2020).

Clasificación de Imagen

La clasificación de imágenes es una tarea fundamental en la visión por computadora. Dado un conjunto de imágenes de entrada, el objetivo de un modelo de clasificación de imágenes es asignar cada imagen a una de un conjunto predefinido de clases. La mayoría de los modelos de clasificación de imágenes utilizan redes neuronales convolucionales (CNNs), que están diseñadas específicamente para procesar datos de píxeles y pueden capturar características espaciales. Los modelos de clasificación de imágenes se entrenan con grandes conjuntos de datos (YOLO Object Detection Explained: A Beginner's Guide | Encord, 2024).

Localización de Objetos

La localización de objetos, es otra tarea importante en la visión por computadora, que identifica la ubicación de un objeto en la imagen. Extiende el modelo de clasificación de imágenes añadiendo tarea de regresión, para predecir las coordenadas del cuadro delimitador del objeto. El cuadro delimitador se representa típicamente por cuatro coordenadas que definen su posición y tamaño. La localización de objetos, es un paso clave en la detección, donde el objetivo no es sólo clasificar el objeto principal de interés en la imagen, sino también identificar su ubicación (YOLO Object Detection Explained: A Beginner's Guide | Encord, 2024).

3.4.1 Clasificación de Algoritmos de detección de objetos

Los algoritmos pueden estar clasificados en dos categorías: un sólo disparo, y detectores de dos o más disparos. Ambos abordan la tarea de detección de diferentes maneras.

Detección de Objetos de un sólo disparo (*single shot object detectors*)

Los detectores de un sólo disparo (SSDs) son un tipo de algoritmo de detección de objetos, que predicen el cuadro delimitador y la clase del objeto en un sólo disparo. Esto significa que en una única pasada de la red, se predice simultáneamente la presencia de un objeto y el cuadro delimitador. Esto hace que los SSDs sean muy rápidos y eficientes, lo que los hace adecuados para tareas que requieren detección en tiempo real. Presentan algunas debilidades frente a otros enfoques, como ser la precisión, especialmente al detectar objetos pequeños en escenas densas. Carecen de la granularidad y detalle a comparación de métodos más complejos como Faster R-CNN, que realizan varias pasadas para refinar detecciones.

YOLO (*you only look once*) y SSD (*single shot multibox detector*) son ejemplos de algoritmos que se clasifican como de un sólo disparo. YOLO divide la imagen de entrada en una cuadrícula y, para cada celda de la cuadrícula, predice un cierto número de cuadros delimitadores y probabilidades de clase. Por otro lado, SSD predice cuadros delimitadores y probabilidades de clase en múltiples escalas en diferentes mapas de características.

Detección de objetos de dos disparos (*two shot object detectors*)

Los algoritmos de detección de objetos de dos disparos o múltiples disparos utilizan un proceso de dos pasos para detectar objetos. El primer paso, implica proponer una serie de cuadros delimitadores que podrían contener un objeto. Esto se hace a menudo utilizando un método llamado *región proposal networks RPNs*. El segundo paso consiste en pasar estas regiones propuestas a través de una red neuronal convolucional para clasificar las clases de objetos dentro del cuadro.

Ejemplos de algoritmos de detección de objetos de dos disparos incluyen R-CNN, Fast R-CNN y Faster R-CNN. Estos algoritmos utilizan redes (RPNs) para proponer cuadros

delimitadores potenciales y luego utilizan redes neuronales convolucionales para clasificar las regiones propuestas.

Tanto los detectores de un sólo disparo como los de dos disparos tienen sus fortalezas y debilidades. Los detectores de un sólo disparo son generalmente más rápidos y eficientes, lo que los hace adecuados para tareas de detección de objetos en tiempo real. Por otro lado, los detectores de dos disparos, aunque son más lentos y computacionalmente intensivos, tienden a ser más precisos, ya que pueden dedicar más recursos computacionales a cada objeto potencial.

3.4.2 Métodos de Detección de Objetos

En este apartado se describirán diferentes enfoques para detección de objetos, como el método de Viola-Jones hasta avanzadas técnicas basadas en redes neuronales profundas. A diferencia de los tradicionales, los enfoques neuronales pueden adaptarse mejor a variaciones de escala y complejidad.

Detección de Objetos: Métodos No Neuronales

Método de Detección de Objetos de Viola-Jones Basado en Características de Haar: El método Viola-Jones, introducido por Paul Viola y Michael Jones, es un modelo de aprendizaje automático para la detección de objetos, particularmente utilizado para rostros. Utiliza una cascada de clasificadores, seleccionando características de conjuntos de características tipo Haar. El algoritmo tiene cuatro etapas:

1. Selección de Características de Haar
2. Creación de una Imagen Integral
3. Entrenamiento con Adaboost
4. Clasificadores en cascada

A pesar de su simplicidad y velocidad, puede lograr altas tasas de detección. Técnicamente entonces tiene un bajo costo computacional, lo que es ideal para dispositivos de bajo rendimiento. Este método no posee un buen rendimiento en condiciones complejas, como ser variaciones de iluminación y rotación del rostro, por lo que sería demasiado impreciso en este proyecto.

Transformada de Características Invariantes a Escala (SIFT): es un método para extraer características distintivas e invariantes de las imágenes. Estas características son invariantes a la escala y rotación de la imagen, y son robustas a cambios en el punto de vista, ruido e iluminación. Las características SIFT se utilizan para coincidir diferentes vistas de un objeto o escena. A pesar de su robustez frente a variaciones de escala, rotación e iluminación, tiene un elevado costo computacional lo que no lo hace viable para aplicaciones que requieren alta velocidad de detección de objetos, existen algoritmos más eficientes y precisos que se detallan más adelante.

Histograma de Gradientes Orientados (HOG): es un descriptor de características utilizado para la detección de objetos en visión por computadora. Implica contar las ocurrencias de la orientación de los gradientes en porciones localizadas de una imagen. Este método es similar

a los histogramas de orientación de bordes, los descriptores de transformada de características invariantes a escala y los contextos de forma, pero se diferencia en que se calcula en una cuadrícula densa de celdas uniformemente espaciadas. Tiene un buen rendimiento de uso computacional, aun así, existen métodos avanzados que aprenden características más complejas.

Detección de Objetos: Métodos Neuronales

Redes Neuronales Convolucionales Basadas en Regiones (R-CNN): Las R-CNN utilizan redes neuronales convolucionales para clasificar regiones de imagen con el fin de detectar objetos. Implica entrenar una CNN en un gran conjunto de datos etiquetado y luego usar la red entrenada para detectar objetos en nuevas imágenes. Las R-CNN y sus sucesoras, Fast R-CNN y Faster R-CNN, son conocidas por su precisión, pero pueden ser computacionalmente intensivas. Dado que posee un elevado tiempo de procesamiento, dificulta su implementación en aplicaciones de tiempo real, donde la velocidad es un requisito clave.

Faster R-CNN: Faster R-CNN es una versión avanzada de R-CNN que introduce una Red de Propuesta de Regiones (RPN) para generar propuestas de regiones. La RPN comparte características convolucionales de imagen completa con la red de detección, lo que permite propuestas de regiones casi gratuitas en términos de costo. La RPN es una red completamente convolucional que predice simultáneamente los límites de los objetos y las puntuaciones de objetividad en cada posición. Faster R-CNN es más rápida que la R-CNN original y Fast R-CNN porque no necesita ejecutar un método separado de propuesta de regiones en la imagen, lo cual puede ser lento. A pesar de ser más rápida sigue siendo ineficiente para aplicaciones en tiempo real, existen mejores propuestas en cuanto a rendimiento de procesamiento se refiere.

Mask R-CNN: Mask R-CNN extiende Faster R-CNN añadiendo una rama para predecir una máscara de objeto en paralelo con la rama existente para el reconocimiento de cuadros delimitadores. Esto permite a Mask R-CNN generar máscaras de segmentación precisas para cada objeto detectado, además de la etiqueta de clase y el cuadro delimitador. La rama de máscara es una pequeña red totalmente convolucional aplicada a cada Región de Interés (RoI), prediciendo una máscara binaria para cada RoI. Mask R-CNN es simple de entrenar y añade sólo una pequeña sobrecarga computacional, permitiendo un sistema rápido y experimentación rápida. Sin embargo, este método es ideal para aplicaciones que requieren una comprensión más detallada del entorno, como la visión autónoma.

Single Shot Detector (SSD): SSD es un método de detección de objetos que elimina la necesidad de múltiples pasadas de red para múltiples escalas. Discretiza el espacio de salida de los cuadros delimitadores en un conjunto de cuadros predeterminados sobre diferentes relaciones de aspecto y escalas por ubicación de mapa de características. SSD es más rápido que métodos como R-CNN porque elimina las propuestas de cuadros delimitadores y las capas de agrupación. Es una buena propuesta para el aplicativo, aun así, existen mejores como YOLO.

RetinaNet: RetinaNet utiliza una red piramidal de características sobre una base para detectar objetos en diferentes escalas y relaciones de aspecto. Introduce una nueva pérdida, la Focal Loss, para abordar el problema del desequilibrio de clases entre el fondo y el primer plano. RetinaNet está diseñado para manejar objetos densos y pequeños. Este método es particularmente

eficaz para detectar objetos pequeños en escenarios densos, por lo cual no es un buen candidato para este proyecto, además al ser más complejo requiere más capacidad de cómputo.

EfficientDet: EfficientDet es un método que escala todas las dimensiones del ancho, profundidad y resolución de la red con un método de escalado compuesto para lograr un mejor rendimiento. Introduce una nueva arquitectura llamada BiFPN, que permite una fusión de características multiescala fácil y eficiente, y un nuevo método de escalado que escala uniformemente la resolución, profundidad y ancho de toda la red de base, la red de características y las redes de predicción de cajas/clases al mismo tiempo. EfficientDet logra precisión de vanguardia con menos parámetros y menos computación en comparación con detectores anteriores. Este método, si bien es eficiente en términos de uso de recursos puede no ofrecer la misma rapidez que YOLO para aplicaciones en tiempo real.

You Only Look Once (YOLO): YOLO, desarrollado por Joseph Redmon y colaboradores, enmarca la detección de objetos como un problema de regresión para cuadros delimitadores espacialmente separados y probabilidades de clase asociadas. Observa toda la imagen en tiempo de prueba, por lo que sus predicciones están informadas por el contexto global en la imagen. YOLO es conocido por su velocidad, lo que lo hace adecuado para aplicaciones en tiempo real. Este método tiene las características esenciales para el proyecto, tiene un balance adecuado entre precisión y rendimiento computacional.

Los métodos neuronales son mejores que los métodos no neuronales en la detección de objetos debido a su capacidad para aprender características más complejas y adaptarse a diversas condiciones de imágenes, como variaciones de iluminación, ángulos, y ruido. Mientras que los métodos no neuronales, como Viola-Jones o HOG, se basan en características predefinidas y funcionan bien en escenarios controlados, los métodos neuronales, aprovechan el poder de las redes neuronales convolucionales para extraer características relevantes de forma automática, lo que permite una mayor precisión en entornos complejos. YOLO es capaz de procesar imágenes en tiempo real, sin necesidad de segmentar o procesar la imagen en múltiples etapas, como otros métodos. Es por ello por lo que, es la opción más adecuada para el desarrollo del Asistente de Concentración.

3.5 YOLO (*You Only Look Once*)

El algoritmo YOLO emplea una única Red Neuronal Convolucional (CNN) que divide la imagen en una cuadrícula. Cada celda en la cuadrícula predice un cierto número de cuadros delimitadores. Junto con cada cuadro delimitador, la celda también predice una probabilidad de clase, que indica la probabilidad de que un objeto específico esté presente en el cuadro.

Proceso de reconocimiento de cuadros delimitadores

El proceso de reconocimiento de cuadros delimitadores en YOLO involucra los siguientes pasos:

1. **Creación de la Cuadrícula:** La imagen se divide en una cuadrícula de $S \times S$. Cada celda de la cuadrícula es responsable de predecir un objeto si el centro del objeto cae dentro de ella. Esto significa que, aunque un objeto pueda ocupar múltiples celdas, sólo la celda que contiene el centro del objeto será la encargada de realizar la predicción. De esta forma, se evita que las celdas adyacentes hagan predicciones redundantes o conflictivas sobre el mismo objeto.
2. **Predicción de Cuadros Delimitadores:** Cada celda de la cuadrícula predice B cuadros delimitadores y puntajes de confianza para esos cuadros. El puntaje de confianza refleja cuán seguro está el modelo de que un cuadro contiene un objeto y cuán precisa cree que es la predicción del cuadro.
3. **Predicción de Probabilidades de Clase:** Cada celda de la cuadrícula también predice C probabilidades de clase condicionales (una por clase para los posibles objetos). Estas probabilidades están condicionadas a la existencia de un objeto en el cuadro.

Non-Max Suppression (NMS) es un paso crucial de post-procesamiento en la detección de objetos. Una vez que se predicen las cajas delimitadoras y las probabilidades de clase, NMS reduce la redundancia eliminando las cajas que se superponen significativamente, con aquellas que tienen los puntajes de confianza más altos.

La Generalización Vectorial en el algoritmo YOLO aborda el desafío de la salida de alta dimensionalidad. Inicialmente, un tensor que abarca las coordenadas de las cajas delimitadoras, los puntajes de objetividad y las probabilidades de clase se aplanan en un vector para un procesamiento más eficiente. Posteriormente, una función softmax convierte los puntajes de clase en probabilidades, resultando en un vector por celda de cuadrícula que contiene todas las salidas de detección necesarias (YOLO Object Detection Explained: A Beginner's Guide | Encord, 2024).

3.5.1 Evolución de YOLO

La evolución de YOLO ha dado lugar a una serie de modelos, cada uno con mejoras significativas en la precisión, eficiencia y rapidez para detectar objetos.

YOLOv1: Fue el primer modelo en abordar la detección de objetos en tiempo real. Trató a la detección de objetos como un problema de regresión, dividiendo la imagen en una cuadrícula para identificar objetos y sus ubicaciones en tiempo real. Su enfoque permitió una detección precisa a gran velocidad, pero sólo para objetos grandes y en escenas simples.

YOLOv2: Introdujo técnicas para mejorar la estabilidad del modelo y detectar objetos pequeños con mucha más precisión. Este modelo fue capaz de manejar imágenes de alta resolución y ofrecer resultados más estables.

YOLOv3: Mejoró nuevamente la precisión y rapidez, implementando técnicas novedosas. Logró un mejor balance entre velocidad y precisión, haciéndola tres veces más rápida que otros modelos.

YOLOv4: Se mejoró la precisión sin incrementar el tamaño del archivo del modelo.

YOLOv5: Introdujo una serie de mejoras enfocadas en hacer el modelo mucho más ligero y eficiente logrando más precisión sin comprometer la rapidez. Esta versión es valorada por su facilidad de uso, lo que la hace una de las más populares en detección de objetos.

YOLOv6: Aumentó la precisión manteniendo el tamaño del archivo y su velocidad. Permitió que el modelo sea usado en dispositivos con limitaciones de recursos.

YOLOv7: Se implementaron más técnicas que ayudaron a la mejora de la precisión del modelo.

YOLOv8: Esta versión reúne las mejores características de sus predecesores. Tiene la capacidad de realizar diferentes tareas como detección de objetos, segmentación, estimación de poses y clasificación. Una técnica clave introducida en YOLOv8 es la detección de objetos a múltiples escalas, permitiendo al modelo detectar objetos de diferentes tamaños en una imagen.

El algoritmo YOLOv8 muestra un aumento del 1.2% en la precisión promedio en comparación con YOLOv7, lo cual representa una mejora significativa. Esto se logró mientras se reducía el tamaño del archivo del modelo, haciendo que el modelo sea más eficiente y fácil de implementar en entornos con recursos limitados.

Es la última versión de YOLO desarrollada por Ultralytics. Como un modelo de vanguardia y estado del arte (SOTA), YOLOv8 se basa en el éxito de versiones anteriores al introducir nuevas características y mejoras para mejorar el rendimiento, la flexibilidad y la eficiencia. YOLOv8 soporta una amplia gama de tareas de IA visual, incluyendo detección, segmentación, estimación de pose, seguimiento y clasificación. Esta versatilidad permite a los usuarios aprovechar las capacidades de YOLOv8 en diversas aplicaciones y dominios (YOLO Object Detection Explained: A Beginner's Guide | Encord, 2024).

3.6 CRISP-DM

Cross Industry Standard Process for Data Mining es una metodología para orientar a los trabajos de minería de datos. Como metodología, incluye descripciones de las fases de un proyecto, tareas necesarias y las relaciones entre dichas tareas, y como modelo de proceso ofrece un ciclo vital de minería de datos (Conceptos básicos de ayuda de CRISP-DM - Documentación de IBM, 2024).

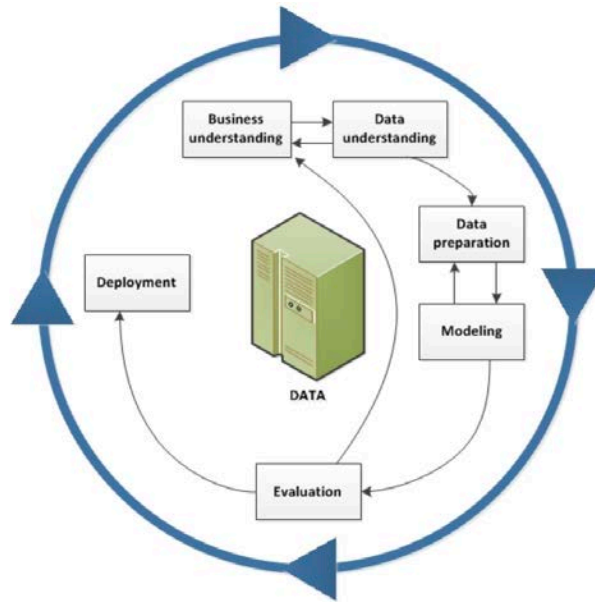


Ilustración 2: Ciclo de vida de minería de datos

Como se puede apreciar en la Ilustración 2, existen seis fases del proceso de minería de datos, estas no estrictamente van en secuencia, sino que en un proyecto se puede avanzar y retroceder sobre estas fases mientras sea necesario. Las flechas indican las relaciones más importantes entre estas fases. Las flechas en círculos de alrededor simbolizan la naturaleza cíclica de la minería de datos, esto es porque los procesos de minería de datos se benefician de experiencias anteriores (Pete Chapman (NCR) et al., 1999).

La descripción de cada ciclo de vida es la siguiente:

Comprensión del Negocio: Esta fase inicial se enfoca en entender los objetivos y requisitos del proyecto desde una perspectiva empresarial, transformando este entendimiento en una definición del problema de minería de datos y un plan preliminar para alcanzar los objetivos.

Comprensión de los Datos: Comienza con la recopilación inicial de datos y continúa con actividades para familiarizarse con ellos, identificar problemas de calidad, descubrir primeras ideas o detectar subconjuntos interesantes que formen hipótesis sobre información oculta.

Preparación de los Datos: Incluye todas las actividades necesarias para construir el conjunto de datos final a partir de los datos iniciales. Las tareas pueden realizarse varias veces y en cualquier orden, e incluyen la selección de tablas, registros y atributos, así como la transformación y limpieza de los datos.

Modelado: Se seleccionan y aplican técnicas de modelado, calibrando sus parámetros a valores óptimos. A menudo se prueban varias técnicas para el mismo problema, y algunas pueden requerir volver a la fase de preparación de los datos.

Evaluación: Se revisan y evalúan los modelos construidos para asegurar que cumplen los objetivos del negocio y detectar cualquier problema no considerado. Al final de esta fase, se decide sobre el uso de los resultados de la minería de datos.

Implementación: La creación del modelo no es el fin del proyecto. El conocimiento adquirido debe organizarse y presentarse de manera útil para el cliente, ya sea mediante informes o la implementación de procesos de minería de datos en la organización. El cliente, y no el analista de datos, suele llevar a cabo la implementación, por lo que es crucial que el cliente comprenda las acciones necesarias para utilizar los modelos creados.

3.7 Roboflow

Es una herramienta, que facilita las tareas de visión por computadora en el campo del *Deep Learning*. Permite a los desarrolladores crear sus aplicaciones de visión por computadora. Admite modelos de detección y clasificación de objetos. Esta herramienta brinda al usuario herramientas para: anotación de datos, preprocesamiento de datos, exportación de conjuntos de datos, entrenamiento de modelos, entre otras.

3.7.1 Herramienta de Anotación

Roboflow provee una rápida y robusta interfaz, a través de la cual los desarrolladores pueden etiquetar imágenes para poder entrenar modelos detectores de objetos en imágenes. Mediante la herramienta, se puede etiquetar las imágenes dibujando cuadros delimitadores manualmente, marcando así el objeto en cuestión con su respectiva clase.

3.8 Técnica Pomodoro

La Técnica Pomodoro es un sistema de organización del tiempo que busca aumentar la productividad, mediante la división de las tareas en bloques específicos. Desarrollada por Francesco Cirillo, este método consiste en trabajar durante intervalos de 25 minutos, llamados "Pomodoros", tras los cuales se toma un descanso corto de 5 minutos. Después de completar cuatro ciclos de trabajo, se realiza una pausa más larga, que puede variar entre 15 a 30 minutos. El propósito de este enfoque, es optimizar la concentración y reducir las distracciones, promoviendo un flujo de trabajo constante. Además, es ideal para mantener un equilibrio entre el esfuerzo y el descanso, lo que contribuye a mejorar el enfoque y conservar la energía mental (Cirillo & Info, 2020).

3.9 Desarrollo Incremental

El desarrollo incremental es una metodología que consiste en crear una implementación inicial, presentarla al usuario para obtener su retroalimentación, y luego mejorarla a través de múltiples versiones hasta lograr un sistema completo y funcional. A diferencia de los modelos de desarrollo en cascada tradicionales, donde las fases de especificación, desarrollo y validación se llevan a cabo de forma separada y secuencial, el desarrollo incremental integra estas actividades y facilita una retroalimentación continua a lo largo del proceso (Sommerville et al., 2011).

Este enfoque se basa en la premisa de que es raro prever y resolver completamente un problema de software desde el inicio. En su lugar, se progresa gradualmente hacia una solución, corrigiendo errores a medida que se presentan. El desarrollo incremental es más económico y flexible, ya que permite adaptarse a cambios en los requisitos del cliente durante el desarrollo del software.

3.10 Node.js

Node.js es un entorno de ejecución de JavaScript basado en eventos asincrónicos, diseñado para crear aplicaciones de red escalables. Node.js ejecuta el motor JavaScript V8, el núcleo de Google Chrome, fuera del navegador, lo que le permite ser muy eficiente.

Una aplicación Node.js se ejecuta en un sólo proceso, sin crear un nuevo hilo para cada solicitud. Node.js proporciona un conjunto de primitivas de E/S asincrónicas en su biblioteca estándar que evitan que el código JavaScript se bloquee.

Cuando Node.js realiza una operación de E/S, como leer desde la red, acceder a una base de datos o al sistema de archivos, en lugar de bloquear el hilo y desperdiciar ciclos de CPU esperando, Node.js reanuda las operaciones cuando llega la respuesta.

Esto permite a Node.js manejar miles de conexiones concurrentes con un sólo servidor sin la carga de gestionar la concurrencia de hilos, lo cual podría ser una fuente significativa de errores (Node.js — About Node.js®, 2024).

3.11 React.js

React es una biblioteca de JavaScript utilizada para construir interfaces de usuario interactivas. Por interfaces de usuario (UI), se refiere a los elementos que los usuarios ven e interactúan en la pantalla.

Parte del éxito de React radica en que no impone estrictamente cómo deben abordarse los otros aspectos del desarrollo de aplicaciones. Esto ha generado un ecosistema próspero de herramientas y soluciones de terceros, incluyendo Next.js.

Sin embargo, esto también significa que construir una aplicación completa con React desde cero requiere cierto esfuerzo. Los desarrolladores deben dedicar tiempo a configurar herramientas y a crear soluciones para requisitos comunes de aplicaciones (React Foundations: About React and Next.js | Next.js, 2024).

3.12 Next.js

Next.js es un marco flexible basado en React, que proporciona las bases necesarias para crear aplicaciones web.

Al construir aplicaciones modernas, hay varios aspectos a considerar, tales como:

- Interfaz de Usuario: cómo los usuarios interactuarán y consumirán la aplicación.
- Enrutamiento: cómo los usuarios navegarán entre las diferentes partes de la aplicación.
- Obtención de Datos: dónde residen los datos y cómo se acceden a ellos.
- Renderizado: cuándo y dónde se renderiza contenido estático o dinámico.
- Integraciones: qué servicios de terceros utiliza (para CMS, autenticación, pagos, etc.) y cómo se conecta a ellos.
- Infraestructura: dónde se despliega, almacena y ejecuta el código de tu aplicación.
- Rendimiento: cómo optimizar la aplicación para los usuarios finales.
- Escalabilidad: cómo se adapta la aplicación a medida que crecen el equipo, datos y tráfico.
- Experiencia del Desarrollador: la experiencia del equipo al construir y mantener la aplicación.

Para cada parte la aplicación, se debería de decidir si se construye una solución por sí mismo o usar otras herramientas, como paquetes, bibliotecas y frameworks.

Next.js es un framework de React que proporciona los componentes necesarios para crear aplicaciones web.

Al decir framework, hace referencia a que Next.js se encarga de las herramientas y configuraciones necesarias para React, y ofrece estructura adicional, características y optimizaciones para desarrollar una aplicación.

