Asistente de concentración impulsado por Inteligencia Artificial para estudiantes universitarios

Artificial intelligence-powered concentration assistant for university students

Gonzalo Guaimas
gonzaloguaimas@gmail.com

Ingeniería en Informática, Facultad de Ingeniería, UCASAL

Resumen

En los últimos años, se observó un fenómeno preocupante entre los jóvenes. Las redes sociales han emergido como una actividad de ocio significativa o mejor dicho de distracción, en particular cuando se trata del momento de estudio de alumnos universitarios. La constante revisión de notificaciones y la búsqueda de actualizaciones se asemejan a un comportamiento adictivo, que consume e interrumpe la concentración del estudiante en las sesiones de estudio. Estos nuevos hábitos impactan en la productividad y el rendimiento académico de los jóvenes estudiantes. En este trabajo se busca desarrollar una aplicación que sirva como herramienta para mejorar la concentración del estudiante en las sesiones de estudio delimitadas por la técnica Pomodoro. Con el fin de detectar los diferentes estados de la concentración del estudiante se hará uso de la inteligencia artificial. Esta misma podrá detectar por medio de la cámara web si el individuo se encuentra concentrado, desconcentrado o utilizando el teléfono celular. De esta manera, se obtendrá un seguimiento de la concentración del estudiante durante la sesión de estudio, al proporcionar una retroalimentación de esta en tiempo real al finalizar. Esta información será de utilidad para el alumno, ya que podrá regular el uso de su dispositivo móvil y servirá de motivación para seguir perfeccionando su concentración y mejorar su rendimiento académico.

Palabras clave: Asistente de concentración, detección de objetos, inteligencia artificial, minería de datos, Redes Neuronales Convolucionales, Python, visión artificial, YOLO

Abstract

In recent years, a worrying phenomenon has been observed among young people. Social networks have emerged as a significant leisure activity and a distraction, particularly when talking about the study time of university students. The constant checking of notifications and searching for updates resembles addictive behavior, consuming and interrupting the student's concentration in their study sessions. These new habits impact the productivity and academic performance of young students. This work seeks to develop an application that serves as a tool to improve a student's concentration in their study sessions marked by the Pomodoro technique. In order to detect the different states of the student's concentration, artificial intelligence will be used; it will be able to detect, by means of a web camera, whether the individual is concentrated, distracted, or using a cell phone. In this way, the student's concentration will be monitored during their study session, providing feedback in real time at the end. This information will be useful for the student since they will be able to regulate the use of their mobile device and will serve as motivation to continue improving their concentration and their academic performance.

Key words: Concentration assistant, object detection, artificial intelligence, data mining, convolutional neural networks, Python, artificial vision, YOLO

Introducción

En los últimos años, ha surgido un problema estudiantil potencial. Es por eso que en este proyecto nos centraremos en los estudiantes universitarios. El surgimiento masivo de las redes sociales ha transformado la dinámica de estudio, al desviar la atención de los estudiantes de las actividades académicas hacia el atractivo mundo de las notificaciones, las actualizaciones y el entretenimiento en línea. Esta alteración en el enfoque y la concentración de los estudiantes durante las sesiones de estudio han planteado un desafío sustancial para el rendimiento académico de la juventud [1]. El presente trabajo surge como respuesta a la necesidad de abordar este problema emergente. El objetivo primordial de esta investigación es desarrollar un prototipo que funcione como una herramienta efectiva a fin de mejorar la concentración de los estudiantes durante los períodos de estudio. Las sesiones de estudio estarán delimitadas por la reconocida técnica Pomodoro.

La raíz del problema se encuentra en la imparable y constante revisión de notificaciones y la búsqueda incesante de actualizaciones, comportamiento que se asemeja a una adicción digital. Con el objetivo abordar esta problemática, se plantea la implementación de la inteligencia artificial y la visión por computadora, que permitirán la detección en tiempo real de los distintos estados de concentración de los estudiantes durante las sesiones de estudio. El sistema tendrá la capacidad de identificar los momentos de concentración, distracción y utilización de dispositivos móviles, de manera que se convertirá en una herramienta valiosa para proporcionar retroalimentación al usuario y mejorar el rendimiento académico.

Este prototipo tiene el fin de ofrecer una solución innovadora que promueva la concentración, la productividad y el rendimiento académico. El resultado de este brindará a los estudiantes una herramienta práctica para optimizar sus esfuerzos académicos en un entorno digitalizado.

