TecnoLógicas

Aplicación web para el análisis de emociones y atención de estudiantes

A Web Application to Analyze Students' Emotions and Attention

Alejandro Piedrahíta-Carvajal[1];
Paula Andrea Rodríguez-Marín[2];
Daniel F. Terraza-Arciniegas[3];
Mauricio Amaya-Gómez[4];
Leonardo Duque-Muñoz[5];
Juan David Martínez-Vargas[6]

[1] Instituto Tecnológico Metropolitano, Medellín-Colombia, alejandropiedrahita264000@correo.itm.edu.co
[2] Instituto Tecnológico Metropolitano, Medellín-Colombia, paularodriguez@itm.edu.co
[3] Instituto Tecnológico Metropolitano, Medellín-Colombia, danielterraza212285@correo.itm.edu.co
[4] Instituto Tecnológico Metropolitano, Medellín-Colombia, mauricioamaya189862@correo.itm.edu.co
[5] Instituto Tecnológico Metropolitano, Medellín-Colombia, leonardoduque@itm.edu.co
[6] Instituto Tecnológico Metropolitano, Medellín-Colombia, juanmartinez@itm.edu.co

ISSN-p: 0123-7799
ISSN-e: 2256-5337

Vol. 24, **nro.** 51, e1821, 2021

Recibido: 03 diciembre 2020
Aceptado: 19 marzo 2021
Disponible: 12 mayo 2021

Cómo citar / How to cite

A. Piedrahíta-Carvajal; P. A. Rodríguez-Marín; D. F. Terraza-Arciniegas; M. Amaya-Gómez; L. Duque-Muñoz; J. D. Martínez-Vargas, "Aplicación web para el análisis de emociones y atención de estudiantes", *TecnoLógicas*, vol. 24, nro. 51, e1821, 2021. https://doi.org/10.22430/22565337.1821

Resumen

El análisis de emociones y el monitoreo del nivel de atención de los estudiantes en entornos virtuales permite a los docentes tomar acciones para mejorar los procesos de enseñanza-aprendizaje. Por esta razón, este trabajo presenta la integración de dos modelos: uno para el reconocimiento de emociones y otro para el análisis de atención, ambos con el objetivo de hacer monitoreo durante la interacción de un estudiante en entornos virtuales. Dicha integración se realiza en una plataforma web desarrollada en el entorno flask, en la que se pueden ejecutar los modelos de inteligencia artificial utilizados para la interacción. Los resultados obtenidos muestran que la plataforma podría ser utilizada por docentes como mediadores del conocimiento, para entender el comportamiento de los estudiantes en entornos virtuales tanto síncronos como asíncronos, y para tomar acciones que mejoren la experiencia de aprendizaje. Como ventaja adicional, los resultados aquí mostrados resaltan las ventajas que trae utilizar el Modelo Vista Controlador (MVC) en aplicaciones web, empleando e integrando técnicas de inteligencia artificial a través del *framework* Flask.

Palabras clave

Aplicación web, monitoreo de atención, reconocimiento de emociones, reconocimiento de rostros.

Abstract

Analyzing and monitoring students' attention level in virtual environments allows teachers to take actions to improve teaching-learning processes. This study introduces the integration of two models, one for emotion recognition and one for attention analysis, both of them aimed at monitoring the interactions of students in virtual environments. Such integration was completed on a web platform employing the Flask framework, where the artificial intelligence models used to analyze the interaction can be executed. The results obtained show that teachers, as knowledge mediators, can use the platform to understand the behavior of the students in synchronous and asynchronous virtual environments and take actions to improve learning experiences. The results also highlight the advantages of employing the Model-View-Controller (MVC) pattern in web applications, using and integrating artificial intelligence techniques through the Flask framework.

Keywords

Web application, Attention monitoring, Emotion recognition, Facial recognition.