Se usa React para construir tu interfaz de usuario y, luego, adoptar gradualmente las características de Next.js para resolver requisitos comunes de aplicaciones, como el enrutamiento, la obtención de datos y el almacenamiento en caché (React Foundations: About React and Next.js | Next.js, 2024).

3.13 Nest.js

Nest.js es un framework para construir aplicaciones del lado del servidor en Node.js de manera eficiente y escalable. Utiliza JavaScript progresivo, está construido con TypeScript y lo soporta completamente (aunque también permite a los desarrolladores codificar en JavaScript puro) y combina elementos de POO (Programación Orientada a Objetos), PF (Programación Funcional) y PRF (Programación Reactiva Funcional).

Nest proporciona un nivel de abstracción por encima de estos frameworks comunes de Node.js, pero también expone sus APIs directamente al desarrollador. Esto ofrece a los desarrolladores la libertad de utilizar la multitud de módulos de terceros disponibles para la plataforma subyacente (Documentation | NestJS - A progressive Node.js framework, 2024).

3.14 ONNX (*Open Neural Network Exchange*)

Microsoft y un grupo de colaboradores, desarrollaron ONNX como un estándar abierto para representar modelos de aprendizaje automático. Se pueden exportar o convertir modelos de varios marcos como TensorFlow, PyTorch, scikit-learn, Keras, Chainer, MXNet y MATLAB al formato estándar ONNX. Una vez en formato ONNX, estos modelos pueden ejecutarse en múltiples plataformas y dispositivos.

ONNX Runtime es un motor de inferencia de alto rendimiento, diseñado para implementar modelos ONNX en entornos de producción. Está optimizado para su uso tanto en la nube como en Edge, y es compatible con sistemas operativos como Linux, Windows y Mac. Aunque ONNX Runtime está desarrollado en C++, también ofrece API en C, Python, C#, Java y JavaScript (Node.js), lo que lo hace adaptable a diversos entornos de desarrollo. Este motor de ejecución soporta tanto redes neuronales profundas (DNN) como modelos de aprendizaje automático tradicionales, integrándose con aceleradores de hardware como TensorRT en GPU Nvidia, OpenVINO en procesadores Intel y DirectML en entornos Windows. Al utilizar ONNX Runtime, los usuarios pueden aprovechar amplias optimizaciones, pruebas y mejoras destinadas a entornos de producción.

ONNX Runtime se utiliza en servicios a gran escala de Microsoft, como Bing, Office y Azure AI. Aunque los aumentos de rendimiento varían según distintos factores, estos servicios de Microsoft han reportado un aumento promedio de hasta el doble en la CPU (Modelos de ONNX: optimización de la inferencia - Azure Machine Learning | Microsoft Learn, 2024) .

4 SOLUCIÓN PROPUESTA

4.1 SOLUCIÓN AL PROBLEMA

En primer lugar, se realizó una investigación que afirme que los estudiantes sufren de distracciones en momentos de estudios, por la consecuencia de los teléfonos y las redes sociales. Es así como se confirmó, con varios artículos periodísticos y trabajos de investigación, que afirmativamente los estudiantes de nivel secundario y universitario tienden a distraerse constantemente con sus celulares en diferentes momentos de su día, perdiendo productividad y atención a sus labores como académicos (Garate-Osuna et al., 2015).

Posteriormente, se pensó en cómo se puede disminuir el impacto del problema usando inteligencia artificial. En esta etapa, se redujo la solución de la problemática a alumnos universitarios, que generalmente estudian con sus computadoras, desde archivos de textos digitales, como ser PDFs, apuntes de clases en editores de textos, resúmenes online, entre otros. Este nuevo enfoque permitió así, imaginar una herramienta que utilice la webcam de la computadora, para poder ver de frente al alumno, y mediante visión artificial detectar si el sujeto se encuentra mirando fijo la pantalla, estado que consideramos que acertadamente está concentrado, o de otra manera que se encuentre mirando de perfil hacia los costados y un último caso, usando su teléfono celular que serían alarmas que muestran símbolos de distracción del estudiante.

Teniendo en cuenta esta solución a alto nivel, se abordó la tarea de explorar tecnologías de visión artificial que permitan llevar a cabo el desarrollo del producto. Se consideraron diferentes alternativas, pero se descubrió una en particular, que se la considera estado del arte, que es el algoritmo YOLO el cual es el más avanzado, robusto y veloz para detectar objetos en tiempo real (YOLO Object Detection Explained: A Beginner's Guide | Encord, 2024).

Durante la fase de investigación de tecnología, se llevaron a cabo diferentes pruebas para validar la factibilidad técnica de la solución entrenando diferentes modelos de detección de estados de concentración de una persona frente a una computadora. Se llevaron a cabo implementaciones para un producto final, sorteando obstáculos de compatibilidad y rendimiento entre diferentes tecnologías y entornos de desarrollo.

4.2 METODOLOGÍA

En este proyecto se utilizarán dos metodologías, una para trabajos de minería de datos denominada CRISP-DM y, por otro lado, para el desarrollo del software, la metodología incremental o desarrollo iterativo.

La metodología de desarrollo iterativo o incremental se caracteriza por proponer un enfoque en el que el trabajo se divide en porciones pequeñas incrementales y se van desarrollando en las diferentes iteraciones a lo largo de la planificación. En cada uno de los ciclos, se realiza el ciclo completo que consta del diseño, análisis, desarrollo y pruebas. Esta metodología ayuda a tener una mayor flexibilidad y adaptabilidad frente a los cambios del proyecto, ya que permite

realizar ajustes y mejoras continuas en cada iteración. Por otro lado, el desarrollo iterativo favorece la entrega temprana de funcionalidades del producto y la retroalimentación del cliente.

Esta metodología tiene las siguientes ventajas frente al presente desarrollo:

- **Flexibilidad y adaptabilidad:** Al ser un proyecto de gran complejidad, en el cual se utilizarán diferentes tecnologías tanto como para entrenar el modelo, desarrollar el *front-end* y *back-end*, entonces implica que a lo largo del proyecto pueden existir múltiples modificaciones o pruebas por medio de diferentes herramientas lo que es crucial en este proyecto tener la posibilidad de adaptarse fácilmente.
- **Detección temprana de errores:** Ya que en cada iteración se entregará una porción de software funcional, la metodología permitirá analizar y encontrar errores dentro del producto, que se deberán mitigar cuanto antes.
- **Entrega gradual de funcionalidades:** Con esta metodología es posible entregar en cada iteración funcionalidades del software, lo que significa ver el avance del producto con el transcurso de la planificación. De esta manera existirá una evaluación de la calidad del sistema.

Crisp-DM (*cross industry standard process for data mining*) es una metodología utilizada para orientar trabajos de minería de datos, la cual define diferentes fases en las cuales existen tareas específicas de cada una. La elección de esta metodología para el proyecto ofrece varias ventajas, al estructurarse en fases iterativas, permite la creación de un modelo robusto y adaptable en donde en cada etapa se ajusta y refina según los requisitos del proyecto. Permite regresar a etapas previas por si los resultados del modelo no cumplen con el rendimiento esperado. La flexibilidad que proporciona es útil para trabajar con modelos de detección de objetos, ya que la calidad de las imágenes y su etiquetado pueden requerir ajustes constantes para mejorar el desempeño y precisión del modelo. A continuación, se hará una breve descripción sobre cómo se aplicará CRISP-DM en el presente proyecto.

Comprensión del Negocio: Se plantearán los objetivos y requisitos del asistente de concentración a nivel de negocio. A partir de los requisitos establecidos se definirá el tipo de modelo necesario para resolver el problema con minería de datos y el plan a seguir para obtener un resultado acorde a la solución esperada.

Comprensión de los Datos: En esta etapa se recopilarán los datos, en este proyecto son imágenes con los diferentes estados a etiquetar (persona concentrada, desconcentrada y usando el celular). Se tendrá en cuenta la calidad de las imágenes y de qué manera se recopilará los datos de manera masiva para entrenar nuestro modelo.

Preparación de los Datos: Las imágenes tomadas de diferentes personas se dividirán en diferentes conjuntos y se eliminarán aquellas imágenes de mala calidad o que proporcione algún tipo de error al modelo.

Modelado: Se realizarán entrenamientos de modelos con el algoritmo de YOLOv5 y YOLOv8 para poder comparar la capacidad que tiene cada uno. Si es necesario y se observa que el rendimiento del modelo es pobre a la hora de clasificar se podrá volver a la fase de preparación de datos.

Evaluación: Se hará una comparación de los modelos entrenados y se elegirá el más eficiente para cumplir los objetivos de negocio. De ser necesario se volverá a la fase de modelado.

Implementación: Una vez que el modelo haya sido seleccionado como el adecuado para el proyecto, se lo implementará en el producto final.

4.3 PLANIFICACIÓN

4.3.1 Estructura de Desglose del Trabajo (EDT)

La Estructura de Desglose del Trabajo (EDT), es una herramienta en la gestión de proyectos que permite descomponer el trabajo en entregables y, tareas pequeñas y manejables. Ayuda a visualizar y organizar todas las actividades necesarias para cumplir con los objetivos del proyecto, distribuyendo el trabajo de manera estructurada y jerárquica.

El EDT se utiliza para garantizar que todo el trabajo definido en el alcance esté claramente identificado, lo que facilita la planificación. Es por lo que, antes de presentar el diagrama de planificación Gantt se desglosará a continuación el proyecto en paquetes de trabajo.

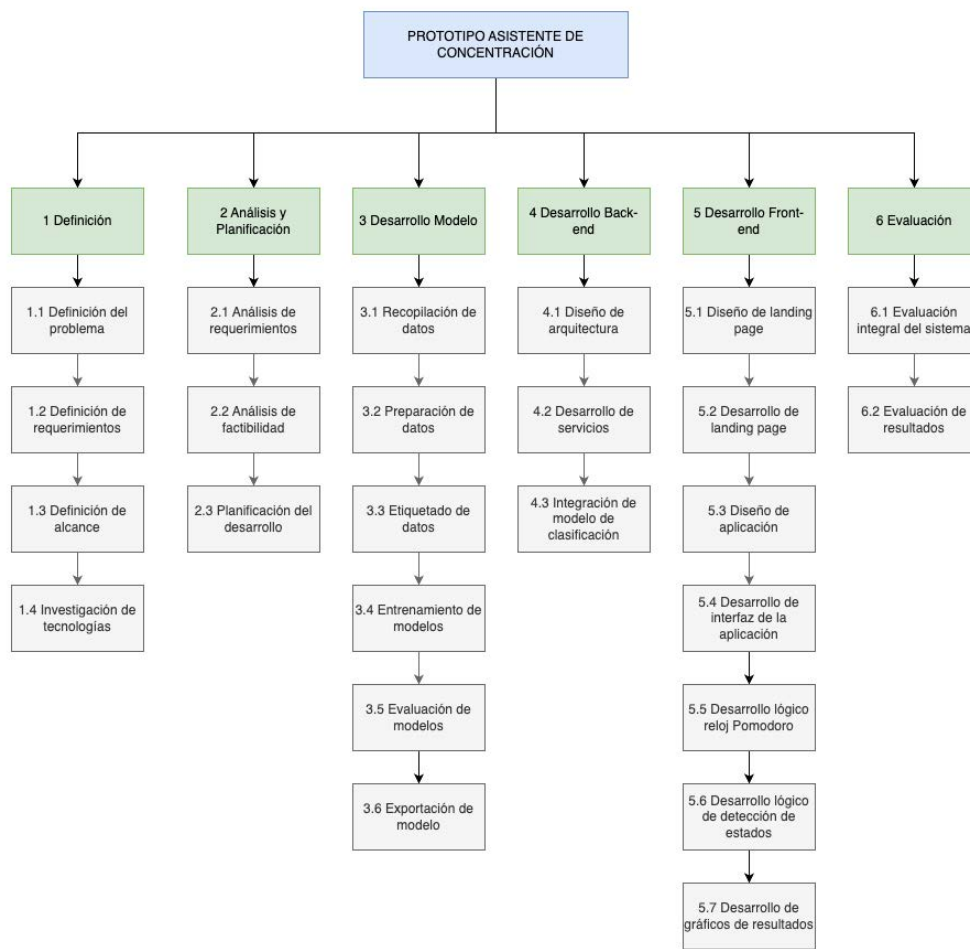


Ilustración 3: Estructura de Desglose de Trabajo

Para obtener una mejor comprensión de la Estructura de Desglose de Trabajo, se presenta el diccionario, el cual define detalladamente cada una de los entregables y tareas del proyecto.

Nivel	Nombre	Descripción
1	Definición	Definición completa del problema, alcance y requerimientos del proyecto. Además, se incluye la investigación de tecnologías.
2	Análisis y Planificación	Planificación y análisis del proyecto junto a estudio de factibilidades
3	Desarrollo Modelo	Proceso de desarrollo y entrenamiento del modelo de detección de estados
4	Desarrollo Back-end	Proceso de desarrollo de servidor con la integración del modelo detector de objetos
5	Desarrollo Front-end	Proceso de diseño, desarrollo e implementación de lógica de interfaz de usuario.
6	Evaluación	Evaluación integral del sistema y sus resultados
1.1	Definición del problema	Definición completa del problema a solucionar.
1.2	Definición de requerimientos	Definición de los requerimientos del proyecto
1.3	Definición de alcance	Definición del alcance del prototipo a desarrollar
1.4	Investigación de tecnologías	Investigación sobre tecnologías relacionadas con la detección de objetos mediante visión artificial
2.1	Análisis de requerimientos	Elaboración de listado de necesidades o requerimientos a desarrollar del proyecto
2.2	Análisis de factibilidad	Análisis de factibilidades, financiera, tecnológica, legal, entre otras.
2.3	Planificación del desarrollo	Elaboración del cronograma de trabajo para alcanzar los objetivos.
3.1	Recopilación de datos	Captura de datos para el entrenamiento del modelo detector de objetos
3.2	Preparación de datos	Segmentación y limpieza de datos
3.3	Etiquetado de datos	Etiquetado de datos para su posterior entrenamiento
3.4	Entrenamiento de modelos	Proceso de entrenamiento de modelos detectores de objetos
3.5	Evaluación de modelos	Comparación y evaluación de modelos
3.6	Exportación de modelo	Exportación de modelo en formato compatible con las tecnologías front-end y back-end
4.1	Diseño de arquitectura	Desarrollo de diagrama del diseño de arquitectura a implementar para la construcción del prototipo
4.2	Desarrollo de servicios	Desarrollo de servicio que utilice el modelo entrenado para detectar estado de concentración a partir de una imagen.
4.3	Integración de modelo de clasificación	Consumo del modelo entrenado para detectar objetos
5.1	Diseño de Landing page	Desarrollo del diseño de Landing page de la aplicación web
5.2	Desarrollo de Landing page	Desarrollo del diseño de la interfaz de Landing page
5.3	Diseño de aplicación	Desarrollo del diseño del aplicativo web del asistente de concentración
5.4	Desarrollo de interfaz de aplicación	Desarrollo a nivel código de la interfaz del aplicativo web
5.5	Desarrollo lógico del reloj Pomodoro	Desarrollo de lógica de simulación de técnica Pomodoro sobre interfaz de usuario
5.6	Desarrollo lógico de detección de estados	Implementación de servicio back-end para la detección de estados de concentración
5.7	Desarrollo de gráficos de resultados	Integración de gráficos que muestran resultados de los estados de concentración durante la sesión.

6.1	Evaluación integral del sistema	Análisis y evaluación del sistema funcionando en conjunto
6.2	Evaluación de resultados	Evaluación de resultados de rendimientos esperados del aplicativo

Tabla 1: Diccionario EDT

4.3.2 Diagrama Gantt

La gestión del tiempo en el proyecto se llevará a cabo mediante un diagrama de Gantt, la cual es una herramienta de gestión de proyectos que muestra las actividades a realizar durante un período de tiempo en relación con el estipulado para el trabajo. Gráficamente se puede observar en barras horizontales a las tareas, su duración y el cronograma de estas. En el eje horizontal está representado el tiempo y en cada tarea se indica la duración temporal.

Resulta de gran importancia mencionar que, el alcance de este trabajo solamente abarca el desarrollo de un prototipo funcional estando a cargo del autor, la planificación, análisis FODA, factibilidades, entre otros análisis se realizarán sin tener en cuenta la implementación y la inversión que requiere poner a punto este proyecto.

A continuación, se presenta el diagrama de Gantt, que se construye sobre la base del EDT.

ASISTENTE DE CONCENTRACIÓN IMPULSADO POR INTELIGENCIA ARTIFICIAL

Nombre Tarea	Fecha Inicio	Fecha Fin	Duración en Días	Rol
Definición del problema	2024-06-01	2024-06-04	3	Líder de proyecto
Definición de requerimientos	2024-06-04	2024-06-08	4	Líder de proyecto
Definición del alcance del proyecto	2024-06-08	2024-06-11	3	Líder de proyecto
Definición posibles soluciones tecnológicas	2024-06-11	2024-06-14	3	Analista funcional
Investigación de tecnologías para entrenar modelos	2024-06-14	2024-06-18	4	Analista funcional
Investigación de tecnologías front-end y back-end	2024-06-18	2024-06-22	4	Analista funcional
Análisis de requerimientos	2024-06-22	2024-06-26	4	Analista funcional
Análisis factibilidad Económica	2024-06-26	2024-06-29	3	Analista funcional
Análisis factibilidad Técnica	2024-06-29	2024-07-02	3	Analista funcional
Análisis factibilidad Legal	2024-07-02	2024-07-05	3	Analista funcional
Análisis de factibilidad Temporal	2024-07-05	2024-07-08	3	Analista funcional
Análisis de factibilidad de Mercado	2024-07-08	2024-07-11	3	Analista funcional
Planificación de desarrollo	2024-07-11	2024-07-15	4	Analista funcional
Recopilación de datos	2024-07-15	2024-07-19	4	Programador AI
Preparación de datos	2024-07-19	2024-07-22	3	Programador AI
Etiquetado de datos	2024-07-22	2024-07-24	2	Programador AI
Entrenamiento de modelos	2024-07-24	2024-07-28	4	Programador AI
Evaluación de modelos	2024-07-28	2024-08-01	4	Programador AI
Exportación de modelo	2024-08-01	2024-08-04	3	Programador AI
Diseño de arquitectura del software	2024-08-04	2024-08-08	4	Programador Full-stack
Desarrollo de back-end	2024-08-08	2024-08-11	3	Programador Full-stack
Implementación del modelo clasificador en back-end	2024-08-11	2024-08-15	4	Programador Full-stack
Prueba de servicios de backend	2024-08-15	2024-08-18	3	Programador Full-stack
Diseño de Landing Page	2024-08-18	2024-08-21	3	Programador Full-stack
Desarrollo de Landing Page	2024-08-21	2024-08-24	3	Programador Full-stack
Diseño de Aplicación	2024-08-24	2024-08-27	3	Programador Full-stack
Desarrollo Interfaz de Aplicación	2024-08-27	2024-08-31	4	Programador Full-stack
Implementación lógica reloj Pomodoro	2024-08-31	2024-09-03	3	Programador Full-stack
Implementación lógica de resultados	2024-09-03	2024-09-06	3	Programador Full-stack
Evaluación integral del sistema	2024-09-06	2024-09-10	4	Analista funcional
Evaluación de resultados	2024-09-10	2024-09-13	3	Líder de proyecto
		Total	104	

Ilustración 4 Planificación del proyecto



Ilustración 5 Diagrama de Gantt

4.3.3 RECURSOS

Las herramientas necesarias para desarrollar el producto propuesto, en materia de hardware son:

- Computadora de escritorio o portátil para el desarrollo del software tanto del entrenamiento del modelo y la aplicación web.
- Internet.
- Webcam incorporada en portátil o externa para computadora de escritorio

Las herramientas software necesarias para el desarrollo son las siguientes:

- Navegador web como ser Google Chrome para utilizar Google Colaboratoy, Roboflow, entre otras herramientas / aplicaciones online.
- Visual Studio como editor de código para desarrollar el front-end y back-end
- Node.js instalado en computadora para poder ejecutar las aplicaciones front-end y back-end.
- GitHub para manejar el versionado del código para ambos proyectos: front-end y back-end, que conforman al sistema.
- Sistema operativo MacOS, Windows o Linux.

- Python instalado para realizar pruebas de los modelos, también librerías del lenguaje como ser OpenCV y PyTorch

4.3.4 ANÁLISIS FODA

El análisis FODA permite a los proyectos o planes de negocio identificar las fortalezas, oportunidades, debilidades y amenazas en general. Con esta herramienta se puede planificar estratégicamente y adecuarse a las tendencias o comportamientos del mercado. Es crucial examinar cada uno de estos factores para planificar adecuadamente el crecimiento de las organizaciones. Al considerar todos estos aspectos en conjunto, el marco FODA puede proporcionar una visión más completa de la situación actual.

Como características internas de una entidad a analizar se encuentran las fortalezas y debilidades como, por ejemplo: recursos, capacidades, habilidades, ventajas competitivas y limitaciones. Se evalúan para identificar en qué áreas la entidad sobresale y en cuáles necesita mejorar. Como externas se clasifican las oportunidades y amenazas que pueden influir en el éxito o fracaso del proyecto. Las oportunidades son situaciones favorables que se pueden aprovechar, mientras que las amenazas son factores negativos o retos que representan riesgos para el negocio.

A continuación, se desarrolla el análisis FODA del Asistente de Concentración impulsado por Inteligencia Artificial para estudiantes Universitarios:

Fortalezas:

- **Innovación tecnológica:** El uso de inteligencia artificial para monitorear la concentración y proporcionar retroalimentación en tiempo real es una característica innovadora que puede atraer a los estudiantes.
- **Compatibilidad con la técnica Pomodoro:** La integración con una técnica de gestión del tiempo conocida por los estudiantes puede aumentar la aceptación y efectividad de la aplicación.
- **Feedback en tiempo real:** La capacidad de proporcionar alertas y retroalimentación inmediata puede ayudar a los estudiantes a corregir su comportamiento rápidamente.

Oportunidades:

- **Creciente preocupación por la salud mental y el bienestar:** La aplicación puede alinearse con iniciativas y programas que buscan mejorar la salud mental y el bienestar de los estudiantes.
- **Expansión a otros grupos de usuarios:** Aunque el enfoque inicial es en estudiantes universitarios, la herramienta puede ser útil para otros grupos que necesitan mejorar su concentración, como profesionales que realizan su labor de manera remota, universidades que toman exámenes de manera online.
- **Colaboraciones con instituciones educativas:** Las universidades y colegios pueden estar interesados en adoptar la aplicación como parte de sus recursos de apoyo a los estudiantes.

- **Mercado en crecimiento para aplicaciones educativas:** La demanda de herramientas tecnológicas que mejoren el aprendizaje y la productividad siempre se encuentra vigente, lo que puede facilitar la entrada y expansión en el mercado.
- **Enfoque en una necesidad actual:** Aborda un problema relevante y creciente entre los jóvenes, el uso excesivo de redes sociales y su impacto en el rendimiento académico.

Debilidades:

- **Dependencia de la tecnología:** Requiere que los estudiantes posean una computadora con cámara web o notebook con características de hardware suficientes para utilizar la herramienta, como así también conexión a internet.
- **Dependencia legal:** Se debe hacer un relevamiento y análisis de las legislaciones relacionadas a la privacidad y seguridad respecto del uso de la visión artificial en los países a implementar.
- **Complejidad en el desarrollo:** Entrenar un modelo que sea lo suficientemente robusto para identificar los estados de concentración puede ser una tarea difícil.

Amenazas:

- **Competencia en el mercado:** Existen muchas aplicaciones y herramientas de productividad en el mercado, lo que puede dificultar la diferenciación y adopción de la aplicación.
- **Cuestiones de privacidad y seguridad:** El uso de cámaras web y la recopilación de datos pueden plantear preocupaciones sobre la privacidad y la seguridad de los usuarios.
- **Regulaciones y políticas:** Cambios en las regulaciones sobre el uso de datos y tecnologías de inteligencia artificial pueden afectar el funcionamiento y la implementación de la aplicación.
- **Falta de adopción:** La efectividad de la aplicación depende de que los estudiantes estén dispuestos a cambiar sus hábitos y a utilizar de manera constante la herramienta.

4.3.4.1 Estrategias FODA

Estrategias FO (Fortalezas + Oportunidades)

- Aprovechar la innovación y auge tecnológico de la inteligencia artificial, y la integración de la técnica Pomodoro, para atraer el interés de instituciones educativas, estableciendo colaboraciones que promuevan el uso de la aplicación, que resuelven problemáticas dentro de los estudiantes.
- Utilizar el feedback en tiempo real como un diferenciador clave del producto, además de la integración de la inteligencia artificial, para expandirse hacia otros grupos de usuarios, como profesionales que realizan teletrabajo.

Estrategias FA (Fortalezas + Amenazas)

- Utilizar la implementación de la inteligencia artificial como una ventaja competitiva frente a la competencia en el mercado, destacando la IA y su efectividad en el reconocimiento en tiempo real de los estados de concentración.
- Los datos no son recopilados, la retroalimentación es inmediata, y de esta forma se demuestra transparencia y seguridad. Por lo que, se abordan las cuestiones de privacidad y seguridad, generando confianza en los usuarios.