Reconocimiento de objetos: La visión artificial

La capacidad de la visión de los humanos nos permite identificar individuos, animales y objetos inanimados con precisión. En el ámbito de la inteligencia artificial y la visión por computadora, se utiliza comúnmente el término “reconocimiento de objetos” para englobar todas estas habilidades. Esto abarca la tarea de identificar la categoría o la clase a la que pertenecen objetos específicos en una imagen, por ejemplo, la detección de una cara, así como la identificación precisa de objetos particulares.

La capacidad de visión no solo se emplea en la identificación de objetos, sino también en la percepción de actividades. Esto incluye la capacidad de reconocer movimientos, como los pasos de un individuo; expresiones faciales, como una sonrisa o una mueca; gestos, como señas con la mano y acciones, como saltar o bailar, entre otros [4].

La minería de datos

La minería de datos es una técnica que se utiliza a fin de procesar grandes conjuntos de datos y descubrir patrones y relaciones ocultas de manera automática o semiautomática [2]. Asimismo, transforma todo en conocimiento práctico, que las empresas utilizan con la finalidad de resolver problemas, aumentar beneficios, entre otras aplicaciones.

CRISP-DM

CRISP-DM es un método probado para orientar los trabajos de minería de datos. Es una metodología que describe fases típicas de un proyecto de minería de datos. Brinda una explicación de cada tarea que se debe llevar a cabo en cada fase y la relación de estas [3]. A continuación, se detallan las fases:

» Comprensión del negocio:

Se identifican los objetivos y el alcance del proyecto, se deben tener en cuenta los problemas a resolver, las limitaciones y el impacto empresarial.

» Comprensión de los datos:

Implica estudiar los datos disponibles, de tal manera que se eviten problemas durante la siguiente fase. Se determinará la calidad de los datos.

» Preparación de los datos:

Es la etapa que lleva más tiempo, dado que se deben de perfeccionar o transformar los datos para utilizar en el modelado. Implica tareas como la limpieza de los datos, la integración y el formateo.

» Modelado de datos:

Mediante un software de minería de datos, se introducen los datos preparados para entrenar el modelo. Se realiza el entrenamiento de modelo, se ajusta y se entrena nuevamente hasta que los resultados sean satisfactorios.

» Evaluación:

Se realiza un análisis de los diferentes resultados y se puede volver a ajustar el modelo.

» Implementación:

Se utiliza el conocimiento para poder compartir resultados a los clientes y recomendaciones.

El aprendizaje automático: *Machine Learning*

El aprendizaje automático o *machine learning* es el proceso que se basa en alimentar una computadora con datos, y esta misma utiliza la técnica de análisis sobre estos datos a fin de aprender a realizar una tarea. El aprendizaje automático puede ser supervisado o no supervisado. El aprendizaje supervisado consiste en aprender una función a partir de ejemplos de sus entradas y salidas, mientras que el no supervisado consiste en aprender a partir de patrones de entradas para los que no especifican los valores de sus salidas.

Esta clasificación es el resultado del aprendizaje de una función de valores discretos. Es un problema del aprendizaje supervisado, el cual se puede atacar con diferentes métodos: árboles de decisión, modelos bayesianos, clasificadores basados en casos, redes neuronales, entre otros [4].

El aprendizaje profundo: *Deep Learning*

El aprendizaje profundo, una rama del aprendizaje automático, se caracteriza por el uso de redes neuronales con tres o más capas. Estas redes neuronales buscan imitar el funcionamiento del cerebro humano, aunque no alcanzan su nivel de capacidad. No obstante, les permiten "aprender" a partir de grandes volúmenes de datos.

El aprendizaje profundo impulsa una amplia gama de servicios y aplicaciones de inteligencia artificial (IA), al automatizar tareas tanto analíticas como físicas sin la intervención humana. Esta tecnología se encuentra en la base de numerosos productos y servicios de uso cotidiano, como asistentes digitales, sistemas de control de televisión por voz y sistemas de detección de fraudes con tarjetas de crédito, así como en tecnologías emergentes, como los vehículos autónomos [5].