1. INTRODUCCIÓN

Los entornos educativos virtuales actuales, carecen de herramientas que permitan incluir la experiencia emocional de los aprendices con el fin de identificar aquellos casos en los cuales la emocionalidad y falta de atención afectan el proceso de aprendizaje. Algunos estudios [1]-[3] han constatado que la disposición emocional de los alumnos facilita el aprendizaje. En [4] se analiza la influencia de la atención en el desempeño escolar, encontrando que las dificultades en este componente generan deficiencias en el procesamiento de la información, lo cual influye negativamente en la adquisición de nuevos aprendizajes y, por lo tanto, en el rendimiento académico. Adicionalmente, en [5] se encontró que los estudiantes con deficiencia de atención presentan dificultad en la adquisición del código lector, así como en el componente de aprendizaje de lógica matemática [6].

El uso de aplicaciones con detección de rostros en tiempo real para el reconocimiento de emociones ha ganado un papel importante en estudios de visión e inteligencia artificial [7].

Cada día surgen nuevos usos para este tipo de técnicas, debido al acceso que tienen las personas a dispositivos que están conectados a internet y tienen cámara que permite la recolección e identificación de los rostros humanos. El desarrollo de métodos para la detección automática de emociones ha estado fuertemente ligado al conocimiento que se tiene sobre los cambios que estas generan en la fisiología de las personas [8], [9].

La unión de estas dos tendencias lleva a incorporar la detección de rostros en tiempo real en ambientes educativos mediados por las tecnologías de la información y la comunicación (TIC), analizando el nivel de compromiso de los estudiantes a través de sus expresiones faciales [10]-[13], donde se hace evidente la necesidad de los docentes de tener un encuentro cercano con sus estudiantes al poder detectar en ellos su atención o falta de atención [13], así como la emoción que el estudiante puede presentar en un momento determinado de la clase sincrónica, esto con el objetivo de obtener un aprendizaje significativo que lleve al estudiante a la adquisición y comprensión de nuevos conocimientos.

A pesar de que algunos trabajos en el estado del arte consideran la detección de emociones en ambientes educativos a través de modelos en los cuales se encasillan las emociones experimentadas por los aprendices [14], es importante considerar su medición teniendo en cuenta factores como la positividad o negatividad de estas como su nivel de atención, ya que esto determina el estímulo recibido por la herramienta de enseñanza para el aprendizaje [15], [16].

Para el desarrollo de una aplicación web que permita el monitoreo constante de los estudiantes, se seleccionó el lenguaje de programación de Python por ser un lenguaje de alto nivel, adecuado para código científico y de ingeniería, que es rápido y flexible [17].

Adicionalmente, Python tiene gran cantidad de librerías que hacen de este un lenguaje versátil para todo tipo de proyectos como la integración de modelos de inteligencia artificial con aprendizaje profundo. Cuenta con facilidades para la programación orientada a objetos, imperativa y funcional, por lo que se considera un lenguaje multiparadigmas [18].

En este artículo, se describe el desarrollo de una aplicación web, la cual permite el monitoreo de los estudiantes en un entorno virtual. Entre las librerías utilizadas se encuentra Pandas para análisis de datos; OpenCV, para procesamiento de imágenes; Torch, para generar algoritmos de aprendizaje de máquina e inteligencia artificial; Flask, para el desarrollo web [18], [19], entre otras. La aplicación propuesta se divide en dos módulos de software. El primero detecta los rostros de los sujetos y, a partir de sus microexpresiones, realiza el reconocimiento de tres emociones inducidas por la actividad que se está realizando en el ambiente virtual (felicidad, neutro y enojo). El segundo módulo detecta el estado de atención que presenta el sujeto (atención, no atención y ausencia de rostro). Este

reconocimiento se realiza en tiempo real. Para esto, se utilizó el paradigma de programación Modelo Vista Controlador (MVC), con el objetivo de separar el modelo de datos de la vista que presenta la información al usuario final y el controlador que gestiona las peticiones de la aplicación y se encarga de la comunicación.

Este documento está organizado de la siguiente manera: primero se presenta la metodología, que consta de la selección de algoritmos para la detección de emociones y atención, seguido del proceso de adquisición de la base de datos para el entrenamiento de los modelos y, finalmente, el desarrollo del aplicativo web. En la sección 3 se presentan los resultados y la discusión, mostrando cómo los algoritmos seleccionados para la detección de emociones y rostros es promisorio para el uso en ambientes virtuales de aprendizaje. Por último, se encuentran las conclusiones.