Estrategias DO (Debilidades + Oportunidades)

- Superar la dependencia de la tecnología colaborando con universidades y otras instituciones, que pueden facilitar a los alumnos el acceso a recursos tecnológicos, como laboratorios de informática. Además, en la actualidad, las instituciones educativas tienen presente que la tecnología puede mejorar el aprendizaje de los estudiantes.
- La expansión a otros grupos de usuarios, como profesionales que realizan teletrabajo, puede atraer recursos y talentos, que ayuden a superar la complejidad en el desarrollo de un modelo detector de distracciones más robusto.

Estrategias DA (Debilidades + Amenazas)

- Realizar un análisis periódico de las regulaciones y políticas relacionadas con la privacidad de datos para evitar sanciones y minimizar los riesgos legales, dada la dependencia legal del proyecto.
- Mitigar la dependencia de la tecnología optimizando la aplicación para dispositivos de menor rendimiento y ampliando la compatibilidad tecnológica para evitar la falta de adopción por parte de los estudiantes.

4.3.5 MATRIZ DE PROBABILIDAD IMPACTO

La matriz de probabilidad e impacto es una herramienta utilizada en la gestión de riesgos para evaluar y priorizar los riesgos potenciales que pueden afectar a un proyecto. Esta matriz permite clasificar los riesgos en función de la probabilidad de que ocurran y el impacto que tendrían en el proyecto si ocurren. La combinación de estos dos factores ayuda a los gestores de proyectos a identificar cuáles riesgos requieren una atención y mitigación inmediata y cuáles pueden ser monitoreados con menor urgencia.

En la matriz, la probabilidad de ocurrencia de un riesgo, se clasifica en diferentes niveles, que pueden ir desde "raro" hasta "casi seguro". Por otro lado, el impacto se evalúa en términos de la severidad de sus consecuencias, desde "despreciable" hasta "catastrófico". Al cruzar estos dos factores en una matriz, se obtiene una representación visual que facilita la toma de decisiones informadas sobre la gestión de riesgos.

Consecuencia/ Probabilidad	Raro	Poco probable	Probable	Muy probable	Casi seguro
Despreciable	Interrupciones menores en el desarrollo de la aplicación: Sin impacto significativo.	Rechazo inicial por parte de algunos usuarios: Impacto insignificante en el éxito del proyecto.			
Menores	Problemas técnicos menores: Requieren pequeñas correcciones.	Preocupaciones menores de privacidad: Algunos usuarios pueden desinstalar la aplicación.	Adopción lenta de la aplicación: Impacto moderado en la expansión inicial.		
Moderadas	Errores en la detección de concentración: Requiere revisiones de IA y ajustes significativos.	Resistencia de algunas instituciones educativas: Dificulta la implementación en ciertos lugares.	Cuestiones de privacidad más serias: Podría llevar a críticas y menor adopción.		
Mayores	Fallas importantes en la IA: Requiere rediseño del sistema.	Competencia agresiva en el mercado: Podría reducir significativamente la cuota de mercado.	Problemas de integración con dispositivos: Afecta la funcionalidad en varios sistemas operativos.	Cambio en regulaciones de privacidad: Impacto grave en la continuidad del proyecto.	
Catastróficas	Violaciones graves de seguridad: Compromete la información del usuario, puede llevar a demandas legales.				

Tabla 2: Matriz de probabilidad e impacto

Descripción de los Riesgos Evaluados en la Matriz

1. Interrupciones menores en el desarrollo de la aplicación (Raro - Despreciable): Problemas técnicos menores que no impactan significativamente el progreso del desarrollo.
2. Rechazo inicial por parte de algunos usuarios (Poco probable - Despreciable): Algunos estudiantes pueden no adoptar la aplicación inicialmente, pero el impacto en el éxito general del proyecto es insignificante.
3. Problemas técnicos menores (Raro - Menores): Pequeños fallos que pueden ser corregidos con ajustes menores en el código.
4. Preocupaciones menores de privacidad (Poco probable - Menores): Algunos usuarios pueden dejar de utilizar la aplicación debido a preocupaciones de privacidad, pero no afectará de manera significativa a la mayoría.

5. Adopción lenta de la aplicación (Probable - Menores): La aplicación puede tener una adopción inicial más lenta de lo esperado, afectando la expansión en las primeras etapas.
6. Errores en la detección de concentración (Raro - Moderadas): Fallos en la precisión de la IA que requieren revisiones y ajustes significativos.
7. Resistencia de algunas instituciones educativas (Poco probable - Moderadas): Algunas instituciones pueden resistirse a implementar la aplicación, dificultando su incorporación en ciertos mercados.
8. Cuestiones de privacidad más serias (Probable - Moderadas): Preocupaciones de privacidad que podrían llevar a críticas y una menor adopción generalizada.
9. Fallas importantes en la IA (Raro - Mayores): Fallos significativos en la inteligencia artificial que requieren un rediseño completo del sistema.
10. Competencia agresiva en el mercado (Poco probable - Mayores): La presencia de competidores grandes podría reducir significativamente la cuota de mercado de la aplicación.
11. Problemas de integración con dispositivos (Probable - Mayores): Dificultades para que la aplicación funcione correctamente en varios navegadores, afectando su funcionalidad.
12. Cambio en regulaciones de privacidad (Muy probable - Mayores): Cambios en las leyes de privacidad que podrían impactar gravemente la viabilidad del proyecto.
13. Violaciones graves de seguridad (Raro - Catastróficas): Compromisos significativos de seguridad que pueden resultar en demandas legales y daños a la reputación del proyecto.

La matriz de probabilidad e impacto es esencial para la planificación estratégica del proyecto, ya que permite identificar y priorizar los riesgos, facilitando la implementación de medidas de mitigación efectivas y asegurando el éxito y sostenibilidad del proyecto.

4.3.6 PLAN DE CONTINGENCIA

Un plan de contingencia es un conjunto de procedimientos y acciones predefinidas que se implementan en respuesta a eventos o riesgos que puedan afectar negativamente un proyecto. Su objetivo es minimizar el impacto de estos eventos y asegurar la continuidad y éxito del proyecto. El plan de contingencia se basa en la identificación y evaluación de riesgos potenciales, estableciendo estrategias claras para abordarlos de manera efectiva y oportuna.

A continuación, se presenta el plan de contingencia basado en los riesgos evaluados en la matriz de probabilidad e impacto del proyecto.

1. Interrupciones menores en el desarrollo de la aplicación (Raro - Despreciable)
 - Acción preventiva: Realizar pruebas frecuentes y mantener un seguimiento de calidad del desarrollo.
 - Plan de contingencia: Corregir cualquier problema técnico menor que surja durante el desarrollo.
2. Rechazo inicial por parte de algunos usuarios (Poco probable - Despreciable)

- Acción preventiva: Realizar encuestas y pruebas beta para recoger feedback de los usuarios potenciales.
- Plan de contingencia: Crear tutoriales que resalten los beneficios de la aplicación para aumentar la aceptación.

3. Problemas técnicos menores (Raro - Menores)

- Acción preventiva: Mantener un ciclo de desarrollo iterativo e incremental con revisiones constantes en cada iteración.
- Plan de contingencia: Establecer un protocolo de corrección rápida y pruebas adicionales para asegurar que los problemas técnicos se solucionen rápidamente.

4. Preocupaciones menores de privacidad (Poco probable - Menores)

- Acción preventiva: Implementar políticas claras de privacidad y seguridad desde el inicio.
- Plan de contingencia: Proporcionar información transparente a los usuarios sobre cómo se manejan sus datos y ofrecer opciones de configuración de privacidad.

5. Adopción lenta de la aplicación (Probable - Menores)

- Acción preventiva: Desarrollar una estrategia de marketing y colaborar con instituciones educativas.
- Plan de contingencia: Ajustar la estrategia de marketing y mejorar las características de la aplicación basándose en el feedback de los usuarios iniciales.

6. Errores en la detección de concentración (Raro - Moderadas)

- Acción preventiva: Probar la precisión del sistema de inteligencia artificial exhaustivamente antes del lanzamiento.
- Plan de contingencia: Implementar un sistema de actualizaciones frecuentes para mejorar la precisión de la detección de concentración.

7. Resistencia de algunas instituciones educativas (Poco probable - Moderadas)

- Acción preventiva: Involucrar a las instituciones educativas en el desarrollo y pruebas de la aplicación.
- Plan de contingencia: Desarrollar estudios de caso y testimonios de instituciones que han adoptado con éxito la aplicación para convencer a otras.

8. Cuestiones de privacidad más serias (Probable - Moderadas)

- Acción preventiva: Cumplir con todas las normativas de privacidad y realizar auditorías de seguridad periódicas.
- Plan de contingencia: Revisar y actualizar las políticas de privacidad y seguridad, y comunicar claramente cualquier cambio a los usuarios.

9. Fallas importantes en la IA (Raro - Mayores)

- Acción preventiva: Realizar pruebas exhaustivas del modelo clasificador.
- Plan de contingencia: Desarrollar un plan de emergencia para rediseñar y optimizar el sistema de IA en caso de fallos significativos.

10. Competencia agresiva en el mercado (Poco probable - Mayores)

- Acción preventiva: Monitorear constantemente el mercado y mejorar continuamente la aplicación.
- Plan de contingencia: Desarrollar características únicas y enfocarse en la diferenciación para mantener una ventaja competitiva.

11. Problemas de integración con dispositivos (Probable - Mayores)

- Acción preventiva: Probar la aplicación en diferentes navegadores y sistemas operativos
- Plan de contingencia: Resolver problemas de integración y lanzar actualizaciones rápidas.

12. Cambio en regulaciones de privacidad (Muy probable - Mayores)

- Acción preventiva: Mantenerse actualizado con las normativas de privacidad y ajustar las políticas de la aplicación en consecuencia.
- Plan de contingencia: Implementar un equipo legal para responder rápidamente a los cambios regulatorios y asegurar el cumplimiento continuo.

13. Violaciones graves de seguridad (Raro - Catastróficas)

- Acción preventiva: Implementar medidas de seguridad avanzadas y realizar auditorías de seguridad regularmente.
- Plan de contingencia: Tener un plan de respuesta a incidentes bien definido, incluyendo la notificación a los usuarios afectados y la implementación de soluciones para prevenir futuras violaciones.

Este plan de contingencia asegura que el proyecto esté preparado para enfrentar los riesgos identificados, minimizando su impacto y garantizando una implementación y operación exitosa de la aplicación.

4.3.7 ESTUDIO DE FACTIBILIDAD

Un estudio de factibilidad es un análisis exhaustivo que evalúa la viabilidad de un proyecto antes de su implementación. Este estudio examina diversos aspectos del proyecto para determinar si es práctico y viable desde diferentes perspectivas. El objetivo es identificar cualquier obstáculo potencial y proporcionar una base sólida para la toma de decisiones informadas. Un estudio de factibilidad generalmente abarca análisis económicos, técnicos, legales, temporales y de mercado, asegurando que todos los factores relevantes se consideren antes de avanzar.

Las factibilidades se van a desarrollar con la suposición de que un equipo de expertos estará llevando a cabo el proyecto dentro para una institución universitaria y no a cargo del autor de este escrito.

4.3.7.1 Factibilidad Económica

La factibilidad económica evalúa si el proyecto es financieramente viable. Se consideran los costos de desarrollo y operación teniendo en cuenta que no se requieren licencias en la instancia de desarrollo y sólo se deben tener en cuenta costos relacionados a recursos humanos. Para este análisis, se calculan los costos en función de los días que cada perfil trabajará en el desarrollo del prototipo, de acuerdo con el cronograma de la planificación. El prototipo será desarrollado por el autor, es por lo que, los valores y roles asignados son simbólicos.

Descripción	Precio	Días/Horas	Subtotal
Programador <i>Full-Stack</i>	\$3.813 AR\$/ hora	33 días / 264hs	\$1.006.632 AR\$
Programador AI	\$4.500 AR\$/ hora	20 días / 160hs	\$720.000 AR\$
Analista funcional	\$5.720 AR\$/ hora	38 días / 304hs	\$1.738.880 AR\$
Líder de proyecto	\$8.580 AR\$/ hora	13 días / 104hs	\$892.320 AR\$
Total			\$4.294.832 AR\$

Tabla 3: Costo de desarrollo del prototipo por perfiles y cantidad de horas

Los valores utilizados fueron calculados en base a los honorarios mínimos recomendados por el sitio web del COPAIPA² al 12 de septiembre de 2024.

Inversión para despliegue

Para la implementación del sistema, se requiere de una gran inversión destinada a infraestructura tecnológica que asegure la operatividad de la aplicación. Los costos fueron calculados en base a precios de la fecha 25 de octubre de 2024.

En primer lugar, se necesita adquirir un dominio web para establecer la presencia y posicionamiento del aplicativo en internet. Este dominio puede adquirirse a través de NicAr³, cuyo precio es de \$8.500 pesos de renovación anual.

Para el alojamiento, dado que el front-end está desarrollado en Next.js, la opción de alojamiento ideal es Vercel⁴, que ofrece una infraestructura optimizada para aplicaciones desarrolladas con esta tecnología, proporciona integración continua (CI/CD) y escalabilidad automática. El costo de alojar una aplicación en esta plataforma es de 20 dólares por mes.

Para el back-end, que incorpora un modelo de inteligencia artificial, se requiere una mayor capacidad de cómputo y almacenamiento debido al procesamiento de datos. En este caso, Digital

² <https://copaipa.org.ar/informatica/>

³ <https://nic.ar/>

⁴ <https://vercel.com/>

Ocean⁵ ofrece los GPU Droplets, que es un producto especializado para ejecutar modelos de machine learning. La versión básica tiene un costo a partir de \$2,50 dólares por GPU/hora bajo contrato de 12 meses.

Este esquema de infraestructura totaliza una estimación de costo anual de \$8.500 pesos para el dominio web, más el total anual de unos \$22.140 dólares para mantener operativo el aplicativo.

Recuperación de la Inversión

Es necesario planificar cómo se recuperará la inversión inicial del proyecto. A continuación, se presentan algunas estrategias posibles para generar ingresos que permitan cubrir los costos de desarrollo, así como también, asegurar la sostenibilidad financiera del proyecto:

Venta de Licencias: Se podría comercializar la aplicación a través de licencias dirigidas a instituciones educativas, como universidades o colegios, que deseen integrar esta herramienta como un recurso para mejorar la concentración y productividad de los estudiantes. Por ejemplo, una licencia anual podría tener un costo de \$20.000 AR\$ por usuario, lo que permitiría recuperar la inversión en recursos humanos en aproximadamente 215 licencias vendidas.

Aportes de Universidades u Organizaciones: Otra opción, es la búsqueda de financiamiento a través de universidades o instituciones, que estén interesadas en el uso de la aplicación como parte de sus programas de apoyo a los estudiantes. Las universidades podrían aportar una subvención o financiamiento para el desarrollo y mantenimiento de la herramienta.

Publicidad: Se podría monetizar la aplicación a través de publicidad dentro de la plataforma. También, podría considerarse la opción de incluir patrocinios de empresas tecnológicas o instituciones interesadas en promover soluciones basadas en inteligencia artificial para la educación.

Demostración gratuita: Se puede ofrecer una versión básica y gratuita del aplicativo para atraer usuarios, y una versión completa. La versión gratuita podría ofrecer: tiempo de uso limitado al día, bloqueo de una sola aplicación, sin personalización de intervalos de tiempo, gráfico simple del tiempo de concentración. Por otro lado, la versión de pago contaría con: análisis avanzado de los tiempos de concentración, bloqueo de aplicaciones específicas para evitar notificaciones, intervalos de tiempo personalizados, consejos para mejorar la concentración, historial de sesiones y uso ilimitado del asistente. El costo de la versión completa podría compensar la inversión inicial a medida que aumente la cantidad de usuarios con plan de pago.

⁵ <https://www.digitalocean.com/>

4.3.7.2 Factibilidad Técnica

La factibilidad técnica examina si el proyecto puede ser desarrollado y operado utilizando la tecnología disponible y el conocimiento técnico existente. Este análisis es crucial para determinar la viabilidad del proyecto desde una perspectiva técnica.

Se puede afirmar que el desarrollo de este proyecto es altamente factible, ya que existen diversas tecnologías de inteligencia artificial que pueden detectar objetos mediante modelos entrenados. Estas tecnologías han avanzado significativamente en los últimos años, permitiendo una detección precisa y eficiente. Además, existen numerosos frameworks para el desarrollo de aplicaciones web, lo que ofrece una amplia variedad de opciones para la implementación. Estos frameworks proporcionan herramientas y bibliotecas que facilitan el desarrollo, reduciendo el tiempo y esfuerzo necesarios para construir aplicaciones robustas y escalables.

En términos de recursos humanos, hay una gran disponibilidad de programadores full stack e ingenieros de inteligencia artificial en todo el mundo. El trabajo remoto se ha convertido en una práctica común en la sociedad actual, y muchas grandes empresas lo consideran de vital importancia. Esta tendencia facilita la contratación de talentos altamente calificados, independientemente de su ubicación geográfica, asegurando que se pueda reunir un equipo de desarrollo competente.

Además, existen servidores y plataformas en la nube que permiten el procesamiento de datos y el almacenamiento necesarios para ejecutar los diversos servicios que requiere la aplicación. Estas soluciones en la nube ofrecen escalabilidad y flexibilidad, permitiendo al proyecto adaptarse a las necesidades cambiantes y manejar grandes volúmenes de datos de manera eficiente.

En conclusión, la factibilidad técnica del proyecto es alta, siempre y cuando se cuente con un equipo de desarrollo competente y se asegure una infraestructura adecuada. Es esencial planificar y probar exhaustivamente la integración de tecnologías avanzadas para garantizar que todas las partes del sistema funcionen de manera coherente y eficiente.

4.3.7.3 Factibilidad Legal

La factibilidad legal analiza si el proyecto cumple con todas las regulaciones y leyes aplicables dentro del país a implementar. El proyecto del asistente de concentración es viable de manera legal siempre y cuando no se almacenen datos, en concreto las de imágenes que son el dato más sensible. El aplicativo no almacena datos personales ni propios de las estadísticas, únicamente procesa las imágenes en tiempo de ejecución, sin persistencia. Esto asegura y permite que el aplicativo cumpla con regulaciones de privacidad al no guardar datos sensibles, sino que se limita a un procesamiento temporal.

Para asegurar la conformidad legal y transparencia con los usuarios, se deberían de elaborar términos y condiciones con el objetivo de establecer reglas claras del uso de la aplicación y definir limitaciones de responsabilidad. Además del documento de términos y condiciones es necesario

elaborar las políticas de privacidad con el propósito de informar a los usuarios cómo la aplicación maneja, utiliza y protege cualquier información procesada durante su uso. Se debe detallar el manejo temporal de los datos, aclarando que no se guarda ni distribuye cualquier información sensible y que los datos son de uso exclusivo en tiempo real y no deja registros en servidores o base de datos. La aceptación de términos y condiciones en una aplicación funciona como un contrato entre ambas partes, en donde expresen el consentimiento mutuo con ciertas obligaciones y derechos específicos. Este contrato protege la privacidad y confidencialidad de los datos temporales manejados en la aplicación.

Entonces la factibilidad legal es viable siempre y cuando se conformen y acepten los términos y condiciones estableciendo límites claros. La consulta permanente con asesores legales es crucial para evitar problemas legales futuros.

4.3.7.4 Factibilidad Temporal

La factibilidad temporal considera si el proyecto puede completarse dentro de un plazo razonable. La estimación del tiempo necesario para cada fase del proyecto es la adecuada teniendo en cuenta la duración de otros proyectos similares. La disponibilidad y asignación del personal es el adecuado y suficiente para cumplir con los diferentes hitos y plazos de la planificación,

- Recursos Humanos: Disponibilidad y asignación de personal adecuado para cumplir con los plazos.
- Gestión de Proyectos: Uso de metodologías de desarrollo de software que aseguran un progreso continuo y cumplimiento de objetivos en tiempo y forma.

La factibilidad de tiempo es alcanzable si se establece un cronograma realista y se gestionan adecuadamente los recursos.

4.3.7.5 Factibilidad de Mercado

La factibilidad de mercado evalúa si existe una demanda suficiente para el producto.

- Demanda del Producto: Evaluación de la necesidad del mercado mediante encuestas y estudios de mercado.
- Competencia: Análisis de aplicaciones similares y su desempeño en el mercado.
- Estrategia de Marketing: Plan para alcanzar y atraer a los usuarios objetivo, incluyendo estudiantes e instituciones educativas.

Esta factibilidad es prometedora si se confirma una demanda significativa y se diferencia la aplicación de la competencia. Una estrategia de marketing efectiva y un análisis de mercado detallado son esenciales.

El análisis a alto nivel de las diversas factibilidades muestra que el proyecto de la aplicación para mejorar la concentración de los estudiantes universitarios es viable desde las perspectivas económicas, técnicas, legales, temporales y de mercado. Sin embargo, es fundamental llevar a cabo una planificación detallada y una gestión eficiente en cada área para asegurar el éxito del

proyecto. Las recomendaciones incluyen la realización de proyecciones financieras precisas, la contratación de personal calificado, la consulta con asesores legales, el establecimiento de un cronograma realista y la implementación de una buena estrategia de marketing.

5 DESARROLLO DEL PROYECTO

5.1 Introducción

El desarrollo del Asistente de Concentración está compuesto de dos etapas: el desarrollo del modelo y el desarrollo de la aplicación web. En este apartado, se documenta el proceso de desarrollo de ambas etapas, describiendo problemáticas encontradas y soluciones implementadas. Es importante destacar que en cada etapa se emplearon las metodologías correspondientes, desarrollo incremental (para la web) y CRISP-DM (para el modelo).

5.2 Etapa 1: Desarrollo del modelo

En la primera etapa del proyecto, se inició el desarrollo del modelo utilizando la librería⁶ YOLO (You Only Look Once). El modelo será entrenado con imágenes capturadas a través de un script⁷ en Python⁸, empleando la librería OpenCV⁹ para la rápida obtención de imágenes. Inicialmente, se capturarán imágenes propias, seguidas por imágenes de otra persona para diversificar el conjunto de datos.

Para el etiquetado de las imágenes, se utilizará la herramienta de etiquetado de Roboflow, que además permitirá realizar *data augmentation* en el conjunto de datos, mejorando así la robustez¹⁰ y variabilidad del modelo.

Siguiendo la metodología CRISP-DM, se desarrollarán y evaluarán distintos modelos a lo largo del proceso. Se comenzará con la comprensión del negocio y de los datos, además de la preparación. Posteriormente de entrenar los modelos, se evaluará el rendimiento en función de la cantidad de épocas¹¹ y se analizará el impacto del uso intensivo o limitado de técnicas de *data augmentation*.

En cada iteración del desarrollo se evaluarán las métricas como precisión¹² y *recall*¹³, entre otras y además se desarrollará un *script* en Python para evaluar de manera práctica el funcionamiento del modelo. Esto servirá para identificar mejoras y optimizar el rendimiento del

⁶ Una librería en programación es un conjunto de funciones y recursos reutilizables que permiten a los desarrolladores realizar tareas comunes sin tener que escribir el código desde cero.

⁷ Término usado en programación para hablar de los fragmentos de código usados para dar forma a herramientas.

⁸ Es un lenguaje de programación versátil, conocido por su legibilidad y amplia aplicabilidad.

⁹ Es una biblioteca de visión por computadora de código abierto que facilita el procesamiento de imágenes y videos, y permite desarrollar aplicaciones que reconocen y procesan objetos en imágenes y videos.

¹⁰ La robustez de un modelo es su capacidad para mantener un buen rendimiento en condiciones variadas y datos no vistos

¹¹ Iteraciones completas en las que un modelo de aprendizaje automático procesa todo el conjunto de datos durante el entrenamiento.

¹² Proporción de verdaderos positivos entre todos los resultados positivos predichos, indicando la exactitud del modelo en sus predicciones positivas.

¹³ Proporción de verdaderos positivos detectados entre todos los casos que eran realmente positivos, midiendo la capacidad del modelo para identificar correctamente todos los casos positivos.

Asistente de Concentración hasta obtener un modelo lo suficientemente robusto y útil para poder realizar la demostración del aplicativo web.

5.2.1 Comprensión del negocio

El asistente de concentración tiene como requisito principal la detección de los estados de concentración de un estudiante, “concentrado” o “desconcentrado” y, además, la presencia de un teléfono denominado como “uso del teléfono”. Estos estados deberán de ser reconocidos cuando el usuario está utilizando su computadora de frente, es por eso, que el ambiente del cual el modelo identificará estos patrones será calmo y sin movimientos. El aplicativo en donde se busca implementar, requiere que cada segundo se clasifique una imagen, por lo que se espera encontrar un modelo que tenga un balance entre velocidad y precisión.

5.2.2 Comprensión de los datos

En esta fase, se lleva a cabo la recolección y análisis inicial de los datos. Esta creación del conjunto de datos es una etapa crucial en el desarrollo del Asistente de Concentración, ya que la calidad y variedad de los datos de entrenamiento impactan directamente en el rendimiento del modelo. Se capturarán imágenes utilizando un programa en Python junto con la librería OpenCV, lo que permitirá una obtención rápida de las imágenes necesarias para su comprensión. El proceso comenzará con la captura de imágenes propias, seguidas por la inclusión de imágenes de otra persona. Cabe aclarar, que el ángulo desde el cuál se tomarán estas imágenes es de la persona de frente.