Las redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales, conocidas como ConvNets (CNN, por su sigla en inglés) se utilizan mayormente en tareas de clasificación y visión por computadora. Antes de la aparición de las CNN, se dependía de métodos manuales de extracción de características que demandaban mucho tiempo para identificar objetos en imágenes. No obstante, en la actualidad, las redes neuronales convolucionales ofrecen un enfoque más escalable para la clasificación de imágenes y la identificación de objetos, aprovechando al aprovechar principios de álgebra lineal, específicamente la multiplicación de matrices, para detectar patrones en una imagen. Sin embargo, es importante señalar que estas redes pueden requerir recursos computacionales significativos y, en ocasiones, el uso de unidades de procesamiento gráfico (GPU, por su sigla en inglés) a fin de entrenar los modelos [6].

El algoritmo *You Only Look Once* (YOLO)

El algoritmo *You Only Look Once* (YOLO, por su sigla en inglés) es un sistema o un algoritmo de código abierto para la detección de objetos en tiempo real. Este sistema utiliza una única red neuronal convolucional a fin de llevar a cabo la detección de objetos en imágenes. En cuanto a su funcionamiento, la red neuronal segmenta la imagen en distintas regiones y realiza predicciones de cuadros de identificación y sus correspondientes probabilidades para cada región. Luego, las cajas resultantes se ponderan en función de las probabilidades anticipadas. Este algoritmo aprende representaciones que son aplicables de manera general a los objetos, lo que permite obtener una precisión elevada en la detección de objetos en nuevas entradas que difieren del conjunto de datos utilizados en el proceso de entrenamiento [7].

La técnica Pomodoro

La idea de la técnica se basa en períodos de trabajo combinados con descansos breves que optimizan la productividad y la atención, según Francesco Cirillo en su libro La técnica Pomodoro. Su principal fundamento consiste en el hecho de que la mente humana es más efectiva en intervalos limitados de tiempo. Esta técnica sugiere sesiones de trabajo de veinticinco minutos, seguidas de un breve descanso. Este enfoque aprovecha la psicología cognitiva, al reconocer los límites naturales de la atención y la importancia de las pausas regulares para mantener un rendimiento adecuado [8].

1. Desarrollo

1.1. Construcción del detector de estados

La construcción del modelo predictivo de reconocimiento es un componente crítico de nuestra aplicación, ya que sin él sería imposible poder detectar y clasificar el estado de concentración del estudiante, y el prototipo utilizaría la técnica Pomodoro sin más. Por otra parte, ya existen aplicaciones de este tipo. Una de las tareas de la minería de datos es la clasificación. En este caso, gracias a las imágenes en las cuales se detectan objetos, o mejor dicho, estados, que por medio de un conjunto de imágenes preclasificadas se construyó el modelo de detección de imágenes. El algoritmo que se utilizará es una red neuronal convolucional llamada YOLO. A continuación se describen las diferentes actividades

realizadas a fin de llegar al entrenamiento del modelo a través de la metodología CRISP-DM.

1.2. Comprensión del negocio

El modelo debe de tener la capacidad de predecir diversos estados del estudiante, como así también objetos, entre estos se incluye:

1. Concentración**:** Se refiere a la detección de un rostro en la imagen que esté mirando directamente hacia el frente, sin desviarse hacia los lados o hacia abajo. Esto implica una atención plena y un enfoque en el punto central de la imagen.
2. Distracción**:** Este escenario se da cuando el rostro en la imagen se encuentra mirando hacia los lados o hacia abajo. En esta situación, la atención y el enfoque del individuo se desvían de la vista frontal.
3. Utilización del teléfono celular: El modelo también debe ser capaz de identificar la presencia de un teléfono celular como objeto en la imagen. Esto es relevante para detectar si una persona está utilizando un dispositivo móvil durante la captura de la imagen.

En la etapa de comprensión de datos se realizó, en primera instancia, la recopilación de datos iniciales, en nuestro caso la captura de imágenes en donde se aprecian los diferentes estados y los objetos a detectar.

Se recolectaron las imágenes de forma masiva mediante un *script* escrito en Python y ejecutado en un *notebook* de Jupyter, que implementó la librería openCv. El algoritmo ejecuta la cámara web a fin de capturar veinte imágenes cada tres segundos por cada etiqueta, que en este caso es "concentrado" (*concentrated*), "desconcentrado" (*distracted*) y "utilización de teléfono celular" (*phone_usage*). El nombre con el que se guardaron las imágenes es (etiqueta).(*randomID*).jpg, en donde *randomID* es una cadena de texto aleatoria y única. Las imágenes se almacenaron en una subcarpeta del directorio *data/images* en específico en la carpeta *data/images/train,* que a su vez contiene otra carpeta paralela *data/images/val*, en la cual se almacenaron cuatro imágenes de cada tipo y el resto en la carpeta de *train* (entrenamiento). Cabe aclarar que colocarle este nombre de etiqueta no implica que ya están clasificadas y listas para ser entrenadas por la red neuronal YOLO, debido a que ese es un paso posterior.