El algoritmo captura imágenes utilizando la cámara web en relación de aspecto cuadrado, y además, permite elegir la cantidad de imágenes que el usuario desea tomar por cada etiqueta. Las imágenes son guardadas en una carpeta específica, con un identificador único agregando además el nombre de la etiqueta para diferenciar las imágenes y a posterior analizarlas. Es útil mencionar que la relación de aspecto cuadrado se debe, a que los algoritmos como YOLO hacen un ajuste de la imagen generalmente a un tamaño de 640x640 píxeles, el cuál es comúnmente utilizado en el entrenamiento de modelos de imágenes debido a su balance entre resolución y eficiencia computacional. Este proceso facilita la creación de un conjunto de datos de imágenes, adecuado para el entrenamiento del modelo.

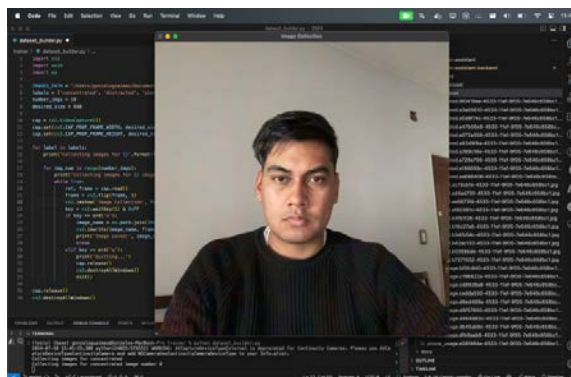


Ilustración 6: Proceso de recopilación de imágenes para crear conjunto de datos “Estado concentrado”

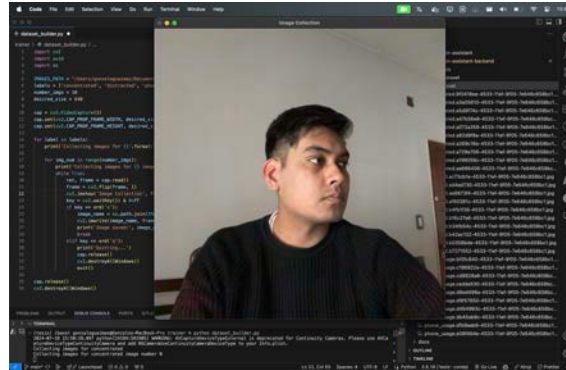


Ilustración 7: Ejemplo de captura de imagen "Estado desconcentrado"



Ilustración 8: Ejemplo de captura de imagen "Estado uso del teléfono"

Según la metodología CRISP-DM, en la etapa de comprensión de los datos, los mismos deben de adaptarse a las necesidades del proyecto, es por eso que se describirá cuál será el criterio de etiquetado de los mismos. Debido a que el proyecto implica desarrollar un prototipo, para su simplificación definiremos los estados a clasificar de manera sencilla para que el modelo pueda identificar estos sin problemas. La complejidad se encuentra cuando se hace la pregunta ¿Cuál es el estado desconcentrado de un estudiante?, las respuestas pueden ser miles, es por eso por lo que se busca definir los estados para formar un conjunto de datos por medio del cual el modelo logre clasificar de manera adecuada:

- Estado concentrado – “*concentrated*”: Rostro de persona mirando de frente con los ojos abiertos, Ilustración 6.
- Estado desconcentrado – “*distracted*”: Persona de perfil mirando hacia el lado derecho, solamente hacia esta dirección, Ilustración 7.
- Estado uso del teléfono – “*phone_usage*”: Teléfono celular en mano derecha de la persona, Ilustración 8.

La metodología establece que se deben de verificar y analizar los datos recolectados, es por ello por lo que se verificó que cada imagen represente lo que se busca etiquetar en cada una. Satisfactoriamente cada nombre de archivo representaba la clase a etiquetar en la imagen, por

ejemplo, si el nombre del archivo contenía “*phone_usage*” entonces en la imagen se observaba un teléfono celular.

5.2.3 Preparación de los datos

En la preparación de los datos, según CRISP-DM se selecciona, limpia y genera un conjunto de datos. Hasta el momento se recopilaban imágenes, pero aún no son útiles para entrenar un modelo detector de objetos, ya que estos necesitan un archivo específico para saber qué clase y en qué zona se encuentra el objeto en cada imagen. Al terminar esta tarea ya se podría decir que se posee un conjunto de datos listo para ser usado en la fase del modelado. Luego de recopilar las imágenes, se procede al etiquetado¹⁴ utilizando la herramienta de Roboflow, que no solo facilita el proceso, sino que también ofrece capacidades avanzadas de *data augmentation*, es decir generar datos adicionales a partir de los datos existentes mediante transformaciones como rotaciones, recortes, para mejorar la variabilidad y robustez del modelo. Estas técnicas de *augmentation* permitirán ampliar el conjunto de datos y mejorar su variabilidad, incrementando así la robustez del modelo.

Cada imagen será etiquetada de manera manual para garantizar que el modelo reciba información clara y útil durante el entrenamiento, a pesar de que esta herramienta ofrece un auto-etiquetado con inteligencia artificial.

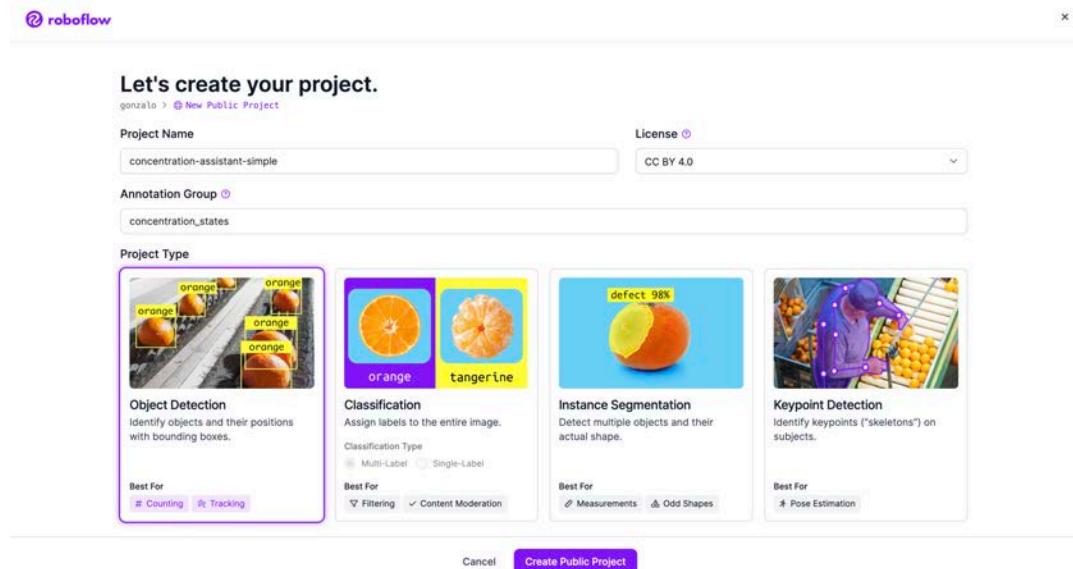


Ilustración 9: Creación de proyecto en plataforma Roboflow

Como se observa en la Ilustración 9, la creación del proyecto implica que el conjunto de datos es configurado como público en la plataforma Roboflow. Esto permite que cualquier usuario de la plataforma pueda acceder, visualizar y utilizar las imágenes y etiquetas incluidas en el conjunto de datos. Esta configuración promueve la transparencia y colaboración en la comunidad

¹⁴ La técnica del etiquetado en aprendizaje automático consiste en asignar etiquetas o categorías a los datos para entrenar modelos de manera supervisada.

científica y de desarrolladores, facilitando la replicación de experimentos y el desarrollo de nuevas aplicaciones basadas en el mismo conjunto de datos.

Además, la disponibilidad pública del conjunto de datos contribuye a la validación y mejora continua del modelo, ya que otros investigadores y desarrolladores pueden proporcionar retroalimentación y sugerencias basadas en sus propios experimentos y análisis. Esta práctica es especialmente valiosa en el campo de la inteligencia artificial y el aprendizaje automático, donde la colaboración y el acceso abierto a los datos pueden acelerar significativamente el progreso y la innovación.

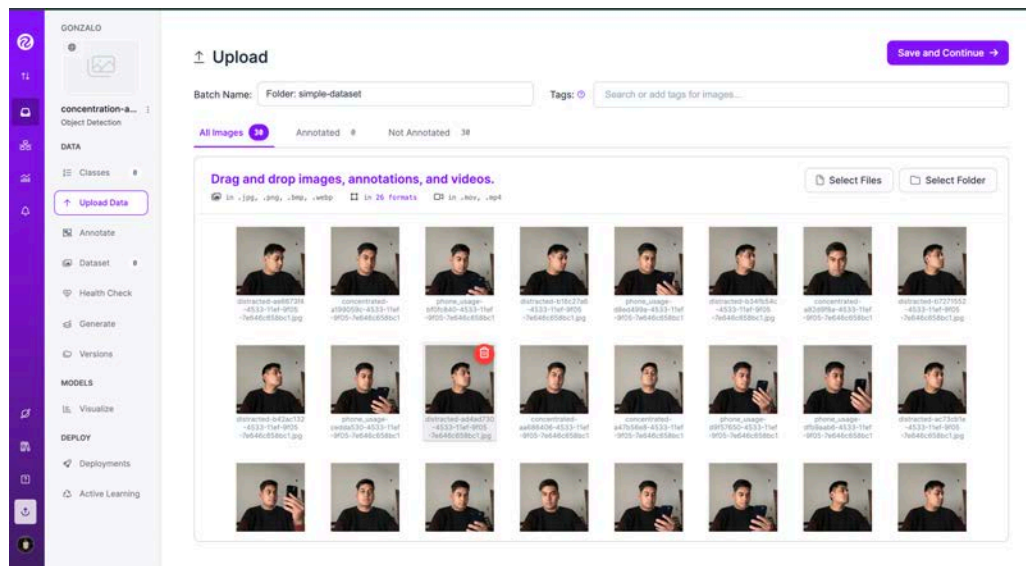


Ilustración 10: Importación del conjunto de datos en la plataforma Roboflow

Una vez importado el conjunto de datos en la plataforma Roboflow desde los archivos de la computadora, se procede a etiquetar manualmente cada imagen dibujando un cuadrado alrededor del objeto en cuestión. En este primer entrenamiento, la identificación del estado concentrado y desconcentrado sólo se tomarán en cuenta los ojos de la persona y no todo el rostro (boca, nariz y orejas) como se observa en las ilustraciones

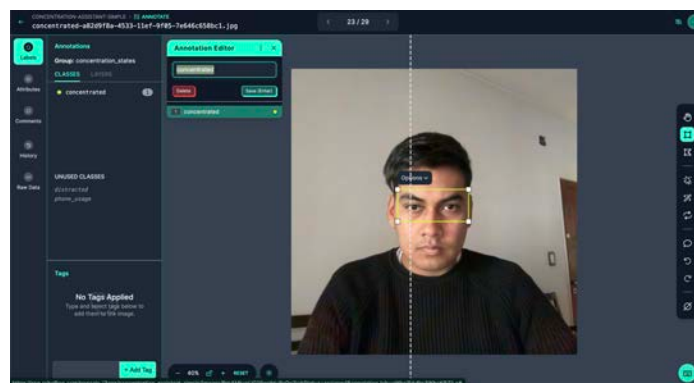


Ilustración 11: Etiquetado estado "concentrado" en plataforma Roboflow

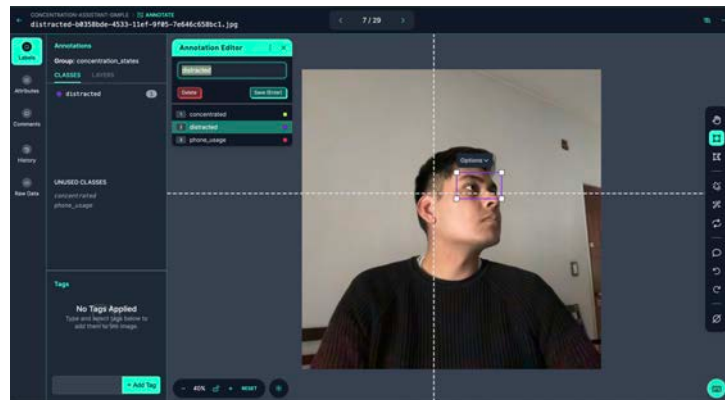


Ilustración 12: Etiquetado estado "desconcentrado" en plataforma Roboflow

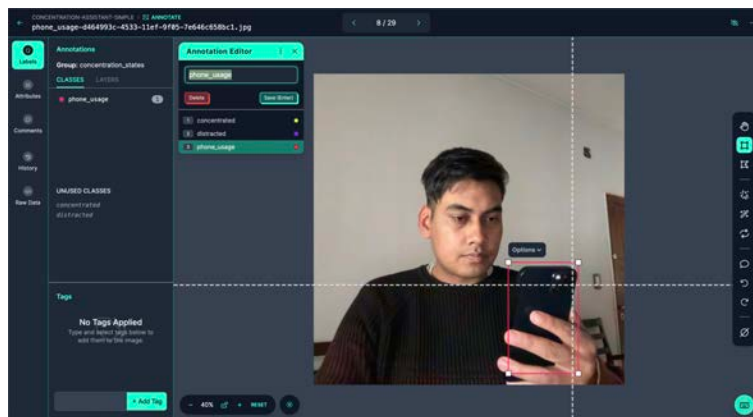


Ilustración 13: Etiquetado estado "uso del teléfono" en plataforma Roboflow

Una vez realizado el etiquetado a cada una de las imágenes que conforman el conjunto de datos, se procede a dividir los datos para entrenamiento, validación y prueba. Esta división es esencial para garantizar que el modelo se entrene adecuadamente y se evalúe de manera precisa. Los datos de entrenamiento se utilizan para ajustar los parámetros del modelo, los datos de validación ayudan a prevenir el sobreajuste¹⁵, y los datos de prueba permiten evaluar el rendimiento final del modelo en datos no vistos, asegurando su capacidad de generalización a nuevas situaciones. Roboflow al ser una herramienta completa permite hacer esta división en su plataforma.

¹⁵ También conocido como *overfitting*: Ocurre cuando un modelo aprende demasiado bien los detalles y el ruido del conjunto de datos de entrenamiento, lo que disminuye su capacidad de generalización y rendimiento en datos nuevos.

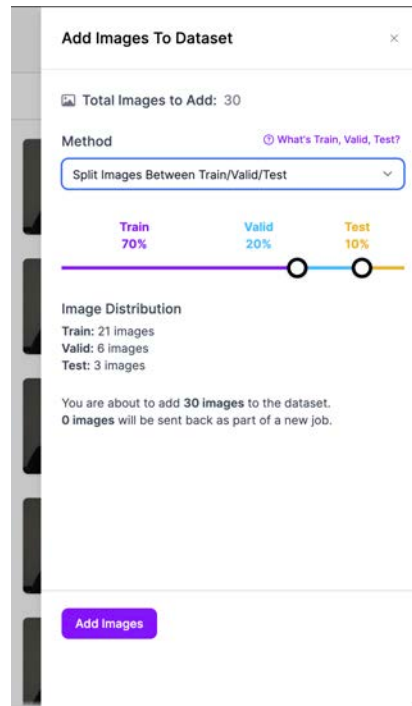


Ilustración 14: Herramienta de división de conjunto de datos en Roboflow

5.2.4 Exportación del conjunto de datos

Una vez que el conjunto de datos ha sido etiquetado y dividido en conjuntos de entrenamiento, validación y testing, se procede a exportarlo en el formato adecuado para su uso en diferentes plataformas y herramientas. En Roboflow, esta exportación se realiza en varios formatos compatibles con diferentes frameworks y bibliotecas de aprendizaje automático. La correcta exportación del conjunto de datos asegura que los datos sean accesibles y utilizables en el entorno de desarrollo elegido.

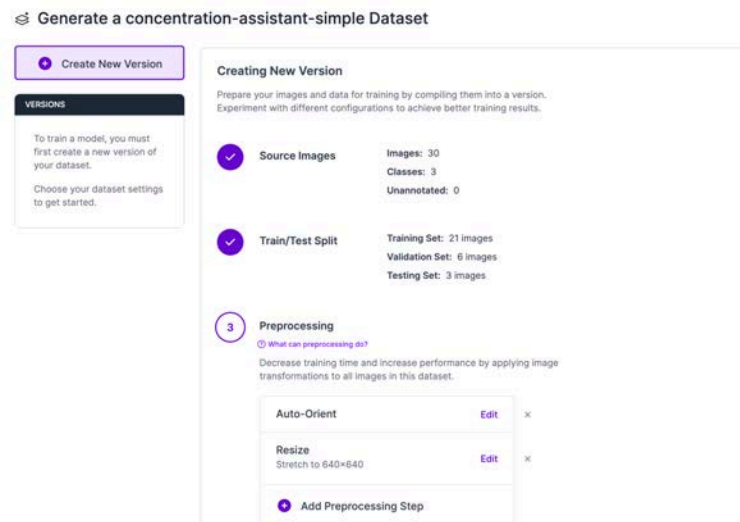


Ilustración 15: Proceso de exportación del conjunto de datos en Roboflow

Roboflow ofrece funcionalidades avanzadas como el preprocesamiento y *data augmentation*. En este primer entrenamiento, sólo se hará una redimensión al tamaño 640x640px sin utilizar opciones de técnicas de *augmentation*, que se muestran en Ilustración 16.

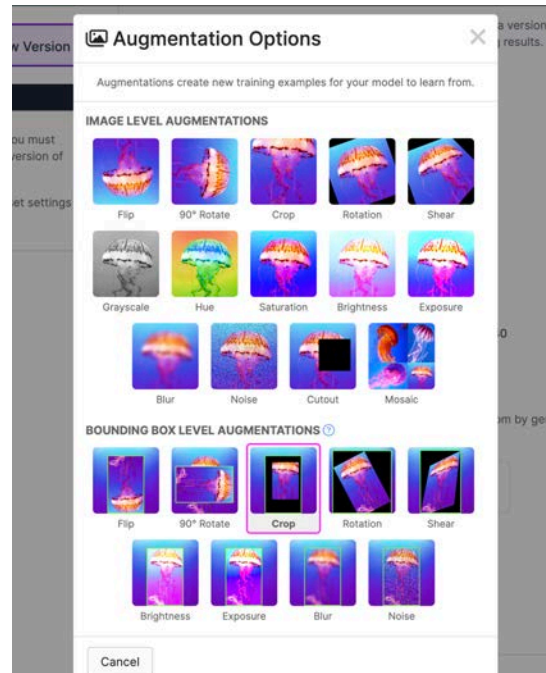


Ilustración 16: Opciones de data augmentation en Roboflow

5.2.5 Entrenamiento de los modelos

Esta etapa denominada también como modelado, lleva a cabo la creación de los modelos de conocimientos a partir de los datos aportados por la etapa de la preparación de los datos. El primer paso establece elegir el algoritmo de modelado, como se vino describiendo a lo largo del proyecto, se usará YOLO. Para entrenar los modelos, se utiliza Google Colaboratory, una plataforma en la nube que proporciona recursos computacionales gratuitos, incluyendo GPUs¹⁶ y TPUs¹⁷, lo cual es ideal para el entrenamiento de modelos de aprendizaje automático que requieren alta capacidad de cómputo. Para importar el conjunto de datos etiquetados y entrenar un modelo con el algoritmo YOLO utilizaremos un código proporcionado por la plataforma que permitirá descargar el mismo directamente en Google Colaboratory mediante unos comandos.

¹⁶ Unidad de Procesamiento Gráfico: Procesador especializado en manejar y acelerar tareas gráficas y de procesamiento paralelo, ampliamente utilizado en el entrenamiento de modelos de aprendizaje automático debido a su capacidad para realizar cálculos en paralelo.

¹⁷ Unidad de Procesamiento Tensorial: Procesador diseñado específicamente para acelerar operaciones de redes neuronales y cálculos tensoriales en aprendizaje automático, optimizando el rendimiento y la eficiencia en la ejecución de modelos de inteligencia artificial.

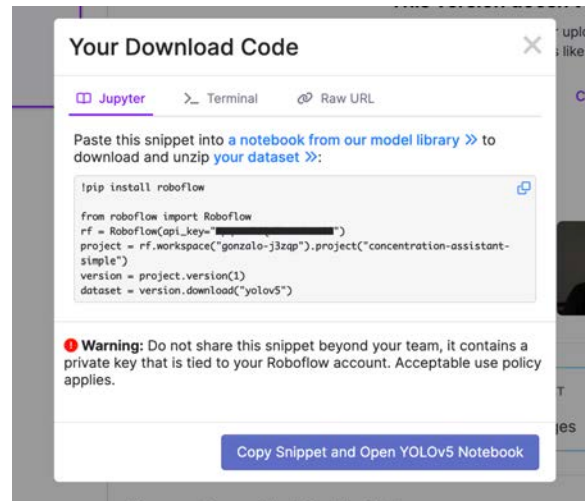


Ilustración 17: Comandos para descargar conjunto de datos desde Roboflow

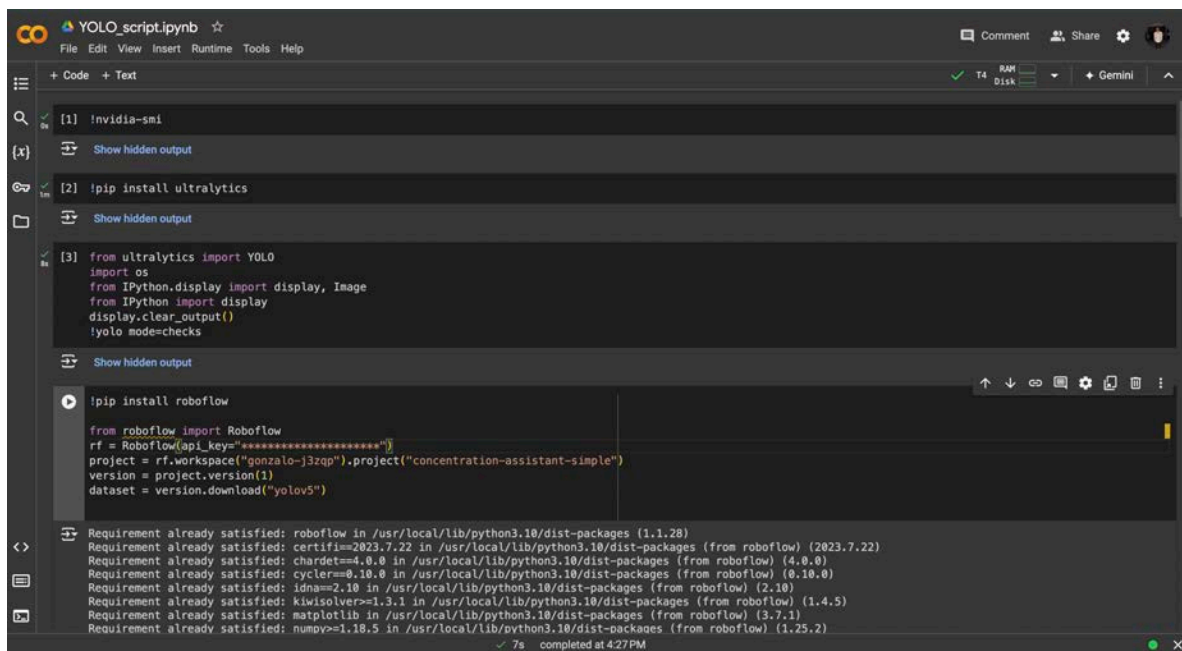


Ilustración 18: Comando iniciales para entrenar modelo con YOLO en Google Colaboratory

Como se puede observar en la Ilustración 18, se necesita en primer lugar correr el comando *nvidia-smi* el cual es útil para verificar que una GPU está disponible y funcionando correctamente en el entorno de Google Colaboratory, así como para monitorizar el uso de la GPU durante el entrenamiento de modelos de aprendizaje automático.

Además, se instala la librería Ultralytics que proporciona métodos simplificados para trabajar con modelos YOLO (You Only Look Once). Facilita el entrenamiento, la evaluación y la inferencia de modelos de detección de objetos, como YOLOv5 y YOLOv8.

El comando para utilizar en cada uno de los entrenamientos posee una configuración como la siguiente:

```
!yolo task=detect mode=train model=yolov8n.pt data=/content/concentration-assistant-simple-7/data.yaml epochs=200 imgsz=640
```

- **task=detect**: Define la tarea como detección de objetos, indicando que el modelo debe aprender a identificar y localizar objetos en imágenes.
- **mode=train**: Establece el modo de operación en entrenamiento, lo que significa que el modelo se ajustará a los datos de entrenamiento proporcionados.
- **model=yolov8n.pt**: Especifica el archivo del modelo preentrenado YOLOv8 Nano que se utilizará como punto de partida para el entrenamiento.
- **data=/content/concentration-assistant-simple-7/data.yaml**: Indica la ubicación del archivo de configuración YAML que define el conjunto de datos y las clases de objetos. Este archivo es esencial para que el modelo entienda cómo leer y utilizar los datos de entrenamiento.
- **epochs=200**: Define el número de épocas, es decir, cuántas veces el modelo pasará por todo el conjunto de datos durante el entrenamiento. Un número mayor de épocas puede mejorar el rendimiento del modelo, pero también puede aumentar el riesgo de sobreajuste.
- **imgsz=640**: Establece el tamaño de las imágenes de entrada a 640x640 píxeles. Este parámetro asegura que todas las imágenes se redimensionen a un tamaño uniforme antes del entrenamiento, lo que ayuda a mejorar la consistencia del modelo.