En paso anterior solo se obtuvo el conjunto de datos de imágenes. Sin embargo, no estaban etiquetadas. En cuanto a esta tarea, que según CRISP-DM es la de preparación de los datos, se realizó de nuevo ese paso. Con el objetivo de etiquetar las imágenes se necesitó utilizar una herramienta llamada LabelImg, la cual proporciona la posibilidad de marcar la zona de interés en las imágenes, que contienen el objeto que se quiere etiquetar. Además, el software permite realizar estas anotaciones en formato YOLO, el cual es el formato necesario para soportar el algoritmo. Estas anotaciones se almacenarán en una carpeta paralela a *data/images*, la cual se denomina *data/labels/train,* y otras cuatro de cada tipo en la carpeta *data/labels/val*.

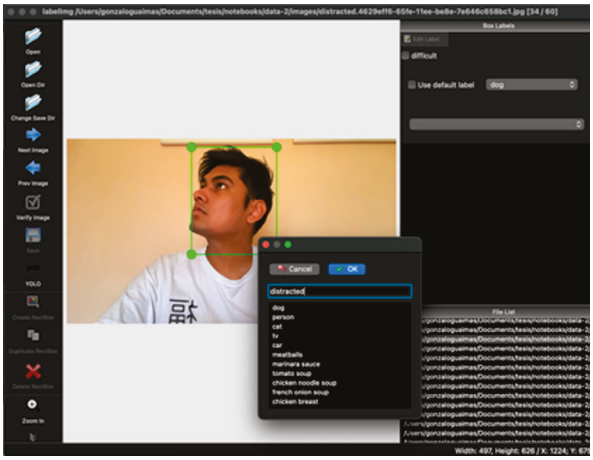

Figura 1: Etiquetado de rasgo facial en estado distraído

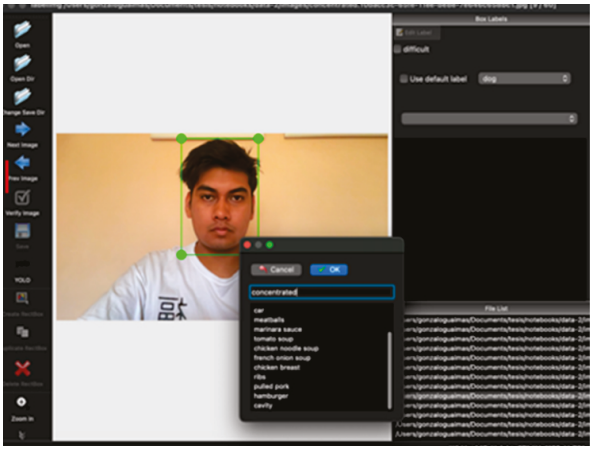

Figura 2: Etiquetado de rasgo facial en estado concentrado

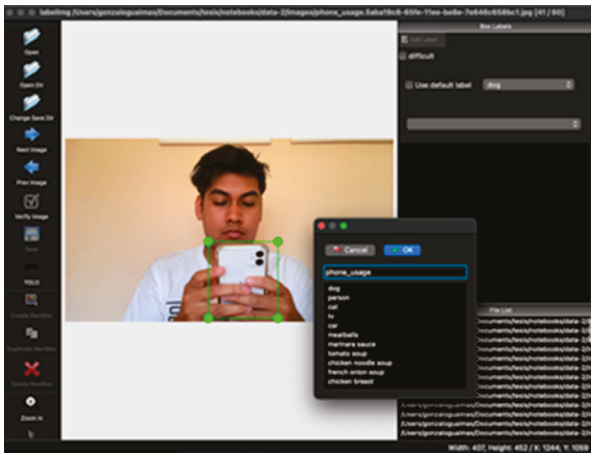

Figura 3: Etiquetado de objeto teléfono celular

Una vez etiquetado todo el conjunto de datos, es el momento de entrenar el modelo mediante el algoritmo YOLO, en este caso la versión 5. Se eligió esta versión debido a que se lo considera un algoritmo robusto con buenos resultados relacionados con la detección de objetos. Estas líneas de código se ejecutaron dentro del entorno de Google Colaboratory debido a la gran capacidad de cómputo que ofrece. El modelo se entrenó cuatro veces. Durante cada entrenamiento se utilizó el mejor peso del anterior a fin de obtener mejores resultados en el modelo.