5.2.6 Entrenamiento 1: YOLOv5 100 épocas

El primer entrenamiento que se realizará será con la versión 5 de YOLO debido a su equilibrio entre precisión y velocidad, siendo una opción efectiva para imágenes de 640x640 píxeles y aplicaciones en tiempo real. YOLOv5n, en particular, se selecciona por su alta velocidad de procesamiento y menor carga computacional, permitiendo un análisis rápido de cada imagen, crucial para aplicaciones que requieren respuestas inmediatas.

Se utilizarán 100 épocas para entrenar el modelo, lo que proporciona una cantidad adecuada de iteraciones para evaluar cómo mejora el rendimiento del modelo. Esta cantidad de épocas es suficiente para obtener un buen equilibrio entre el ajuste del modelo y evitar el sobreajuste. En iteraciones posteriores, se hará el uso de YOLOv8, una versión más reciente y estable.

Model	size (pixels)	mAP _{val} 50-95	mAP _{val} 50	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	45	6.3	0.6	1.9	4.5
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7
YOLOv5n6	1280	36.0	54.4	153	8.1	2.1	3.2	4.6
YOLOv5s6	1280	44.8	63.7	385	8.2	3.6	12.6	16.8
YOLOv5m6	1280	51.3	69.3	887	11.1	6.8	35.7	50.0
YOLOv5l6	1280	53.7	71.3	1784	15.8	10.5	76.8	111.4
YOLOv5x6	1280	55.0	72.7	3136	26.2	19.4	140.7	209.8
+ TTA	1536	55.8	72.7	-	-	-	-	-

Ilustración 19: Tabla de modelos pre entrenados para utilizar con YOLOv5

La tabla que muestra la Ilustración 19 se aprecian las opciones de los modelos pre entrenados para usar en el entrenamiento con el algoritmo YOLO en su versión 5. En este primer entrenamiento se utilizará el modelo YOLOv5n. Aunque existen versiones más avanzadas como YOLOv5s, YOLOv5m, y YOLOv5l, que ofrecen mayor precisión, carga computacional y tiempos de procesamiento más lentos, las cuales son características no deseadas para este proyecto.

Comando para comenzar el entrenamiento del modelo:

```
!yolo task=detect mode=train model=yolov5x.pt data=/content/concentration-assistant-simple-1/data.yaml epochs=100 imgsz=640
```

Luego de unos minutos se obtuvieron los resultados pertenecientes al entrenamiento con el algoritmo YOLOv5, como se muestra en la columna izquierda de la ilustración 20. En la carpeta *weights* se encuentran los modelos de PyTorch generados durante el entrenamiento: uno marcado como *best*, que representa el modelo con el mejor rendimiento en los datos de validación, y el otro como *last*, que es el modelo final después de la última época de entrenamiento. Además, se generaron imágenes con datos de rendimiento del modelo (Anexo 1), incluyendo métricas como precisión, recall y la matriz de confusión, que proporcionan una visión detallada de la efectividad del modelo en la detección de objetos.

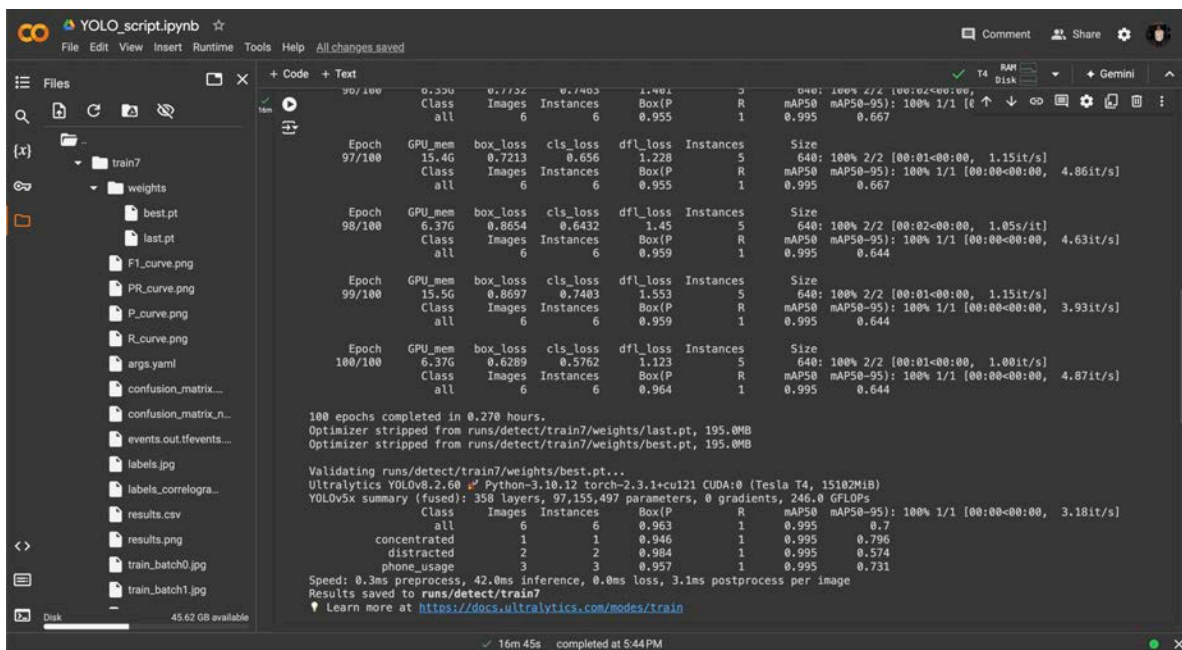


Ilustración 20: Finalización de primer entrenamiento de modelo con YOLOv5

Para comprender el mensaje generado por cada época del entrenamiento del modelo se describe brevemente las columnas:

1. **Epoch:** El número de la época actual y el número total de épocas de entrenamiento.
2. **GPU_mem:** La cantidad de memoria de GPU utilizada durante el entrenamiento.

3. **box_loss**: Pérdida de la caja. Representa el error en la predicción de las coordenadas de las cajas delimitadoras. Un valor menor indica que las cajas predichas se están ajustando mejor a las cajas verdaderas.
4. **cls_loss**: Pérdida de clasificación. Representa el error en la predicción de las clases de los objetos dentro de las cajas delimitadoras. Un valor menor indica una mejor predicción de las clases.
5. **dfl_loss**: Pérdida de distribución focalizada (*Distribution Focal Loss*). Se refiere a una pérdida utilizada para mejorar la precisión de la ubicación de los bordes de las cajas delimitadoras, proporcionando una mejor precisión espacial.
6. **Instances**: Número de instancias de objetos detectados en una imagen durante el entrenamiento.
7. **Size**: Tamaño de las imágenes utilizadas durante el entrenamiento.

El entrenamiento con el algoritmo YOLOv5 concluyó con éxito después de 100 épocas. Siguiendo la metodología CRISP-DM, luego de cada entrenamiento sigue la etapa de evaluación, que es donde se analiza si el modelo cumple con las expectativas desarrolladas en la comprensión del negocio. A continuación se presentan los puntos más relevantes de los resultados:

1. **Pérdidas (*Losses*):**
 - o La pérdida de la caja (**box_loss**) disminuyó de 0.9203 en la época 91 a 0.6289 en la época 100. Lo que es bastante significativo y quiere decir que mejoró bastante la delimitación del objeto en una caja entre esas épocas.
 - o La pérdida de la clasificación (**cls_loss**) también mostró una reducción general, terminando en 0.5762.
2. **Métricas de rendimiento:**
 - o Precisión (P): Alcanzó 0.964 al final del entrenamiento.
 - o Recall (R): Se mantuvo constante en 1 a lo largo de las épocas.
3. **Validación Final:**
 - o La precisión (P) final fue de 0.963.
 - o El recall (R) final fue de 1.

En resumen, el modelo ha demostrado un excelente rendimiento en términos de precisión y recall. El valor de *recall* de 1 indica que el modelo es excelente para detectar todos los objetos relevantes en las imágenes proporcionadas. Esto puede significar que el conjunto de validación no es suficientemente desafiante o que el modelo está sobre ajustado a los datos de entrenamiento aprendiendo patrones sumamente específicos de los datos en vez de generalizarlos. Esto indica que el modelo tiene una falta de fiabilidad, por lo que es necesario realizar validaciones adicionales para asegurar que el modelo funcione bien en situaciones reales. No es entonces un modelo candidato para su uso.

5.2.7 Entrenamiento 2: YOLOv8 100 épocas

Luego de haber entrenado y evaluado el primer modelo, debido a que la metodología es de tipo cíclica y se puede volver a fases anteriores, volveremos a la etapa de modelado y a posterior su evaluación. Con el modelo YOLOv5 ya entrenado, se procede a utilizar YOLOv8, el cual ofrece

mejoras significativas en precisión y eficiencia en comparación con sus predecesores. Hace uso de técnicas avanzadas y optimizaciones que lo posicionan como una de las mejores opciones para tareas de visión por computadora.

En este segundo entrenamiento, utilizaremos el modelo YOLOv8n (nano) en lugar de versiones más potentes como YOLOv8x. La elección de YOLOv8n se debe a su excelente equilibrio entre velocidad y precisión, esencial para aplicaciones en tiempo real. Aunque modelos más avanzados como YOLOv8x ofrecen una mayor precisión resultan en tiempos de procesamiento más lentos y una carga computacional significativamente mayor.

Con este nuevo entrenamiento, esperamos obtener una comparación clara entre YOLOv5 y YOLOv8, evaluando cómo las mejoras en YOLOv8 impactan en la precisión y velocidad del modelo para la aplicación a desarrollar.

Model	size (pixels)	mAP ^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Ilustración 21: Tabla de modelos pre entrenados para utilizar con YOLOv8

Luego de realizar el mismo proceso de exportar el conjunto de datos con Roboflow con la versión 8 de YOLO e importarlo en Google Colaboratory se procedió a ejecutar el nuevo entrenamiento con la misma cantidad de 100 épocas:

`!yolo task=detect mode=train model=yolov8n.pt data=/content/concentration-assistant-simple-1/data.yaml epochs=100 imgsz=640`

```

YOLO_script.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
99/100 2.3% W.013V 1.237 0.3446 0 0.995 1 0.995 0.666
Class Images Instances Box(P R mAP50 mAP50-95): 100% 1/1 [6 4 0 0:00:00:00, 6.581it/s]
Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
97/100 2.3% 0.6188 1.221 0.9563 5 640: 100% 2/2 [00:00:00:00, 5.581it/s]
Class Images Instances Box(P R mAP50 mAP50-95): 100% 1/1 [00:00:00:00, 13.541it/s]
Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
98/100 2.3% 0.632 1.226 0.9461 5 640: 100% 2/2 [00:00:00:00, 5.721it/s]
Class Images Instances Box(P R mAP50 mAP50-95): 100% 1/1 [00:00:00:00, 13.521it/s]
Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
99/100 2.3% 0.6344 1.318 0.9769 5 640: 100% 2/2 [00:00:00:00, 7.171it/s]
Class Images Instances Box(P R mAP50 mAP50-95): 100% 1/1 [00:00:00:00, 9.731it/s]
Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
100/100 2.3% 0.652 1.208 0.9815 5 640: 100% 2/2 [00:00:00:00, 3.481it/s]
Class Images Instances Box(P R mAP50 mAP50-95): 100% 1/1 [00:00:00:00, 7.451it/s]

100 epochs completed in 0.836 hours.
Optimizer stripped from runs/detect/train/weights/best.pt, 6.39M
Optimizer stripped from runs/detect/train/weights/best.pt, 6.39M

Validating runs/detect/train/weights/best.pt...
Ultralytics YOLOv8.2.68 Python-3.10.12 torch-2.3.1+cu121 CUDA:0 (Tesla T4, 15182MiB)
Model summary (fused): 168 layers, 3,806,233 parameters, 0 gradients, 8.1 GFLOPs
Class Images Instances Box(P R mAP50 mAP50-95): 100% 1/1 [00:00:00:00, 8.471it/s]
concentrated 1 1 0.924 1 0.995 0.895
distracted 2 2 0.961 1 0.995 0.547
pheme_usage 3 3 0.988 1 0.995 0.653

Speed: 0.2ms preprocess, 4.2ms inference, 0.0ms loss, 4.1ms postprocess per image
Results saved to runs/detect/train
Learn more at https://docs.ultralytics.com/models/train
2m 34s completed at 6:09 PM

```

Ilustración 22: Finalización de entrenamiento con YOLOv8 con 100 épocas

El entrenamiento con el algoritmo YOLOv8 concluyó después de 100 épocas arrojando los siguientes resultados:

1. Pérdidas
 - o Box loss: Disminuyó de 0.7763 a 0.652 en la época 100 que no fue tan significativo
 - o Class loss: Disminuyó de 1.391 a 1.208, tampoco ha sido tan significativo
2. Métricas de rendimiento:
 - o Precisión (P): Alcanzó hasta 0.956, similar al entrenamiento de YOLOv5
 - o Recall (R): Se mantuvo constante en 1
3. Validación Final:
 - o Precisión (P): 0.958
 - o Recall (R): 1

5.2.8 Evaluación: YOLOv5 y YOLOv8 - 100 épocas

Comparar los modelos YOLOv5 y YOLOv8 proporciona información valiosa sobre sus rendimientos. A continuación se presentan las comparaciones basadas en los resultados proporcionados:

1. **Consumo de Memoria GPU:** YOLOv8 es significativamente más eficiente en términos de consumo de memoria GPU, con un uso constante de 2.3G frente a los 6.35G a 15.5G de YOLOv5.
2. **Pérdidas:** Ambas versiones muestran reducciones en las pérdidas a lo largo de las épocas. Sin embargo, YOLOv5 tiene una pérdida de clasificación más baja al final del entrenamiento (0.5762 vs. 1.208).
3. **Métricas de rendimiento:** Precisión y Recall: Ambas versiones alcanzan altos niveles de precisión y recall (cerca de 1), aunque YOLOv8 muestra un valor de precisión ligeramente menor al final del entrenamiento.
4. **Validación Final:** Las clases específicas muestran que YOLOv8 tiene un rendimiento superior en "*concentrated*" (0.895 vs. 0.796) y un rendimiento inferior en "*phone usage*" (0.653 vs. 0.731) en comparación con YOLOv5.

Ambos modelos presentan un buen rendimiento, pero YOLOv8 demuestra ser más eficiente en términos de consumo de memoria GPU y tiene un rendimiento ligeramente superior durante la validación final. Sin embargo, YOLOv5 muestra pérdidas de clasificación más bajas y tiene un rendimiento superior en la clase "*phone usage*". A continuación, se evaluará de manera práctica los modelos.

Para evaluar los modelos de manera práctica se desarrolló otro *script* en Python. En el mismo se hace uso de la librería de Ultralytics y además openCV para acceder a la cámara web y dibujar en la imagen el cuadro delimitador con su etiqueta y la confianza¹⁸ de la clasificación. Para ambos algoritmos se utiliza el modelo de mejor rendimiento *best.pt* con una confianza mínima de 0.9, es decir si la confianza de clasificación es menor, no se graficará el cuadro delimitador.

¹⁸ Medida en que un modelo predice con certeza o seguridad su salida. Representa la certeza del modelo en una predicción específica, que se expresa a menudo como una probabilidad o una puntuación.

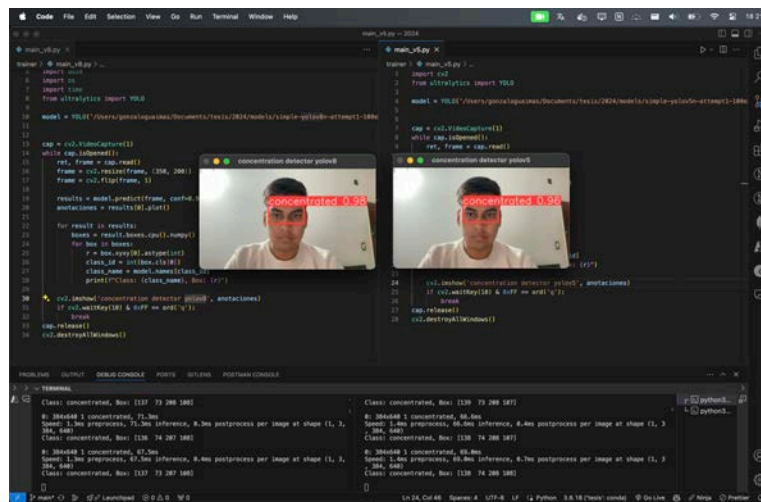


Ilustración 23: Comparación de estado “concentrado” entre modelos entrenados con YOLOv5 y YOLOv8

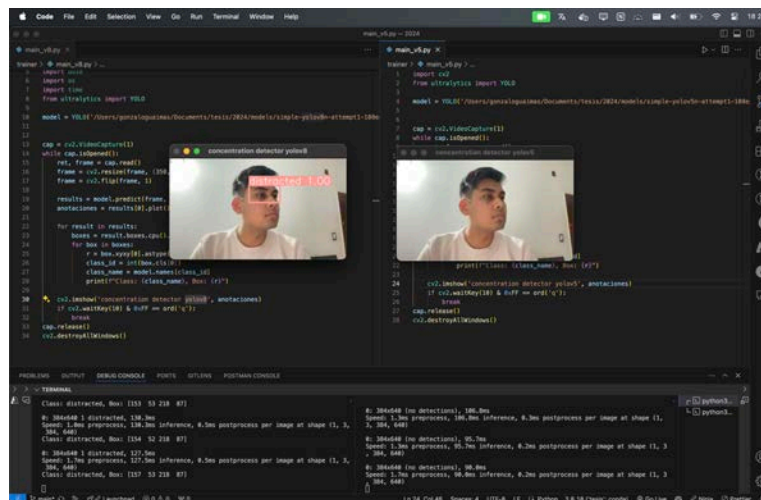


Ilustración 24: Comparación de estado “desconcentrado” entre modelos entrenados con YOLOv5 y YOLOv8

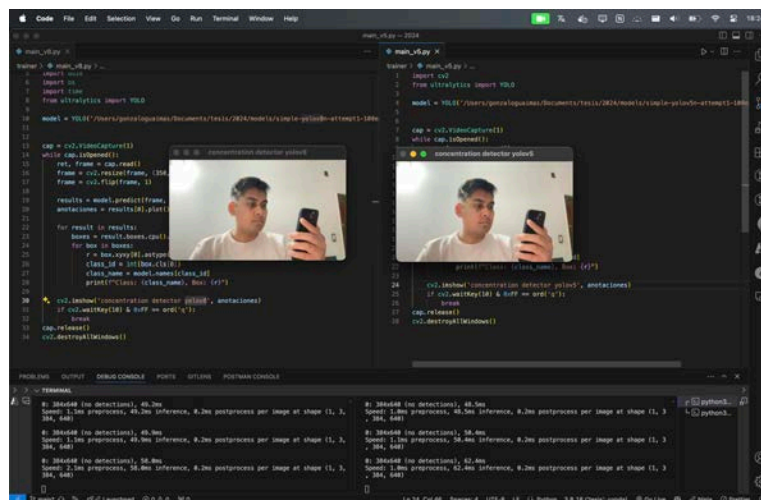


Ilustración 25: Comparación de estado "uso del teléfono" entre modelos entrenados con YOLOv5 y YOLOv8

Cómo se logra observar en las Ilustraciones 23, 24 y 25 ambos modelos son utilizados al mismo tiempo con el fin de comparar la operabilidad, del lado izquierdo YOLOv8 y del derecho YOLOv5. Se puede concluir, que a pesar de que las medidas de rendimiento son similares, en la práctica el mejor modelo es el de YOLOv8 ya que puede lograr detectar el estado de desconcentración además que el de concentración a diferencia de YOLOv5, mientras que ambos no lograron detectar el uso del teléfono con una confianza de al menos el 90%.

De esta manera se concluye que el modelo entrenado con YOLOv8 es superior al modelo YOLOv5, pero aun así no es lo suficientemente robusto para detectar de manera precisa las diferentes clases de los estados de concentración y el uso del teléfono.

5.2.9 Entrenamiento 3: YOLOv8 100 épocas con *augmentation*

En este nuevo entrenamiento, se reutilizará el conjunto de datos incorporando técnicas de *data augmentation*. La fase a la que retornamos de la metodología CRISP-DM es la preparación de los datos, debido a que se generará otro conjunto de datos, distinto con el que se comenzó entrenando los modelos. Recordemos que el *data augmentation* es una estrategia utilizada para diversificar y aumentar el tamaño del conjunto de datos, aplicando transformaciones como rotaciones, cambios de brillo y contraste, recortes, entre otros.

Al aplicar esta técnica, se espera que el modelo YOLOv8 mejore su precisión y robustez. Este proceso también ayuda a mitigar problemas de sobreajuste, asegurando que el modelo no solo memorice las imágenes de entrenamiento, sino que aprenda características relevantes que son útiles en un contexto más amplio.

Para este entrenamiento, se utilizarán nuevamente 100 épocas, lo que permitirá evaluar cómo la inclusión de *data augmentation* influye en la mejora del rendimiento del modelo.

Las opciones de *data augmentation* que se utilizarán se muestran en las siguientes Ilustraciones 26 y 27:

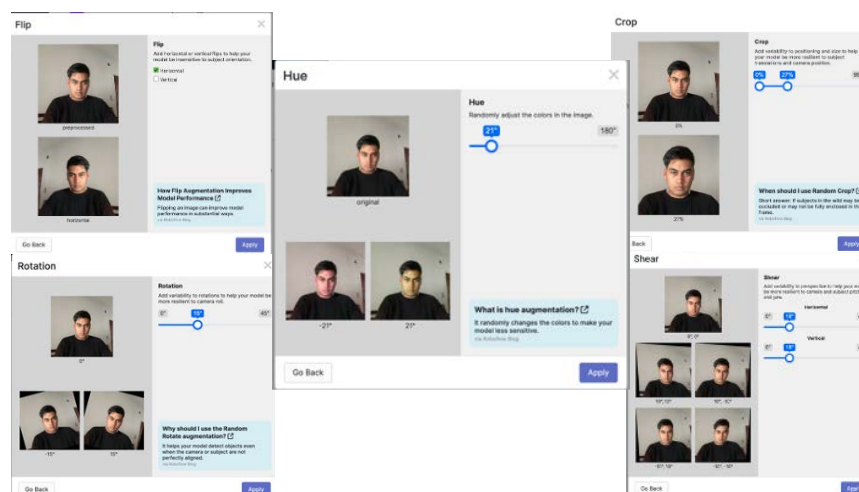


Ilustración 26: Primera parte de opciones elegidas de *data augmentation*

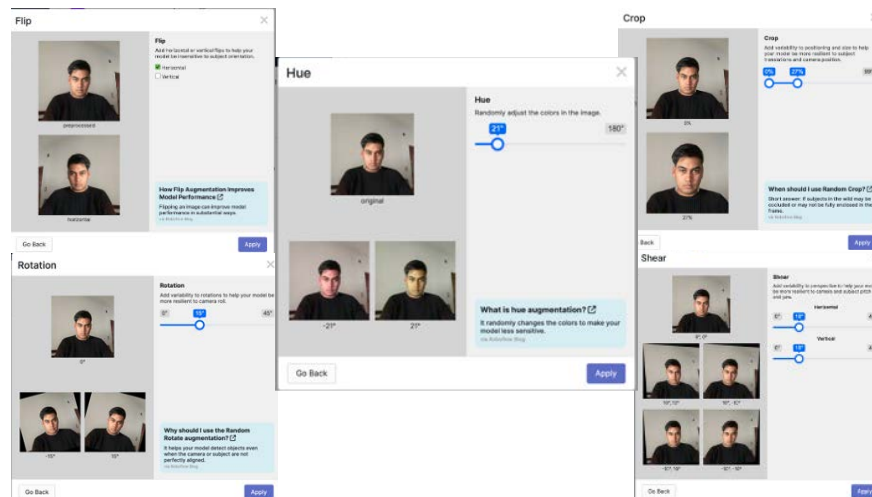


Ilustración 27: Segunda parte de opciones elegidas de data augmentation

Una vez seleccionadas las técnicas, Roboflow muestra un resumen de la selección, como se observa en ilustración 28.



Ilustración 28: Técnicas seleccionadas para data augmentation

Al incorporar las técnicas, el conjunto de datos se ve diversificado y se agranda de 30 imágenes a un total de 72, 63 para entrenamiento, 6 para validación y 3 para pruebas.

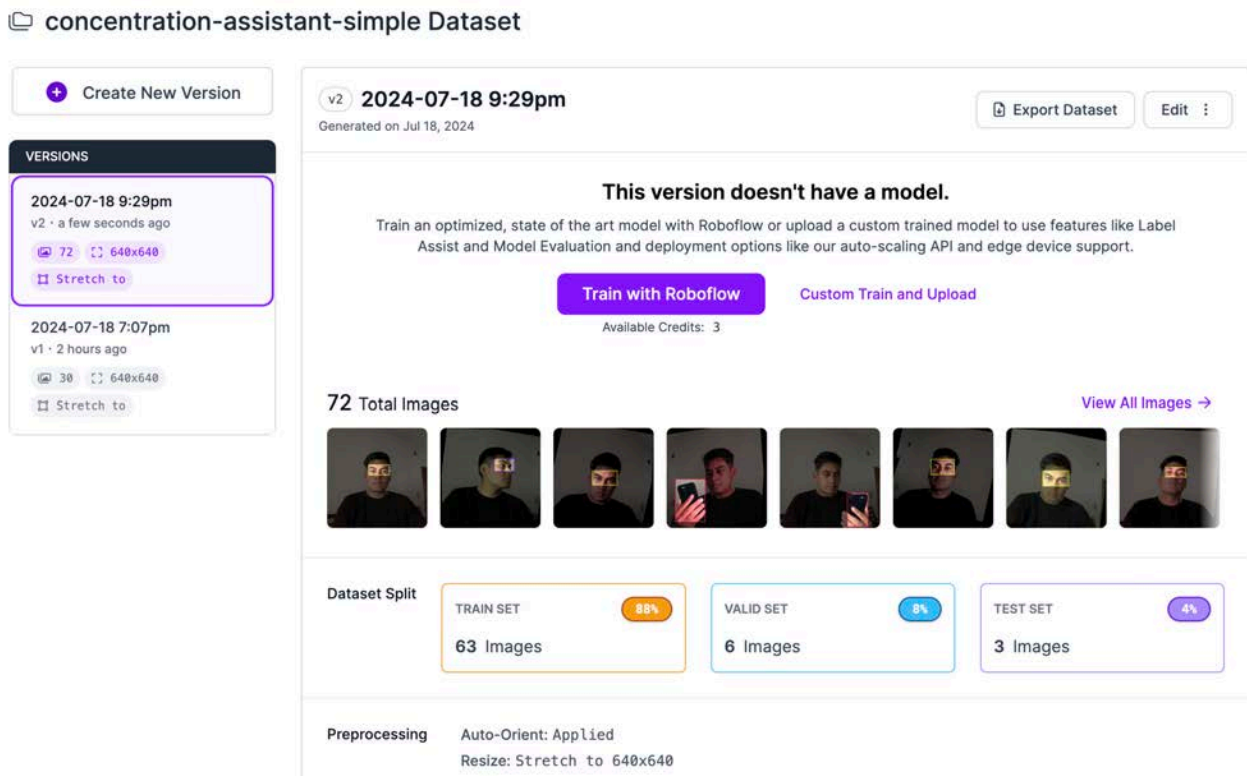


Ilustración 29: Conjunto de datos generado con técnicas de augmentation en plataforma Roboflow

Luego de repetir el mismo procedimiento de entrenar el modelo en Google Colaboratory con YOLOv8 y un conjunto de datos con *augmentation* se procede a continuación a comparar los tres modelos hasta el momento obtenidos (YOLOv8 sin aumento, YOLOv5 y YOLOv8 con *data augmentation*).

Entrenamiento:

1. Pérdidas:
 - o Box loss: Disminuyó de 0.6582 a 0.5528 en la última época
 - o Class loss: Disminuyó de 0.8918 a 0.8248 en la última época
2. Métricas de rendimiento:
 - o Precisión (P): Alcanzó hasta 0.968
 - o Recall (R): Se mantuvo constante en 1

Validación Final:

- Precisión (P): 0.902
- Recall (R): 1

Comparación YOLOv8 sin *augmentation*, YOLOv5 y YOLOv8 con *augmentation*.

1. Pérdidas:
 - o Box loss:

- YOLOv8 sin *augmentation*: 0.652
 - YOLOv5: 0.6289
 - YOLOv8 con *augmentation*: 0.5528
 - o Class loss:
 - YOLOv8 sin *augmentation*: 1.208
 - YOLOv5: 0.5762
 - YOLOv8 con *augmentation*: 0.8248
2. Métricas de rendimiento:
- o Precisión (P):
 - YOLOv8 sin *augmentation*: Hasta 0.956
 - YOLOv5: Hasta 0.964
 - YOLOv8 con *augmentation*: Hasta 0.968
 - o Recall (R): Constante en 1 para todos.
3. Validación Final:
- o Precisión (P):
 - YOLOv8 sin *augmentation*: 0.958
 - YOLOv5: 0.963
 - YOLOv8 con *augmentation*: 0.902
 - o Recall (R): 1 para todos.

Conclusión

1. Pérdidas:
- o YOLOv8 con *data augmentation*: muestra la menor pérdida general al final del entrenamiento, lo cual indica una mejor adaptación al conjunto de datos con las técnicas de *augmentation* aplicadas.
2. Precisión y Recall:
- o Todos los modelos tienen un recall perfecto (1), pero la precisión final es más alta en YOLOv5 y YOLOv8 sin *augmentation* en comparación con YOLOv8 con *augmentation*, aunque este último sigue siendo competitivo.

Luego de realizar un análisis de las precisiones y *recall*, se procede a ejecutar el modelo en el *script* y evaluar su operabilidad de manera práctica.

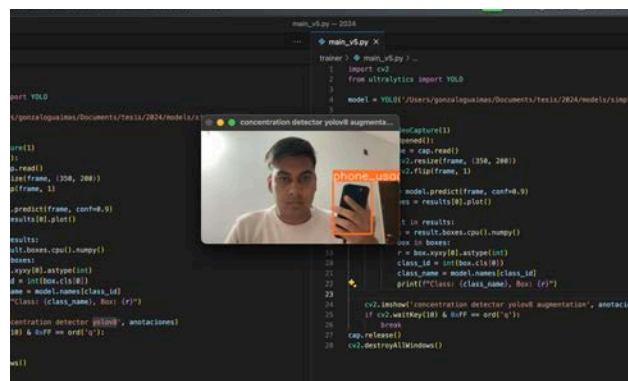


Ilustración 30: Detección de clase "uso del teléfono" con modelo entrenado con YOLOv8 con *data augmentation*

Se observa de manera práctica que el modelo es capaz de reconocer muy bien la clase “uso del teléfono”, como se muestra en Ilustración 30, sin embargo, no es capaz de reconocer el estado concentrado ni desconcentrado con una confianza mínima del 90%, por lo que no es un modelo prometedor.

5.2.10 Entrenamiento 4: YOLOv8 200 épocas con menos *augmentation*

En este punto, se realizará un nuevo entrenamiento del modelo YOLOv8, extendiendo el número de épocas a 200 y utilizando una menor cantidad de técnicas de *data augmentation* en comparación con el entrenamiento anterior. El objetivo de esta estrategia es encontrar un equilibrio entre la diversidad del conjunto de datos y la estabilidad del entrenamiento.

Reducir la cantidad de técnicas puede ayudar a evitar el sobreajuste a transformaciones específicas, permitiendo al modelo centrarse más en las características de los datos. Al incrementar el número de épocas a 200, se le brinda al modelo más oportunidades para ajustar sus parámetros y mejorar su rendimiento.

Con este enfoque, se espera que el modelo YOLOv8 logre un mejor rendimiento al beneficiarse tanto de la mayor cantidad de datos a través de *augmentation* como de un periodo de entrenamiento más prolongado. Esta combinación puede resultar en un mejor modelo.

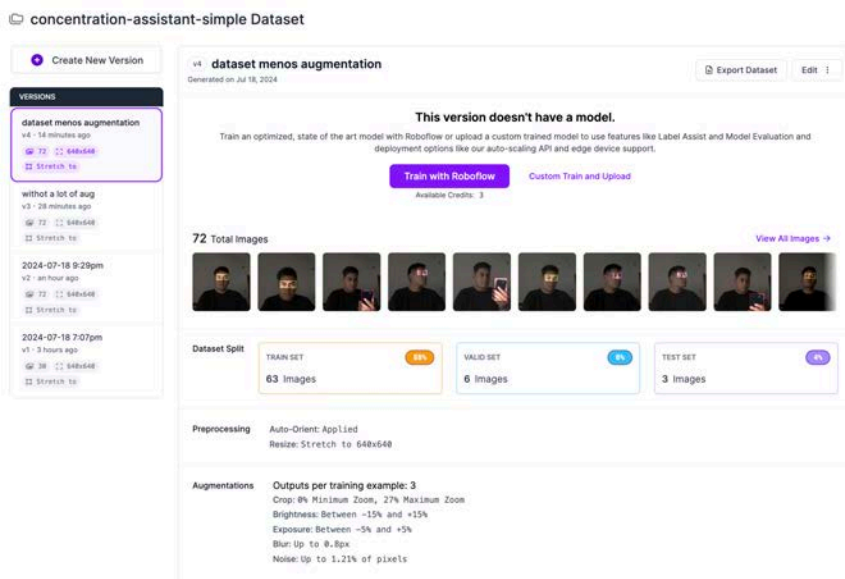


Ilustración 31: Nuevo conjunto de datos con menos técnicas de *data augmentation*

En esta oportunidad, el entrenamiento del modelo se detuvo antes de completar las 200 épocas previstas debido al criterio de parada temprana, esto se observa en la Ilustración 32. Este criterio monitorea métricas de rendimiento del modelo y detiene el entrenamiento cuando no se observa una mejora significativa después de un número determinado de épocas. La parada temprana previene el sobreajuste, asegurando que el modelo no continúe ajustándose a los datos de entrenamiento sacrificando su capacidad para generalizar a datos nuevos. Esta estrategia

optimiza el uso de recursos computacionales y garantiza que el modelo alcance su mejor rendimiento sin desperdiciar tiempo en épocas adicionales innecesarias.

```

+ Code + Text
Epoch 170/200
GPU_mem 2.26G
Class Images
all 6
box_loss 0.4082
Instances 6
cls_loss 0.41
Box(P) 0.8463
Instances 26
Size 640: 100% 4/4 [00:00:00:00, 4.93it/s]
mAP50 mAP50-95: 100% 1/1 [00:00:00:00, 6.20it/s]
0.995 0.694

Epoch 171/200
GPU_mem 2.26G
Class Images
all 6
box_loss 0.3642
Instances 6
cls_loss 0.4221
Box(P) 0.8473
Instances 20
Size 640: 100% 4/4 [00:00:00:00, 5.81it/s]
mAP50 mAP50-95: 100% 1/1 [00:00:00:00, 6.06it/s]
0.995 0.693

Epoch 172/200
GPU_mem 2.26G
Class Images
all 6
box_loss 0.347
Instances 6
cls_loss 0.3948
Box(P) 0.8305
Instances 24
Size 640: 100% 4/4 [00:00:00:00, 5.28it/s]
mAP50 mAP50-95: 100% 1/1 [00:00:00:00, 9.49it/s]
0.995 0.693

EarlyStopping: Training stopped early as no improvement observed in last 100 epochs. Best results observed at epoch 72, best model :
To update EarlyStopping(patience=100) pass a new patience value, i.e. 'patience=300' or use 'patience=0' to disable EarlyStopping.

172 epochs completed in 0.097 hours.
Optimizer stripped from runs/detect/train14/weights/last.pt, 6.3MB
Optimizer stripped from runs/detect/train14/weights/best.pt, 6.3MB

Validating runs/detect/train14/weights/best.pt...
Ultralytics YOLOv8.2.60 Python-3.10.12 torch-2.3.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
Model summary (fused): 168 layers, 3,006,233 parameters, 0 gradients, 8.1 GFLOPs
Class Images Instances Box(P) R mAP50 mAP50-95: 100% 1/1 [00:00:00:00, 17.27it/s]
all 6 6 0.962 1 0.995 0.752
concentrated 1 1 0.969 1 0.995 0.895
distracted 2 2 0.971 1 0.995 0.597
phone_usage 3 3 0.948 1 0.995 0.764

Speed: 0.2ms preprocess, 2.2ms inference, 0.0ms loss, 1.1ms postprocess per image
Results saved to runs/detect/train14
Learn more at https://docs.ultralytics.com/modes/train

```

Ilustración 32: Parada temprana en entrenamiento de modelo de 200 épocas

Algunas conclusiones de este entrenamiento son:

1. Pérdidas:

- o Box Loss: Ha fluctuado entre 0.347 y 0.4082, con una tendencia general a la baja, indicando una mejora en la precisión de las cajas delimitadoras a lo largo del entrenamiento.
- o Class Loss: También ha fluctuado, pero en general ha sido bastante estable, sugiriendo que el modelo ha aprendido a clasificar las instancias de manera efectiva.

2. Detección Temprana:

- o El entrenamiento se detuvo temprano debido a que no se observó una mejora en las métricas durante las últimas 100 épocas. Esto sugiere que el modelo alcanzó un punto donde no se observaron mejoras significativas en las métricas de evaluación.

Las estadísticas del modelo presentan un rendimiento excelente en términos de precisión. La decisión de la detención del entrenamiento temprano parece razonable, ya que las métricas de validación se mantuvieron estables y altas. Aún así, se realizaron las evaluaciones prácticas y no se obtuvieron los resultados esperados, ya que el modelo no detectaba de manera precisa las diferentes clases. Esto indica que se debe reconsiderar la estrategia ya que no se están cumpliendo los requerimientos planteados en la comprensión del negocio, que era lograr identificar de manera precisa y clara los estados de concentración y el uso del teléfono. Se debe crear un conjunto de datos distinto para entrenar un modelo que detecte de manera más efectiva.

Volviendo a la etapa de comprensión del negocio y de los datos, se pensó que una posible mejora es cambiar el enfoque en el etiquetado en las imágenes. Anteriormente, se usaban sólo los ojos de la persona para determinar si estaba concentrado o desconcentrado. En lugar de esto, ahora se definirá a una persona como concentrada si el área etiquetada incluye desde los ojos hasta la

boca, y como desconcentrada si se etiqueta de perfil, incluyendo la nariz, ojos y boca. Esta nueva estrategia debería proporcionar al modelo más información relevante para mejorar su capacidad de detección y clasificación.

5.2.11 Entrenamiento 5: YOLOv8 200 épocas con poco *augmentation* y otro enfoque

Nuevamente el proyecto se encuentra en la etapa de preparación de los datos ya que se adoptará un nuevo enfoque en el etiquetado de las imágenes. En lugar de centrarse únicamente en los ojos, se ampliará la región etiquetada para incluir desde los ojos hasta la boca para clasificar la concentración y desconcentración. Este cambio tiene como objetivo proporcionar al modelo más información contextual, mejorando su capacidad para detectar y clasificar correctamente las diferentes clases. Adicionalmente, el entrenamiento se realizará con imágenes del rostro de otra persona, lo que ayudará a aumentar la variabilidad del conjunto de datos y mejorar la robustez del modelo en la detección de diferentes individuos. Con estas modificaciones, se espera que el modelo entrenado con YOLOv8 logre un rendimiento superior en la tarea de detección de concentración y desconcentración.

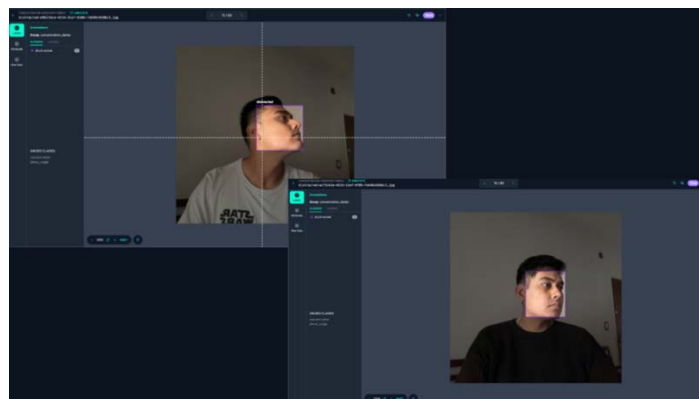


Ilustración 33: Nuevo cuadro delimitador para etiqueta “desconcentrado”

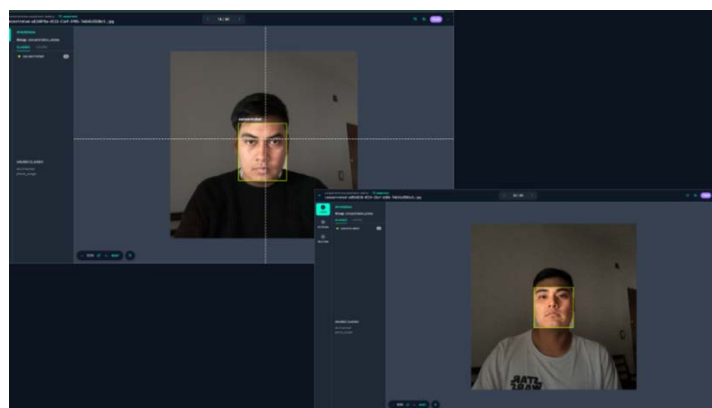


Ilustración 34: Nuevo cuadro delimitador para etiqueta “concentrado”

El nuevo conjunto de datos con algunas técnicas de *augmentation* aplicadas se compone de 128 imágenes de entrenamiento, 12 de validación y 6 de prueba, lo que hace un conjunto de datos bastante completo y robusto a la vez.

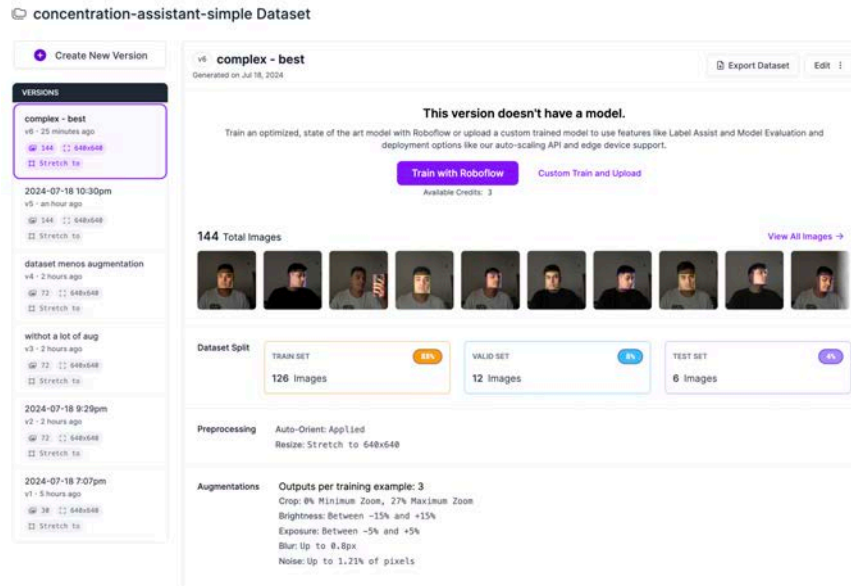


Ilustración 35: Conjunto de datos de dos perfiles con todas algunas técnicas de data augmentation

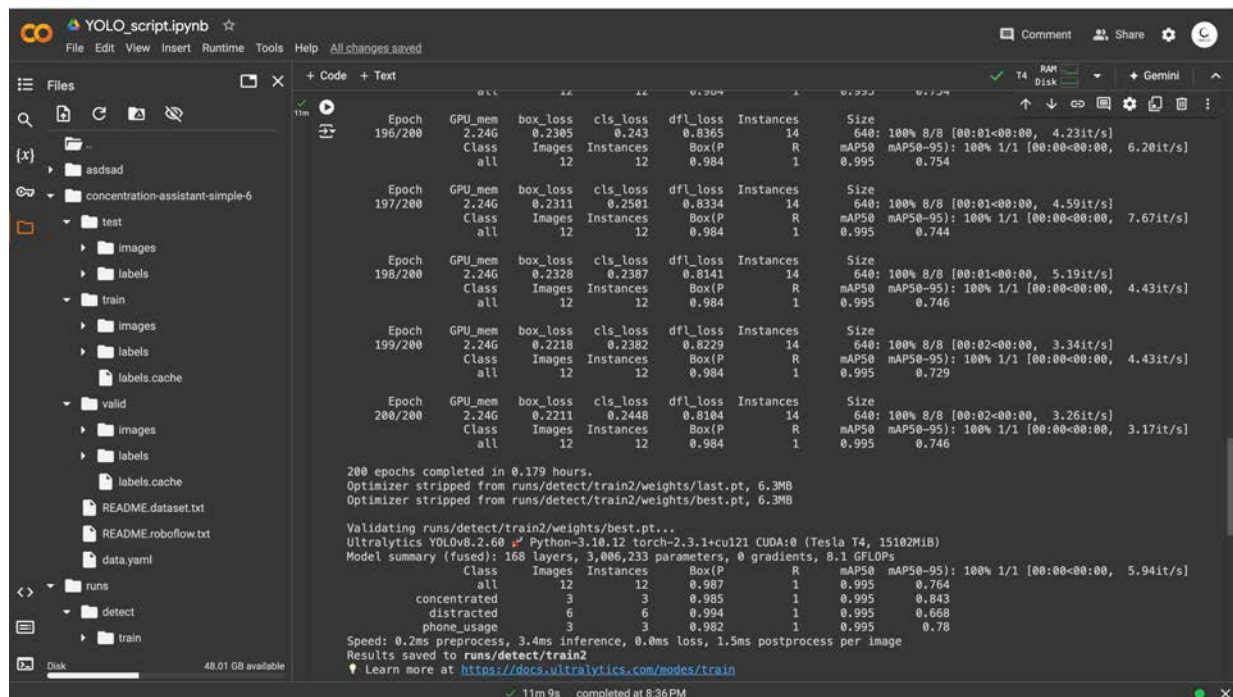


Ilustración 36: Finalización del entrenamiento del conjunto de datos más completo

Con este nuevo conjunto de datos, el entrenamiento no tuvo detención temprana y llegó a las 200 épocas completas. En las pruebas prácticas, el modelo arrojó excelentes resultados,

detectando las tres clases con una confianza mínima del 90% de manera adecuada. Esto indica una notable mejora en la capacidad del modelo para diferenciar entre los estados de concentración y desconcentración, gracias a las técnicas de *augmentation* y el nuevo enfoque de etiquetado.

Además, la inclusión de imágenes del rostro de otra persona en el conjunto de datos contribuyó significativamente a la robustez del modelo. Este resultado muestra que el entrenamiento con YOLOv8, con este enfoque y conjunto de datos, puede ser el más adecuado y el mejor candidato para ser utilizado en el prototipo del asistente de concentración.

Es importante mencionar que, aunque el modelo alcanzó un rendimiento excelente, existe la posibilidad de seguir mejorando si se aumentara el número de épocas de entrenamiento. Extender el entrenamiento más allá de las 200 épocas podría permitir al modelo afinar aún más sus parámetros y lograr una precisión y robustez aún mayores. No obstante, los resultados actuales ya demuestran que el modelo es altamente efectivo y cumple con los requisitos del proyecto, posicionándolo como la mejor opción para la implementación en el asistente de concentración.

5.2.12 Entrenamiento 6: YOLOv8 200 épocas con todas las técnicas de *augmentation* y otro enfoque

En este último entrenamiento, se llevó a cabo un entrenamiento del modelo con 200 épocas, incorporando todas las técnicas de *data augmentation* disponibles para diversificar y enriquecer el conjunto de datos.

A pesar de aumentar el número de técnicas, el modelo no logró un rendimiento satisfactorio en las pruebas prácticas. La detección de las diferentes clases no fue precisa, lo que indica que, a pesar del uso extensivo de técnicas de *augmentation* y el nuevo enfoque de etiquetado, el modelo no alcanzó el nivel de efectividad esperado.

5.2.13 Evaluación final y selección del modelo

Por cada uno de los entrenamientos se ha realizado una evaluación de los modelos teniendo en cuenta si cumplen con los requerimientos planteados en la comprensión del negocio. Ahora llega el momento de compararlos a todos y realizar una evaluación final de los resultados obtenidos. A continuación, se presenta una tabla con las conclusiones de cada uno de los modelos entrenados con el fin de justificar la elección.

Entrenamiento	Conclusión
1.-YOLOv5 – 100 épocas – sin <i>augmentation</i>	El modelo se entrenó con la versión 5 de YOLO con 100 épocas, obtuvo 96.3% de precisión y un 100% de recall, lo que en números significa que es un modelo excelente. A pesar de esto, en machine learning un modelo tan bueno puede significar problemas, como el sobreajuste, en donde el modelo aprende patrones específicos y no de manera generalizada. Este modelo por lo tanto tenía una falta de fiabilidad, por lo que se procedió

	a validar su uso y efectivamente carecía de precisión y sólo podía reconocer el estado “concentrado”.
2.- YOLOv8 – 100 épocas – sin augmentation	En este segundo entrenamiento se utilizó YOLOv8 con 100 épocas el cual contó con una precisión del 95.8% y un 100% de recall, al igual que el modelo anterior, poseía el mismo problema de fiabilidad. En su validación práctica este modelo sólo pudo clasificar el estado “concentrado” y “desconcentrado”, lo que se consideraría mejor que el anterior modelo, pero aun así no podría ser elegido.
3.- YOLOv8 – 100 épocas – con augmentation	Se entrenó el modelo con YOLOv8 y 100 épocas, en este, se usaron todas las técnicas de data augmentation que proporciona la plataforma Roboflow. Las métricas de precisión y recall fueron similares a los anteriores, con la diferencia de que en la práctica el modelo sólo era capaz de reconocer la clase de “uso del teléfono”.
4.- YOLOv8 – 200 épocas – con menos augmentation	En este caso se entrenó el modelo con 200 épocas y menos técnicas de data augmentation. El entrenamiento no llegó hasta las 200 épocas por la detección temprana, el cual permite optimizar el uso de recursos cuando el modelo no mejora a lo largo de las iteraciones. En fines prácticos el modelo no detectaba con precisión ninguna de las clases, por lo que se descartó. En este punto se decidió tomar otro enfoque sobre el conjunto de datos, ya que se optaba por entrenar las clases a través de la zona de los ojos y los modelos no eran capaces de aprender de manera adecuada.
5.- YOLOv8 – 200 épocas – con menos augmentation – otro enfoque.	El nuevo conjunto de datos consistió en etiquetar el rostro completo de la persona, lo que debería de permitir que el modelo pueda aprender patrones más específicos para diferenciar cada clase. Además, se utilizó otra persona más para diversificar el conjunto de datos. El entrenamiento con YOLOv8 y 200 épocas y algunas técnicas de augmentation no tuvo detección temprana, lo que significa que el modelo mejoró considerablemente durante cada iteración. En las pruebas prácticas el modelo fue capaz de reconocer todas las etiquetas de manera precisa con una confianza mínima del 90%. Este modelo se lo consideró como candidato para el uso dentro del aplicativo por su excelente rendimiento.
6.- YOLOv8 – 200 épocas – con augmentation – otro enfoque.	Un último entrenamiento se realizó de la misma manera que el modelo anterior, pero esta vez usando todas las técnicas de data augmentation para diversificar y enriquecer más el conjunto de datos, esperando que este sea más robusto que el anterior. A pesar de esto, en la práctica el modelo no era preciso a la hora de clasificar los estados, por lo que se tuvo que descartar el mismo.