Considerando el primer entrenamiento se utilizó el peso predeterminado en YOLO del archivo *yolov5.pt*.

```
!python train.py --img 640 --batch 4 --epochs 300 --data dataset.yaml --weights yolov5s.pt --cache
```

Cada uno de estos parámetro significa lo siguiente:

» *python train.py*: Ejecuta el *script* de la librería YOLOv5 preparado para el entrenamiento de un modelo.
» *–img 640*: Especifica el tamaño de las imágenes de entrada durante el entrenamiento, las imágenes se redimensionan a un tamaño de 640x640 píxeles.
» *–batch 4*: Indica el tamaño del lote utilizado durante el entrenamiento. Se refiere al número de entrenamientos que se utiliza en cada paso de optimización.
» *–epochs 300*: Indica el número de épocas de entrenamiento. Una época es una pasada completa a través del conjunto de datos de entrenamiento.
» *–data dataset.yaml*: Este archivo contiene la configuración YAML, que contiene información sobre el conjunto de datos, así como también clases de objetos a detectar.
» *–weights yolov5s.pt*: En cuanto al primer entrenamiento, se utilizaron los pesos predeterminados, luego con cada entrenamiento genera un archivo con el mejor peso encontrado, el cual se utiliza en el siguiente entrenamiento.
» *–cache*: Este parámetro permite habilitar el almacenamiento en caché. Esto significa que los datos se almacenan en caché, en la memoria con el objetivo de acelerar el acceso a ellos durante el entrenamiento.

Teniendo en cuenta los posteriores entrenamientos, se utilizó la siguiente línea de código hasta el cuarto entrenamiento:

```
!python train.py --img 640 --batch 4 --epochs 300 --data dataset.yaml --weights best.pt --cache
```

Luego del cuarto entrenamiento, ya se consideró que se contaba con un modelo óptimo y listo para utilizarse. Sin embargo, con el fin de corroborar, se llevó a cabo un análisis de la matriz de confusión y así asegurarse de la efectividad de este.

Con el fin de realizar la evaluación del modelo, la librería yolov5 nos proporcionó la matriz de confusión, con la que se pudo determinar qué error se cometía en el modelo. El algoritmo usa una confianza de 0,25 y un límite de unión de 0,45, por lo que, con el propósito de clasificar un objeto, debe de tener un 50 % de probabilidad de incluirse en una clase determinada.

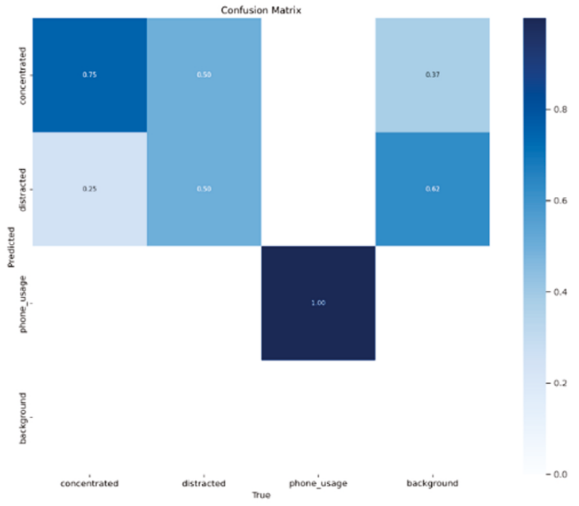

Figura 4: Matriz de confusión del modelo

Se pudo observar que en el 75 % de los casos el modelo se clasificó de manera correcta el estado concentrado. En el 50 % de casos se reconoció un rostro como distraído, mientras que en el otro 50 % se equivocó y lo clasificó como concentrado.

Considerando el caso de utilización de teléfono celular, en el 100 % de los casos se pudo reconocer de manera correcta el uso de este. Por lo tanto, a partir de estos datos se pudo determinar que el modelo entrenado es un buen modelo, con el que se puede trabajar.