Tabla 4: Tabla de conclusiones de modelos entrenados

Después de completar múltiples entrenamientos con diferentes configuraciones y enfoques, se eligió el modelo más adecuado para el prototipo del asistente de concentración. Entre los modelos evaluados, el que ha demostrado ser el más efectivo en la práctica es el del entrenamiento 5: YOLOv8 200 épocas con poco augmentation y otro enfoque.

Este modelo se entrenó con una estrategia de etiquetado diferente, que ampliaba las áreas de interés para incluir desde los ojos hasta la boca. A diferencia de los enfoques anteriores que emplearon todas las técnicas de *augmentation*, este entrenamiento las utilizó de manera moderada. A pesar de utilizar todas las técnicas de *augmentation* en otras fases, se encontró que el modelo con menos y un enfoque de etiquetado más específico resultó ser el más efectivo en la práctica.

El modelo seleccionado muestra un rendimiento sobresaliente durante la evaluación práctica, detectando las tres clases con una confianza por lo menos del 90%. Esto confirma que la combinación de un entrenamiento más prolongado, el ajuste fino del etiquetado y una cantidad controlada de *augmentation* proporciona mejor precisión y robustez. Como resultado, el modelo entrenado con YOLOv8 se ha identificado como el candidato más adecuado para ser implementado en el prototipo del asistente de concentración, ofreciendo el mejor rendimiento en las pruebas prácticas, se observa esto en la Ilustración 37.

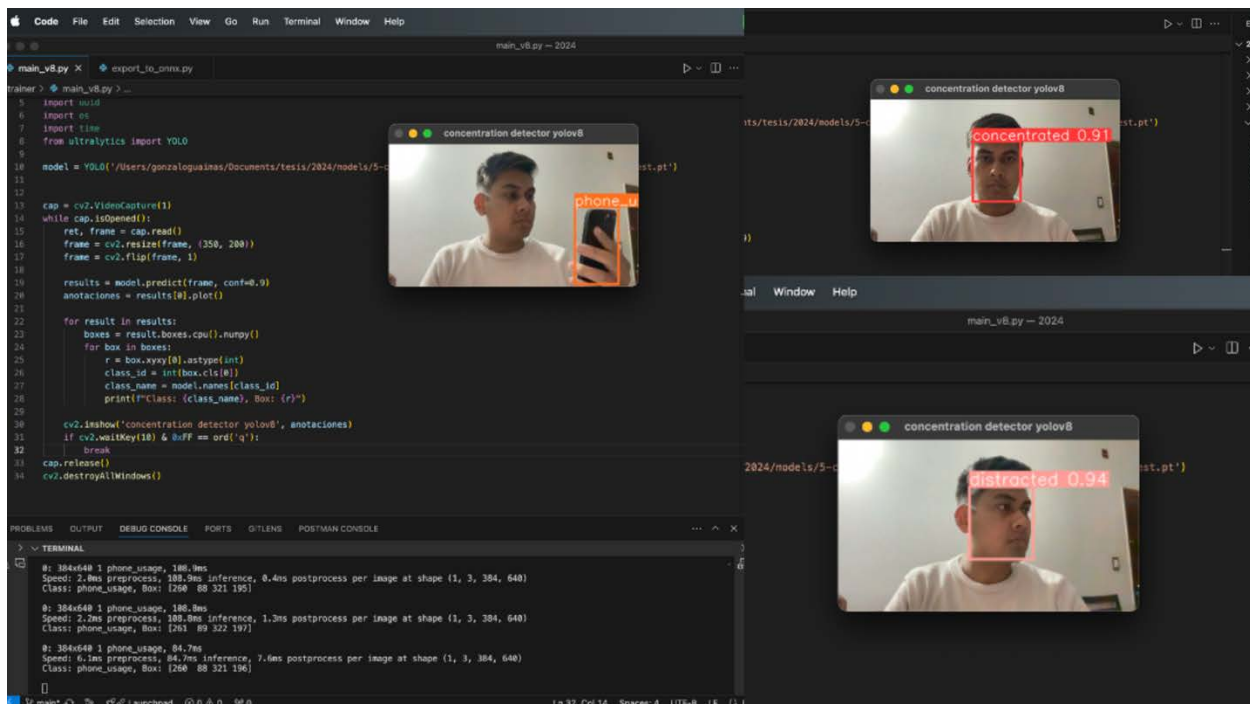


Ilustración 37: Evaluación práctica del modelo con poca *augmentation* y diferente etiquetado

5.3 Etapa 2: Desarrollo del aplicativo web

La última etapa de la metodología CRISP-DM es el despliegue, es en donde se definen las estrategias para su implementación y se lleva a cabo la misma. Es por eso por lo que, en esta sección, se explica brevemente el desarrollo de la aplicación web con la implementación del modelo clasificador de objetos. En esta sección no se busca detallar específicamente cómo fue el desarrollo con la metodología iterativa e incremental, ya que el aspecto principal del proyecto es el entrenamiento del modelo. La metodología iterativa e incremental permitió construir el sistema

en ciclos repetitivos, agregando funcionalidades progresivamente, permitiendo revisiones y mejoras en cada una.

Se decide desarrollar la aplicación como una página web por varias razones. En primer lugar, evita la necesidad de instalar software en los dispositivos, lo que facilita el acceso y uso inmediato por parte de los usuarios. Además garantiza compatibilidad con múltiples dispositivos y sistemas operativos. Asimismo, el desarrollo web permite actualizaciones sin requerir la intervención de los usuarios. Aunque también se podría haber optado por desarrollar una aplicación móvil o de escritorio, se escogió esta opción debido a la experiencia propia del autor en el desarrollo web, lo que asegura un desarrollo rápido del software en cuestión.

La aplicación se desarrolló utilizando Next.js, React y TailwindCSS¹⁹. La misma se compone de una *landing page*²⁰ que presenta el producto y una sección de la aplicación donde se implementa el uso del modelo entrenado para detectar los estados de concentración. También en esa subpágina se incluye un reloj que simula la técnica Pomodoro y un gráfico con analíticas que muestra los estados de concentración detectados en tiempo real.

5.3.1 Incompatibilidad entre Python y JavaScript

El modelo de detección de estados de concentración fué entrenado utilizando PyTorch, junto con la librería Ultralytics que contiene el algoritmo YOLOv8, resultando en un archivo de modelo con extensión .pt. Este formato es específico de PyTorch para utilizarlo en el lenguaje de Python, y no es directamente compatible con JavaScript, el lenguaje que los navegadores web pueden interpretar y con el cual se desarrollará la aplicación. Esto generó una imposibilidad de ejecutar el modelo en este entorno.

Dado que los navegadores solo entienden JavaScript, y no existen librerías en este lenguaje que puedan leer o ejecutar modelos de PyTorch directamente, se necesitó buscar una solución para esta incompatibilidad.

5.3.2 Solución incompatibilidad entre Python y JavaScript

Para abordar la incompatibilidad entre el formato del modelo PyTorch (.pt) y la ejecución en un entorno JavaScript, se utilizó la librería ONNX (Open Neural Network Exchange). ONNX es un formato de intercambio abierto, diseñado para facilitar la interoperabilidad entre diferentes marcos de aprendizaje automático y herramientas. Al convertir el modelo de PyTorch (.pt) a un modelo ONNX (.onnx), se logra que el modelo sea compatible con entornos JavaScript.

La conversión a ONNX permite que el modelo se ejecute en un entorno compatible con JavaScript mediante el uso de ONNX Runtime, una biblioteca de alto rendimiento para inferencia de modelos ONNX. Esto no solo resuelve la incompatibilidad entre Python y JavaScript, sino que

¹⁹ Framework de CSS de utilidad que permite diseñar interfaces rápidamente mediante clases predefinidas, facilitando la personalización y la creación de estilos sin escribir CSS desde cero.

²⁰ Página web diseñada específicamente para convertir visitantes en leads o clientes, generalmente enfocada en una oferta o producto.

también proporciona un entorno eficiente para ejecutar el modelo en la web. Esta solución garantiza que el modelo entrenado pueda integrarse de manera efectiva en la aplicación.

Ventajas de utilizar ONNX:

1. **Interoperabilidad:** ONNX facilita el uso de modelos entrenados en diferentes frameworks de aprendizaje automático y lenguajes. Al convertir el modelo PyTorch a ONNX, se puede integrar fácilmente con aplicaciones basadas en JavaScript y otros entornos de ejecución.
2. **Eficiencia:** ONNX Runtime está optimizado para ofrecer un rendimiento rápido y eficiente en la inferencia de modelos. Esto es crucial para mantener tiempos de respuesta rápidos en aplicaciones web interactivas.
3. **Portabilidad:** Al usar ONNX, se asegura que el modelo pueda ser ejecutado en diferentes plataformas y dispositivos, lo que aumenta la flexibilidad del despliegue.
4. **Ecosistema y soporte:** ONNX cuenta con un amplio soporte y una comunidad activa, lo que garantiza la disponibilidad de herramientas y recursos para la conversión y ejecución de modelos.

La conversión del modelo y el uso de ONNX Runtime permitió resolver la incompatibilidad entre Python y JavaScript.

5.3.3 Desarrollo sección *Landing page*

La *landing page* de la aplicación está diseñada para ofrecer una introducción completa al producto desde el primer ingreso del usuario. Al igual que otras plataformas web, esta página de bienvenida presenta las características, beneficios y posibilidades que ofrece la aplicación de manera clara y atractiva. La intención es que el usuario entienda rápidamente el valor de la aplicación y cómo puede mejorar su concentración y productividad.



Ilustración 38: Sección principal *Landing page*

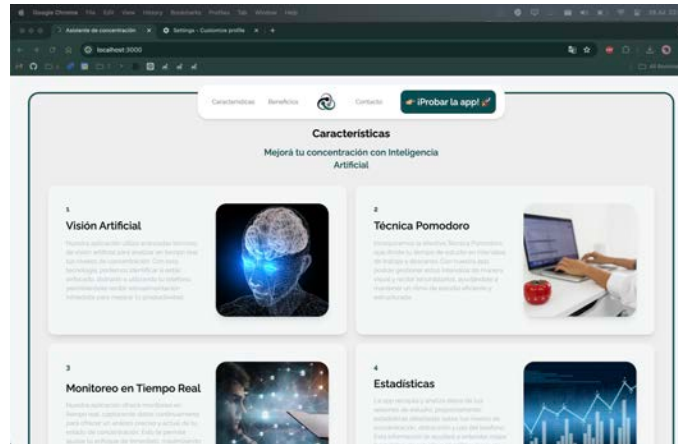


Ilustración 39: Sección características de la app

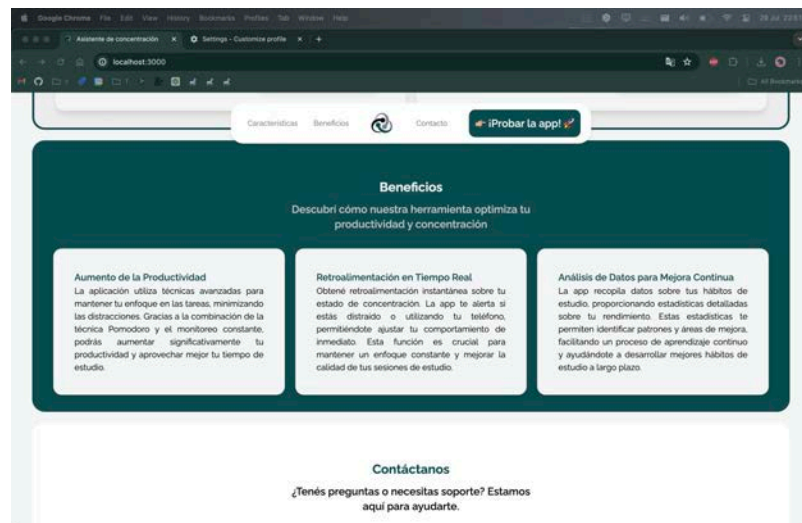


Ilustración 40: Sección beneficios de la app

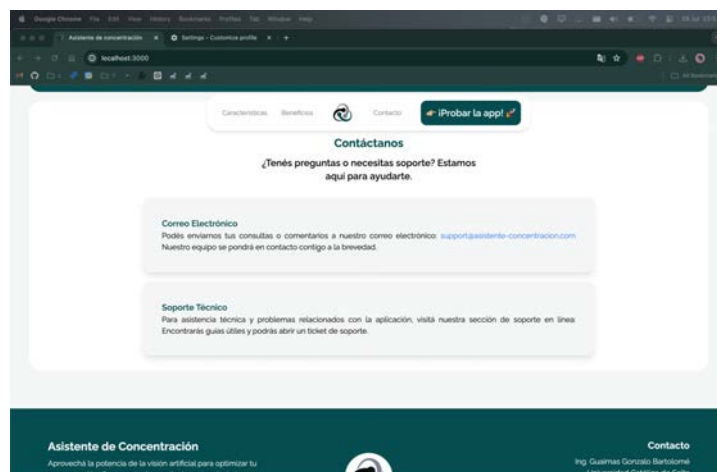


Ilustración 41: Sección contacto

5.3.5 Desarrollo sección Aplicativo – Asistente de Concentración

Dado que el desarrollo del aplicativo es una demostración, no se requiere ningún tipo de autenticación, para ingresar a la aplicación principal, esta decisión se tomó para disminuir el tiempo de desarrollo.

Al ingresar, los usuarios encuentran una interfaz dividida en dos secciones principales, diseñadas para proporcionar una experiencia intuitiva.

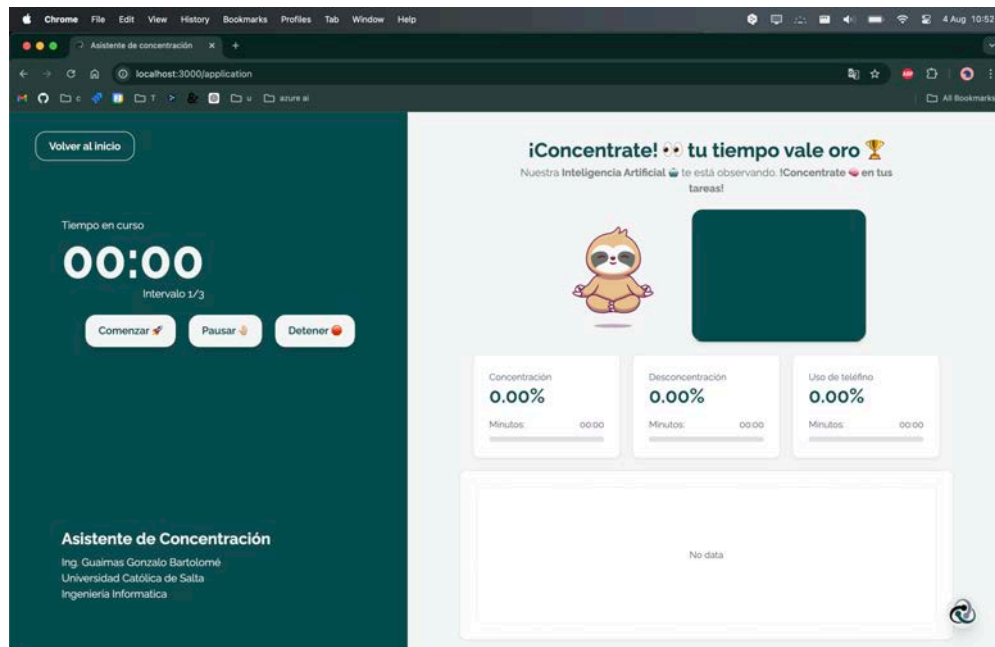


Ilustración 42: Asistente de concentración

Del lado izquierdo, se ubica un reloj que simula el conteo de la técnica Pomodoro, una metodología para mejorar la productividad mediante intervalos de trabajo y descanso. Este reloj permite a los estudiantes seguir de manera visual sus ciclos, ayudando a mantener un ritmo de estudio.

Del lado derecho, se presenta la retroalimentación en tiempo real sobre el estado de concentración del estudiante. Esta sección muestra tanto una imagen en vivo capturada por la cámara del dispositivo como un análisis en tiempo real. El sistema identifica y muestra el porcentaje de tiempo que el usuario ha pasado en distintos estados: concentrado, desconcentrado y usando el teléfono. Además de manera gráfica se ilustran esos datos con animaciones, la de uso del teléfono permite llamar la atención al estudiante para que el mismo se auto discipline.

5.3.6 Desarrollo lógico del reloj de simulación Pomodoro

La lógica del reloj de simulación Pomodoro se implementó de manera sencilla para cumplir con los objetivos de la demostración. El reloj muestra el progreso del tiempo de la sesión de estudio. El sistema cuenta con tres botones principales: uno para comenzar el ciclo, uno para pausar y otro para detenerlo por completo. Cabe aclarar que el encendido de la cámara sólo se realiza cuando el usuario está en un intervalo de trabajo y no así cuando está en descanso o aún no haya iniciado la actividad, con esto se evita el uso del recurso de la cámara.

Se establecieron ciclos de actividad de 10 minutos, seguidos por 2 minutos de descanso. Esta configuración permite simular la técnica Pomodoro en un formato corto, con el propósito de realizar demostraciones y pruebas rápidas.

5.3.7 Problema de rendimiento en la implementación del modelo

La documentación de ONNX proporciona un ejemplo de implementación del runtime en Next.js. Sin embargo, dicho ejemplo utiliza una versión de Next.js desactualizada. En respuesta a esta limitación, se realizó una búsqueda de alternativas en GitHub sobre repositorios²¹ de otros desarrolladores que habían implementado ONNX en versiones más recientes de Next.js. Se logró extraer algunas líneas de código de estos repositorios, facilitando así el proceso de desarrollo.

Con la implementación de ONNX en la web, surge una problemática. Funcionaba de manera adecuada para clasificar imágenes individuales subidas como archivos, pero en la propia implementación procesa las imágenes cada segundo de los fragmentos que captura el video de la webcam. Por lo que para clasificar una imagen a la vez no existían problemas de rendimiento, pero con imágenes cada un segundo sí.

En concreto, el mal rendimiento se detectó debido a que la interfaz se bloqueaba y el contador de segundos se retrasaba en consecuencia, respecto al tiempo real. Para mitigar este problema, se exploraron soluciones alternativas, como el uso de *service workers*²². Finalmente, se optó por una solución factible: dividir la aplicación en una arquitectura de back-end y front-end.

5.3.8 Desarrollo back-end

El back-end del sistema se desarrolló utilizando Nest.js, un framework robusto y modular para la creación de aplicaciones de servidor. En esta aplicación, se implementó un *endpoint*²³ dedicado a recibir imágenes. Este *endpoint* procesa las imágenes utilizando el *runtime* de ONNX y devuelve la clasificación correspondiente al front-end. Para optimizar el rendimiento, la carga

²¹ Espacio de almacenamiento en línea donde se guardan archivos y el historial de cambios de un proyecto de software, permitiendo la colaboración, control de versiones y seguimiento de desarrollo.

²² Script que permite a una web funcionar sin conexión, gestionar caché y enviar notificaciones en segundo plano.

²³ URL específica en una API donde se accede a un recurso o servicio, permitiendo la comunicación entre el cliente y el servidor.

del modelo se realiza durante la configuración inicial del proyecto Nest.js, evitando así la recarga en cada solicitud.

Entonces, en la solución adoptada, en cada segundo se captura una imagen del video en tiempo real de la webcam y se la envía al back-end. En el lado del servidor, se procesa la imagen utilizando el modelo ONNX y devuelve la clasificación al front-end. Esta separación de responsabilidades evita el bloqueo del web app, permitiendo que continúe funcionando de manera fluida.

5.3.9 Arquitectura y diseño lógico final utilizado

Para ofrecer una visión clara y detallada de la arquitectura y la lógica implementada en el sistema, se presenta a continuación un diagrama que ilustra su funcionamiento. Esta arquitectura se ha diseñado para abordar problemas de rendimiento y asegurar una experiencia de usuario fluida, especialmente en escenarios que requieren procesamiento intensivo, como la clasificación de imágenes en tiempo real.

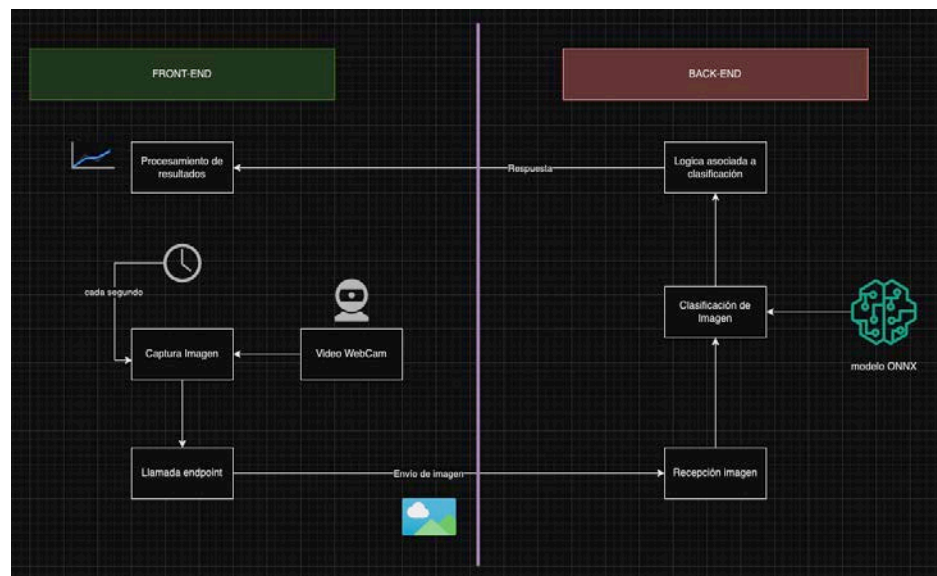


Ilustración 43: Arquitectura del sistema

La arquitectura está dividida en dos proyectos: el Front-End y el Back-End, cada uno con responsabilidades definidas. (Anexo 2)

Front-End

1. **Video WebCam:** La cámara web captura un flujo continuo de video en tiempo real. Cada segundo, una imagen es extraída de este, para su posterior procesamiento.
2. **Captura de Imagen:** Se encarga de tomar una imagen instantánea cada segundo del video capturado por la cámara web. Este proceso asegura una tasa constante de datos para ser analizados y clasificados.

3. **Llamada a Endpoint:** La imagen capturada es enviada al endpoint del back-end. Esta comunicación se realiza de manera asíncrona para no interrumpir la interfaz del usuario. La imagen se envía para su procesamiento por el modelo de clasificación.
4. **Procesamiento de Resultados:** Una vez recibida la clasificación de la imagen desde el back-end, el front-end se encarga de procesar estos resultados y mostrarlos al usuario final en forma de porcentajes y gráficos. Además, se realizan cálculos sobre los estados de concentración, desconcentración y uso del teléfono.

Back-End

1. **Recepción de Imagen:** El back-end, recibe las imágenes a través de un endpoint. Este es responsable de preparar el dato y ejecutar el proceso de clasificación.
2. **Clasificación de Imagen:** Utilizando un modelo ONNX, el back-end clasifica las imágenes recibidas. Este modelo se carga una vez durante la configuración inicial del servidor, optimizando el rendimiento al evitar recargas innecesarias en cada solicitud. Además, se asigna el tamaño de imagen a recibir, la confianza mínima para clasificar, entre otras configuraciones.
3. **Respuesta:** Los resultados de la clasificación se envían de vuelta al front-end. Esta respuesta incluye la etiqueta de clasificación y su confianza.

La arquitectura dividida en front-end y back-end es una solución para manejar la carga de procesamiento, evitando que la interfaz del usuario se bloquee debido al procesamiento intensivo de imágenes cada segundo. Esta separación permite mantener una experiencia de usuario fluida, mientras que el procesamiento pesado es realizado en el servidor. Para realizar la ejecución y prueba del sistema ver (Anexo 3).

6 RESULTADOS

Tras haber detallado el desarrollo del proyecto, tanto del entrenamiento de los modelos como el del aplicativo, es momento de presentar la funcionalidad integral de la aplicación para su comprensión final. La revisión final de todo el proceso llevado a cabo en el proyecto es una tarea más de la etapa del despliegue según la metodología CRISP-DM.

A continuación, se presentan diferentes capturas de pantalla del aplicativo. Estas imágenes ilustran cómo el modelo se ejecuta en el back-end, mientras la interfaz de usuario consume las respuestas del mismo en tiempo real. Además, se muestra cómo los resultados son presentados al usuario de manera gráfica y porcentual, facilitando así la interpretación de los datos. Para visualizar en video la ejecución del sistema ver (Anexo 4).