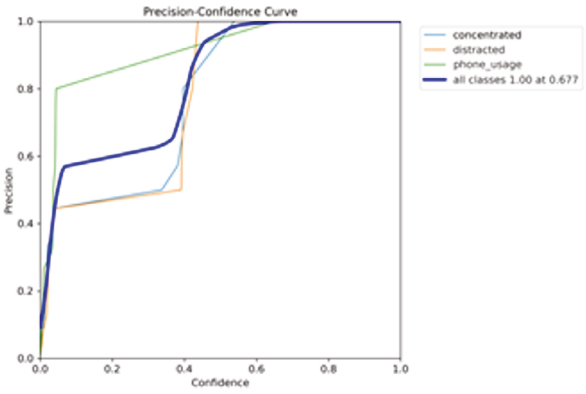

Figura 5: Gráfico de precisión del modelo

Además, la precisión promedio fue de 0,677, lo que nos indica que de cada diez imágenes, alrededor de siete se clasifican de manera correcta.

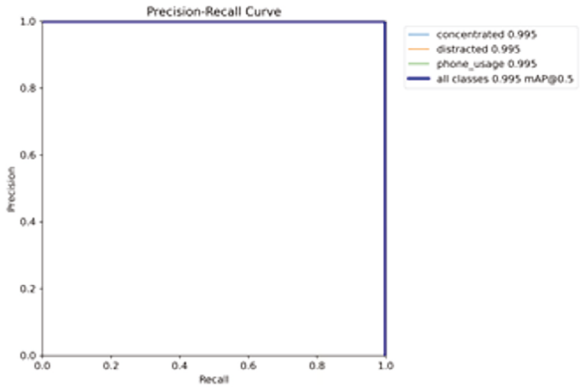

Figura 6: Gráfico de sensibilidad del modelo

El gráfico de sensibilidad indica que en promedio fue de 0,995 lo que es una muy buena medida, casi perfecta al considerar la cantidad de objetos por clase clasificados de manera correcta. Esto indicó que de cada diez imágenes, alrededor de diez son correctas y casi ninguna es un falso positivo.

En conclusión, la evaluación de este modelo fue positiva, por lo que se lo aprobó para implementarlo y ponerlo en marcha en el prototipo.

1.3. Implementación del modelo

Considerando la construcción de la interfaz gráfica, en donde se hizo uso del modelo, se utilizó la librería Flet de Python. La ventaja que se tiene al desarrollar sobre esta librería es la operatividad multiplataforma de la misma, es decir, la aplicación Desktop se puede ejecutar en sistemas operativos Windows, Mac, iOS y Linux.

Teniendo en cuenta los requerimientos mínimos de nuestra aplicación se construyeron los siguientes aspectos:

- cronómetro que marca los minutos de la sesión Pomodoro y toda la lógica que conlleva, por ejemplo, la función de comenzar la sesión, detenerla, cambiar de tipo de sesión entre actividad o descanso de forma automática;
- tablero que muestra la información estadística útil de la sesión actual por ejemplo, el tiempo de concentración, el tiempo de desconcentración, el máximo tiempo concentrado y un gráfico temporal;
- visor que permite visualizar lo que la cámara web observa y la región de interés en donde detecta un objeto y lo clasifica con ayuda del modelo entrenado.

Dentro de la aplicación, en la sección superior, el cronómetro marca los minutos transcurridos de la sesión Pomodoro junto a un botón para dar comienzo a la misma.

A continuación, se muestra el tablero con el porcentaje de concentración o desconcentración y del minuto a minuto de la sesión junto con unas tarjetas que marcan en números estos datos interesantes para el estudiante.

En la parte inferior de la aplicación se observa lo que captura la imagen de la cámara web y, por medio de la visión artificial, se dibuja un cuadro alrededor del objeto que detecta el modelo. Esto es una representación o muestra a fin de demostrar el estado de concentración del estudiante a la hora de tener activa la sesión.

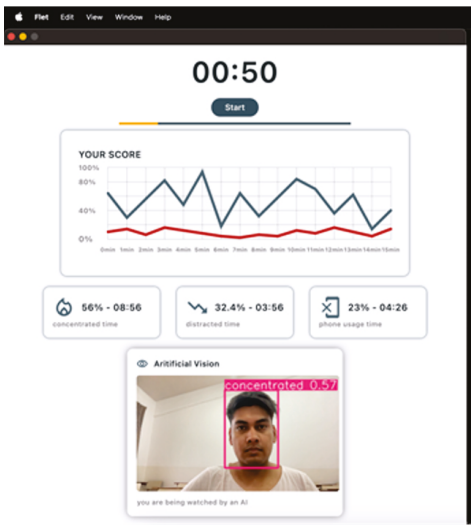

Figura 7: Interfaz de Asistente de concentración detectando el estado concentrado