En la ilustración 44, se observa la interfaz del aplicativo ejecutándose y detectando el estado de concentración del usuario a través de la webcam. En esta primera observación se aprecia que el 100% del tiempo, el estudiante estuvo concentrado, así como lo marca la tarjeta y la gráfica en color verde.

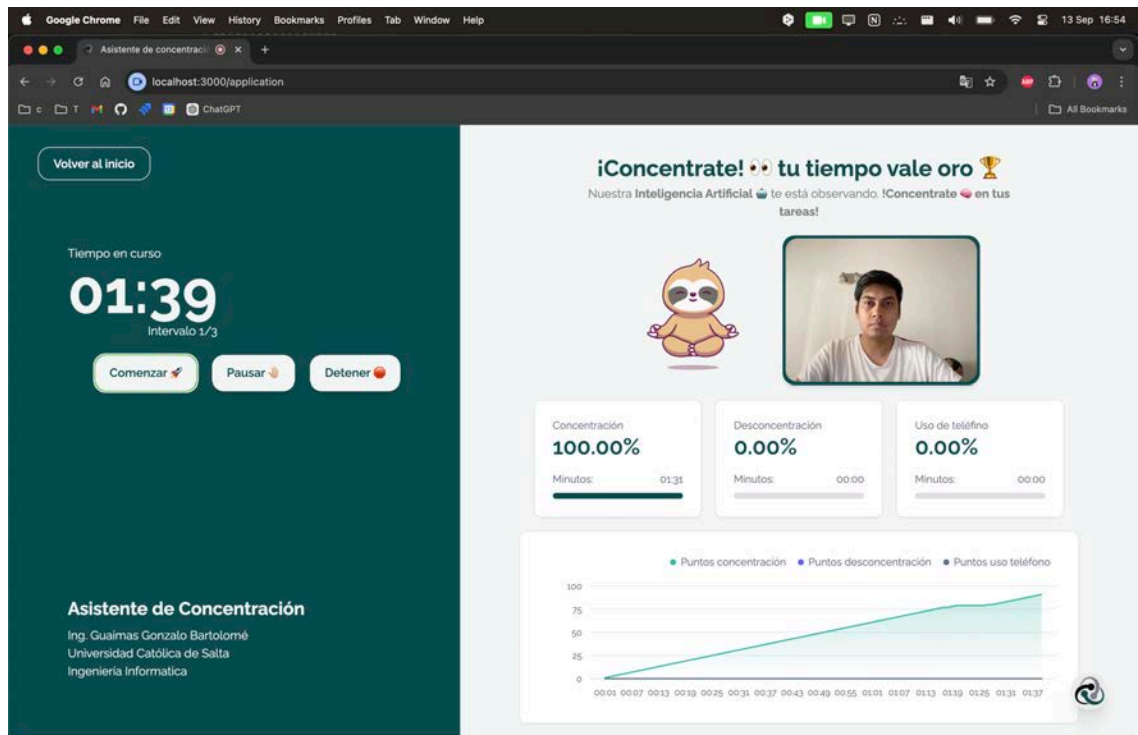


Ilustración 44: Reconocimiento de estado "Concentrado" en aplicativo

Por otro lado, tal como se explicó la arquitectura del sistema, el back-end cada segundo recibe fragmentos de imagen del video capturado por la webcam, los cuales son procesados por el modelo. Este retorna resultados de cuál es el estado de concentración que predice y su confianza con la que lo hace.

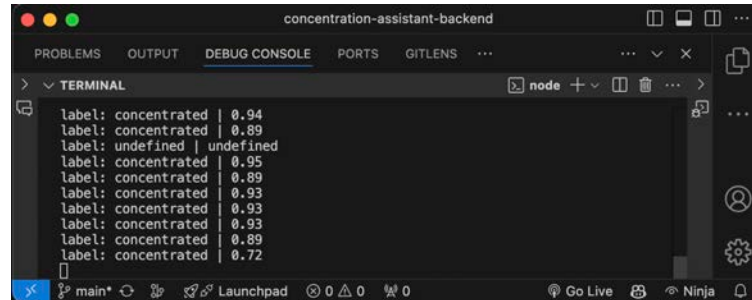


Ilustración 45: Respuesta de estado "Concentrado" en back-end

El valor *undefined* que se observa en ilustración 45 significa que el modelo fue incapaz de reconocer un estado de concentración en la imagen, por lo cual se considera un valor vacío.

Después de unos minutos se realizó otra captura de pantalla luego de haber estado unos segundos mirando de perfil, lo que al modelo se lo entrenó para reconocerlo como estado de desconcentración. De esta forma se aprecia en la gráfica una nueva línea violeta que representa el estado de desconcentración, acompañado del porcentaje de tiempo que representó ese estado, así también el valor de desconcentración cambió de un 100% a un 84% aproximadamente.

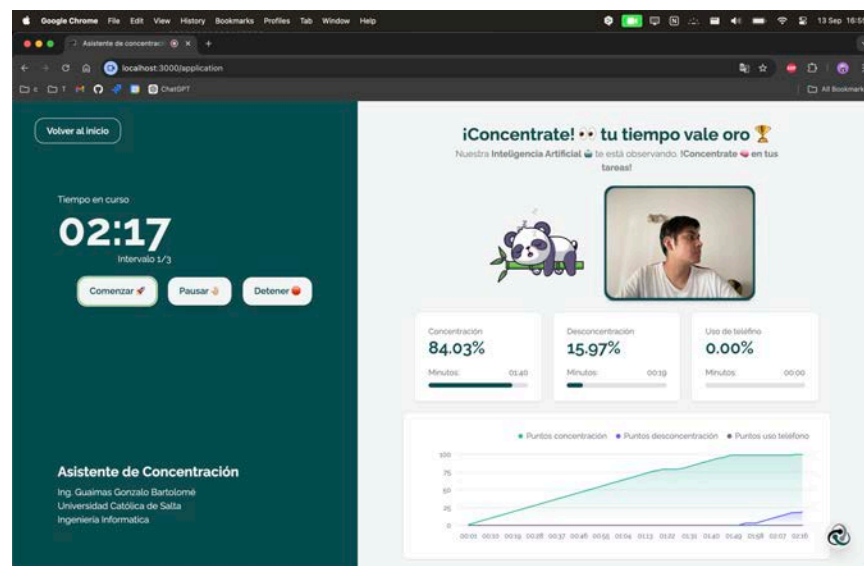


Ilustración 46: Reconocimiento de estado "Desconcentrado" en aplicativo

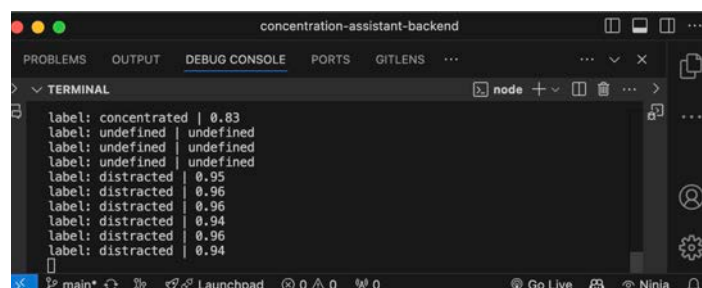


Ilustración 47: Respuesta de estado "Desconcentrado" en back-end

Después de corroborar la detección del estado de desconcentración, en la ilustración 48 se observa en el video de la webcam la presencia del teléfono celular en manos del usuario. El modelo fue entrenado para detectar a este objeto, así se observa en ilustración 49 las respuestas del servidor, clasificando al mismo como “*phone_usage*” con una confianza por encima del 90% en la mayoría de los fragmentos capturados. Además, un símbolo de alarma se visualiza para llamar la atención al estudiante, como así también la gráfica y los porcentajes demuestran el período de tiempo de uso del teléfono.

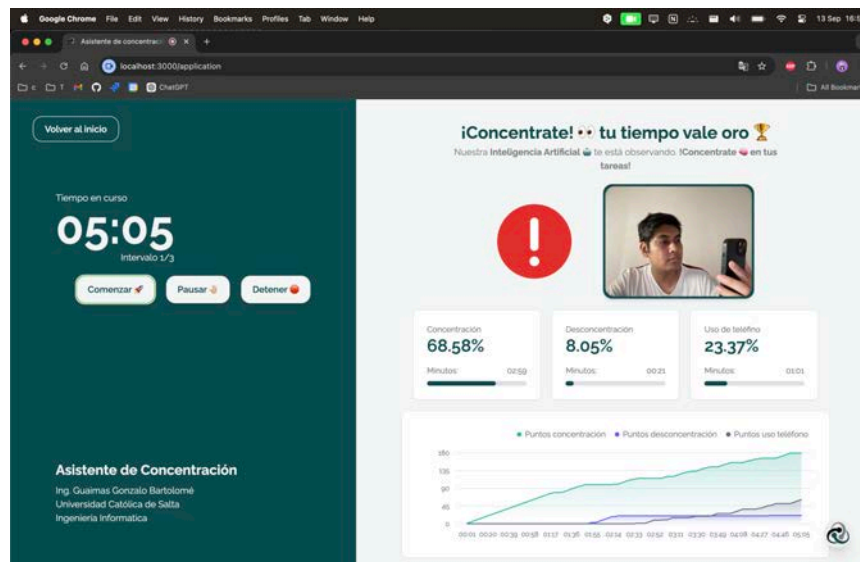


Ilustración 48: Reconocimiento de estado "Uso de teléfono" en aplicativo

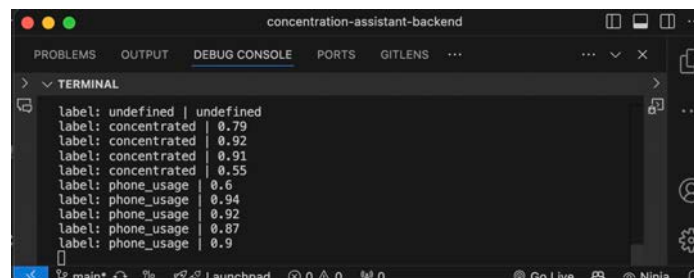


Ilustración 49: Respuesta de estado "Uso de teléfono" en back-end

Entonces, en conclusión, tras cinco minutos de ejecución, el gráfico muestra que el estudiante estuvo "concentrado" la mayor parte del tiempo (curva verde). Sin embargo, a partir de los dos minutos, comenzó a distraerse hasta la finalización de la sesión (curva violeta). Después de los dos minutos y medios, el asistente de concentración detectó, a través del video y el modelo, el uso del teléfono celular (curva gris), lo que activó una señal de alerta visual. Además, el análisis porcentual y temporal indica que el estudiante estuvo concentrado en sus tareas aproximadamente el 68.58% del tiempo total, se desconcentró alrededor del 8.05%, y el 23.37% de la sesión lo pasó usando el teléfono.

De esta manera, al finalizar la sesión, el estudiante puede analizar detalladamente la manera en que distribuyó su tiempo entre la concentración, la distracción y el uso del teléfono. Esta

herramienta proporciona una visión clara y cuantificable de los hábitos de estudio del estudiante, lo que facilita la identificación de patrones de comportamiento que pueden estar afectando negativamente su rendimiento académico. Al contar con datos específicos sobre su capacidad de mantener el enfoque, el usuario puede tomar medidas concretas para mejorar sus hábitos.

Por ejemplo, el análisis de los resultados puede revelar que los últimos minutos de la sesión el estudiante tiende a distraerse más. Además, la detección del uso del teléfono permite al estudiante ser consciente de las distracciones digitales.

El proceso de autoevaluación fomentará una mayor disciplina, ya que el estudiante se vuelve más consciente de sus comportamientos y actitudes durante las sesiones de estudio. Al tomar decisiones informadas, para modificar estos comportamientos, puede mejorar gradualmente su capacidad de concentración, aumentar su eficiencia en el estudio y, en última instancia, mejorar su rendimiento académico. Mediante esta herramienta se puede mejorar entonces el enfoque en la gestión del tiempo y la atención, los cuales son fundamentales para desarrollar habilidades de estudio más efectivas y lograr un mejor desempeño en el ámbito académico.

7 FUTURAS LÍNEAS DE INVESTIGACIÓN

Este proyecto abre oportunidades para futuros desarrollos de herramientas, en el campo de las tecnologías educativas y de la inteligencia artificial aplicada a la mejora de la concentración de alumnos. Una posible área de investigación es la optimización y robustez del modelo de IA para reconocer a todos los estados posibles de concentración y desconcentración de las personas. Por otro lado, al ser un modelo más complejo que del prototipo presentado, se debería de investigar, cómo realizar la ejecución del modelo sobre servidores y que el costo por computación no influya sobre la factibilidad financiera en un principio.

Otra línea de investigación interesante, sería la integración de otro modelo, que no sólo detecte la concentración, sino que también reconozca otros datos, como la detección de emociones mediante reconocimiento facial. Esto permitiría una evaluación más profunda del comportamiento del usuario a lo largo de su sesión de estudio, por ejemplo se podría analizar si el estudiante se aburre en los primeros minutos, o si se lo percibe alegre, enojado o frustrado, entre otras emociones claves para realizar una mejor retroalimentación al estudiante.

Sería valioso analizar el impacto a largo plazo de la aplicación en el rendimiento académico de los usuarios, observando si la reducción de distracciones digitales y la mejora de la concentración, generan cambios significativos en el rendimiento académico del estudiante.

Por otro lado, se podría realizar una investigación tecnológica adicional, sobre cómo enfrentar a las notificaciones y mensajes emergentes por el uso de redes sociales en el navegador web, ya que el estudiante puede acceder y el prototipo presentado no es capaz de detectar esto mismo. La integración debe monitorear en tiempo real el uso de redes sociales o aplicaciones no relacionadas al ámbito académico, para impedir o alarmar el acceso a estas que conllevan a la distracción que se busca eliminar. Esta posible integración logrará potenciar la herramienta haciéndola más eficiente.

8 CONCLUSIÓN

El desarrollo de esta aplicación, diseñada para disminuir las llamadas distracciones digitales y mejorar la concentración de los estudiantes universitarios, ha demostrado ser una solución a una problemática contemporánea creciente. La implementación de una arquitectura que separa el procesamiento de la clasificación de imágenes en un servidor y la presentación de resultados en un aplicativo web ha sido clave para asegurar un buen rendimiento, una experiencia de usuario fluida y una aplicación de demostración al alcance de todos.

Al utilizar herramientas de inteligencia artificial y la técnica Pomodoro, se logró no solo identificar los estados de concentración, desconcentración y uso del celular del estudiante, sino también proporcionar retroalimentación en tiempo real, lo cual es fundamental para mejorar la disciplina del estudiante universitario frente a sus tareas. El proceso de desarrollo no estuvo exento de desafíos y problemáticas, especialmente en la implementación de modelos que son grandes consumidores de recursos computacionales. Sin embargo, la superación de estos obstáculos a través del uso de recursos disponibles demuestra la capacidad de innovación que hoy en día puede realizarse con las nuevas tecnologías, en especial las relacionadas a la inteligencia artificial.

La aplicación no sólo cumple con su objetivo de mejorar la concentración del estudiante y, por ende, el rendimiento académico, sino que también ofrece un nuevo método novedoso y práctico para abordar las distracciones generadas por las redes sociales que se encuentran en los dispositivos celulares. Este proyecto establece una base para futuras investigaciones y desarrollos en el campo de las herramientas de apoyo académico mediante el uso de tecnologías avanzadas, reafirmando el impacto positivo de las TIC cuando son utilizadas estratégicamente para mejorar la calidad y el desempeño académico.

BIBLIOGRAFÍA

Aziz, L., Salam, M. S. B. H., Sheikh, U. U., & Ayub, S. (2020). Exploring deep learning-based architecture, strategies, applications and current trends in generic object detection: A comprehensive review. En *IEEE Access* (Vol. 8, pp. 170461–170495). Institute of Electrical and Electronics Engineers Inc. <https://doi.org/10.1109/ACCESS.2020.3021508>

Cirillo, F., & Info, B. (2020). *The Pomodoro Technique (The Pomodoro)*.

Conceptos básicos de ayuda de CRISP-DM - Documentación de IBM. (s/f). Recuperado el 29 de junio de 2024, de <https://www.ibm.com/docs/es/spss-modeler/saas?topic=dm-crisp-help-overview>

Documentation | NestJS - A progressive Node.js framework. (s/f). Recuperado el 4 de julio de 2024, de <https://docs.nestjs.com/>

Garate-Osuna, L. E., Nava-Quevedo, A., & Pacheco-Millán, C. (2015). IMPACTO DE LAS REDES SOCIALES EN EL COMPORTAMIENTO DE ESTUDIANTES DE NIVEL SUPERIOR IMPACT OF SOCIAL NETWORKS ON THE BEHAVIOR OF HIGHER LEVEL STUDENTS. *RITI Journal*, 3.

Informe de la UNESCO recomienda prohibir el uso de celulares en las escuelas - Infobae. (s/f). Recuperado el 29 de junio de 2024, de <https://www.infobae.com/sociedad/2023/07/26/informe-de-la-unesco-recomienda-prohibir-el-uso-de-celulares-en-las-escuelas/>

La distracción con el celular y la indisciplina afectan el aprendizaje en Argentina - Infobae. (s/f). Recuperado el 29 de junio de 2024, de <https://www.infobae.com/educacion/2023/12/07/la-distraccion-con-el-celular-y-la-indisciplina-afectan-el-aprendizaje-en-argentina/>

Modelos de ONNX: optimización de la inferencia - Azure Machine Learning | Microsoft Learn. (s/f). Recuperado el 4 de julio de 2024, de <https://learn.microsoft.com/es-es/azure/machine-learning/concept-onnx?view=azureml-api-2>

Node.js — About Node.js®. (s/f). Recuperado el 4 de julio de 2024, de <https://nodejs.org/en/about>

Pete Chapman (NCR), Julian Clinton (SPSS), Randy Kerber (NCR), Thomas Khabaza (SPSS), Thomas Reinartz (DaimlerChrysler), Colin Shearer (SPSS), & Rüdiger Wirth (DaimlerChrysler). (1999). *CRISP-DM 1.0 Step-by-step data mining guide*. DaimlerChrysler.

¿Qué es Deep Learning? | IBM. (s/f). Recuperado el 30 de junio de 2024, de <https://www.ibm.com/es-es/topics/deep-learning>

¿Qué es el aprendizaje supervisado? | IBM. (s/f). Recuperado el 29 de junio de 2024, de <https://www.ibm.com/es-es/topics/supervised-learning>

React Foundations: About React and Next.js | Next.js. (s/f). Recuperado el 4 de julio de 2024, de <https://nextjs.org/learn/react-foundations/what-is-react-and-nextjs>

Rosa María Muñoz Franco, Adoración Díaz López, & Juan Antonio Sabariego García. (s/f). *IMPACTO DE LAS REDES SOCIALES EN EL RENDIMIENTO ACADÉMICO DE LOS ADOLESCENTES*. <https://orcid.org/0000-0002-8841-3701>

Russell, S. J. (Stuart J., Norvig, Peter., Corchado Rodríguez, J. Manuel., & Joyanes Aguilar, Luis. (2004). *Inteligencia artificial : un enfoque moderno*. Pearson Prentice Hall.

Sommerville, I., Campos Olguín, V., & Fuenlabrada Velázquez, S. (2011). *Ingeniería de software*. Pearson Educación de México.

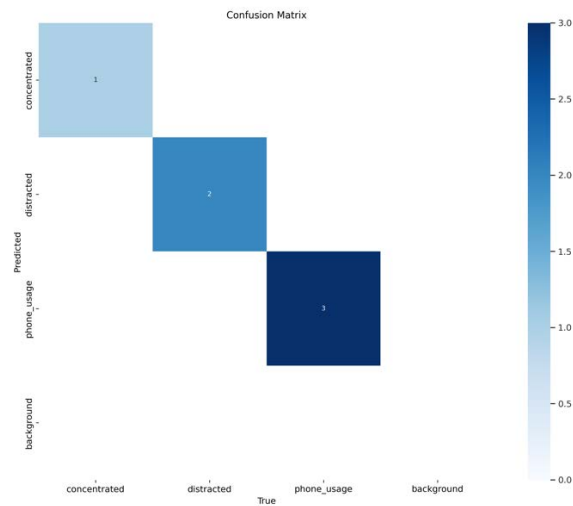
Witten Ian, Eibe Frank, & Hall Mark. (2011). *Data Mining Third Edition*.

YOLO Object Detection Explained: A Beginner's Guide | Encord. (s/f). Recuperado el 3 de julio de 2024, de <https://encord.com/blog/yolo-object-detection-guide/>

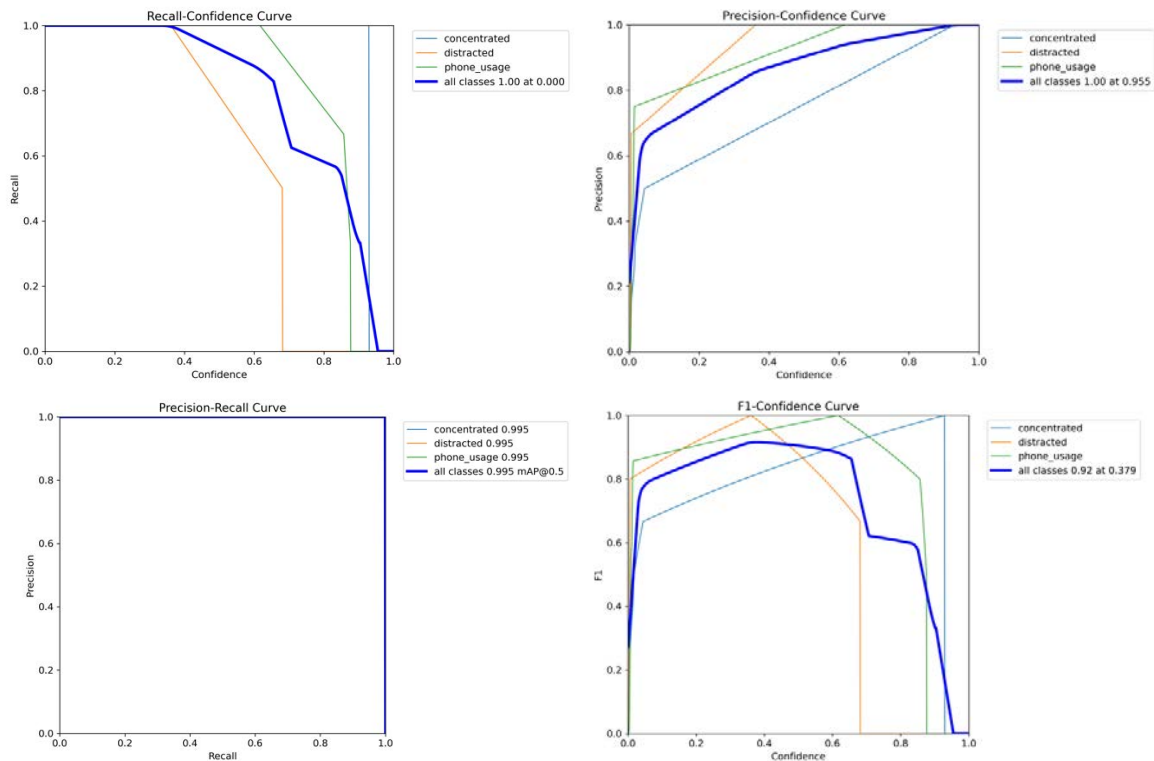
ANEXOS

Anexo 1: Datos de rendimiento del modelo 1

Gráfica de Matriz de Confusión.



Curvas de recall, precisión y confianza.



Anexo 2: Acceso a repositorios de código

Para poder acceder a los repositorios del presente proyecto, se debe visitar los siguientes enlaces que dirigen a la plataforma de GitHub del autor:

Back-End

<https://github.com/GonzaloGuaimas/concentration-assistant-backend.git>

Front-End

<https://github.com/GonzaloGuaimas/concentration-assistant-nextjs>

Anexo 3: Ejecución del sistema de manera local

Para ejecutar la aplicación localmente, tanto el front-end (Next.js) como el back-end (Nest.js), se necesita seguir los siguientes pasos:

1. Instalación de Node.js

Se puede descargar la versión más reciente desde el siguiente enlace:

<https://nodejs.org/>

2. Instalación de Git

Es necesario contar con Git para clonar los repositorios del proyecto desde GitHub.

<https://git-scm.com/>

3. Clonar los repositorios

Ejecutar los siguientes comandos en la terminal:

- Para el **back-end**:

`git clone https://github.com/GonzaloGuaimas/concentration-assistant-backend.git`

- Para el **front-end**:

`git clone https://github.com/GonzaloGuaimas/concentration-assistant-nextjs.git`

4. Instalar dependencias

Una vez clonados los repositorios, acceder a las carpetas del proyecto y ejecutar el siguiente comando para instalar las dependencias necesarias en cada uno de ellos:

- **Back-end (NestJS):**

`cd concentration-assistant-backend
npm install`

- **Front-end (Next.js):**

```
cd concentration-assistant-nextjs  
npm install
```

5. Ejecución de los servidores

Iniciar el servidor para cada parte del proyecto:

- **Back-end (NestJS):**
npm run start
- **Front-end (Next.js):**
npm run dev

6. Acceder a la aplicación

Una vez que ambos servidores estén ejecutándose, acceder a la aplicación abriendo el navegador y visitando la siguiente URL:

<http://localhost:3000>

De esta manera, se puede ejecutar y probar el sistema de manera local.

Anexo 4: Video demostración de funcionamiento del aplicativo

https://youtu.be/a1dkHE_t3CU