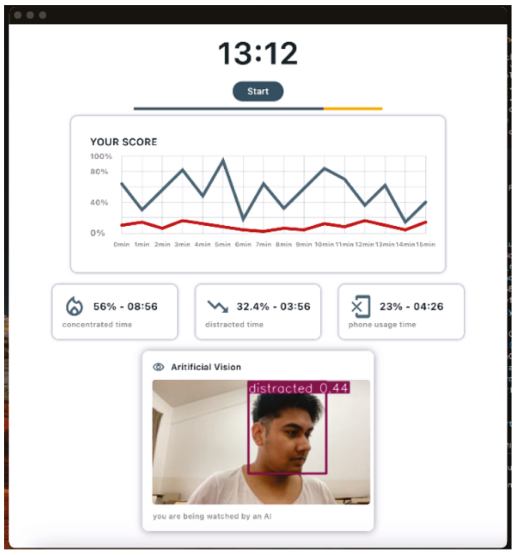

Figura 8: Interfaz de Asistente de concentración detectando el estado desconcentrado o distraído

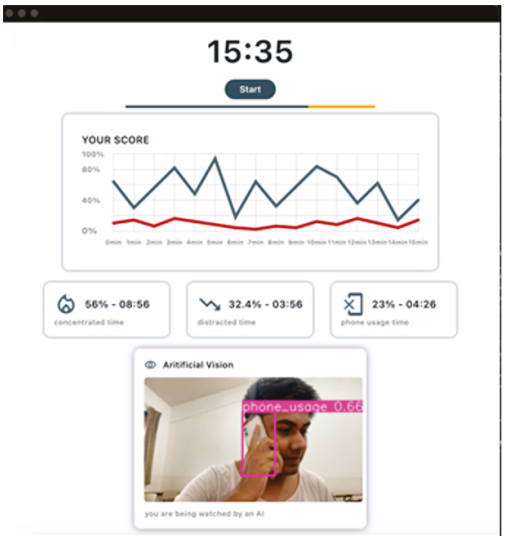

Figura 9: Interfaz de Asistente de concentración detectando el uso de teléfono móvil

Como se pudo observar en las diferentes capturas, se hizo uso del modelo sobre una interfaz gráfica, en otras palabras, una aplicación de escritorio.

2. Conclusión

En conclusión, el desarrollo de este prototipo dio respuesta a la problemática planteada acerca de la distracción de los estudiantes universitarios debido a las redes sociales y al uso de dispositivos móviles en el período de estudio. Se pudo validar que a través del entrenamiento de un modelo detector de objetos se logra alertar diferentes estados del estudiante, como así también el uso de teléfono móvil. Además, se desarrolló la herramienta como una aplicación de escritorio multiplataforma y se demostró que dentro de la misma se puede hacer uso del modelo como parte de la visión artificial de la herramienta. A futuro, este prototipo brindará retroalimentación para el estudiante y por medio de esta podrá corregir su concentración en sus sesiones de estudio y, en consecuencia, mejorar su productividad, mejorando el rendimiento académico.

Referencias

[1] L. E. Garate-Osuna, A Nava-Quevedo, C. Pacheco Millán. “Impacto de las redes sociales en el comportamiento de estudiantes de nivel superior. Revista de Investigación en Tecnologías de la Información: RITI”, 3(5), 19-23, 2015.

[2] I. Witten, E. Frank, M. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*, 4ta ed.

[3] IBM Knowledge Center. *Guía de CRISP-DM de IBM SPSS Modeler*

[4] S. J. Russell, J, Stuart, P. Norvig, J. M. Corchado Rodríguez, L. Joyanes Aguilar. *Inteligencia artificial: un enfoque moderno*. Pearson Prentice Hall, 2004.

[5] IBM. “¿Qué es *Deep Learning*?”. n.d. Disponible en https://www.ibm.com/es-es/topics/deep-learning

[6] IBM. “¿Qué son las redes neuronales convulcionales?”. n.d. Disponible en https://www.ibm.com/es-es/topics/convolutional-neural-networks

[7] A. Soto Serrano. “YOLO Object Detector for Onboard Driving Images. n.d.

[8] F. Cirillo. The Pomodoro Technique. n.d.