2. METODOLOGÍA

En este capítulo explicamos el desarrollo de los módulos que componen la plataforma web y se presentan los resultados obtenidos.

La aplicación web propuesta consta de dos módulos luego del reconocimiento del rostro en la cámara. El primero es el reconocimiento de emociones, el cual tiene tres clases: felicidad, neutral y enojo (modelo 1). El segundo permite identificar si una persona está prestando atención o no a la pantalla del computador, para esto se cuenta con dos clases: atención y no atención (modelo 2). Para el desarrollo del aplicativo se siguieron las fases de selección de la metodología, adquisición de las bases de datos con el entrenamiento y el desarrollo del aplicativo web que integra los modelos propuestos. A continuación, se detalla cada uno de estos procesos:

2.1 Adquisición de la base de datos

Se adquirieron dos bases de datos para realizar el entrenamiento de los modelos. Estos datos fueron obtenidos con la ayuda de los miembros del Semillero de Inteligencia Artificial del Instituto Tecnológico Metropolitano de Medellín, Colombia (ITM).

Base de datos 1: se registró en video el comportamiento de 9 estudiantes mientras observaban videos diseñados para generar tres emociones: felicidad, neutral o enojo. De los registros de video, tomado a 30 fotogramas por segundo (fps), se almacenaron cada uno de los *frames* en formato de imagen. Como resultado, se obtuvieron 9873 imágenes repartidas en las tres emociones, para la categoría Neutral 2425, para Felicidad 3907 y para Enojo 3539.

Base de datos 2: se registró en video el comportamiento de 15 estudiantes mientras atendían conferencias virtuales. Cada estudiante se instruyó para atender dos conferencias, una de su interés y otra con un tema de libre elección. Los participantes fueron instruidos para atender la primera videoconferencia y desatender la segunda, con el fin de tener datos suficientes de ambas clases. Como resultado se almacenaron secuencias de video de 1 segundo tomadas a 30 fps, con una distribución entre clase atención de 1228 muestras y clase no atención de 438 muestras.

2.2 Metodologías para el reconocimiento de rostros

Para la selección de las metodologías que permiten el reconocimiento de rostros, se estudiaron los métodos de visión artificial propuestas en [19]-[25] analizando sus pros y sus contras. Se concluyó que las metodologías presentadas en [23], [24] se adaptan mejor al

problema que se quiere resolver y presentan, además, un bajo costo computacional. Consecuentemente, se implementaron los siguientes modelos:

Modelo 1: este modelo se basa en el algoritmo Haar-cascade, propuesto por [25] y que fue diseñado para reconocer objetos en una imagen. En este trabajo se utilizó la implementación de Haar-cascade de la librería OpenCV para reconocer los rostros en una escena. Inicialmente, el algoritmo se alimenta de imágenes con rostros (muestras positivas) e imágenes sin rostros (muestras negativas) para entrenar el clasificador. El algoritmo convierte la imagen a escala de grises, luego obtiene características Haar (bordes, líneas y rectángulos). Estas características son utilizadas como *kernels* para realizar la convolución con las imágenes. Para cada característica encuentra el mejor umbral que clasifica las imágenes positivas de las negativas. Luego se seleccionan las características que presentan la menor tasa de error. Al principio de la clasificación, cada característica presenta el mismo peso; luego de cada clasificación, los pesos de las imágenes mal clasificadas son incrementados, se repite el proceso y se calculan las tasas de error y los nuevos pesos.

El proceso se continúa hasta que la precisión requerida o la tasa de error calculada sean adecuadas o el número de características requeridas sean encontradas. Una vez reconocido el rostro con Haar-cascade, se procede a hacer el reconocimiento de la emoción utilizando el algoritmo Histograma de Patrones Binarios Locales (LPBH, por sus siglas en inglés).

El algoritmo LPBH es un operador de texturas, el cual etiqueta los pixeles de una imagen umbralizando el vecindario de cada pixel y considerando el resultado como un número binario. Este proceso se explica en la Figura 1. La clasificación se realiza mediante el cálculo de la similitud entre los histogramas [26].

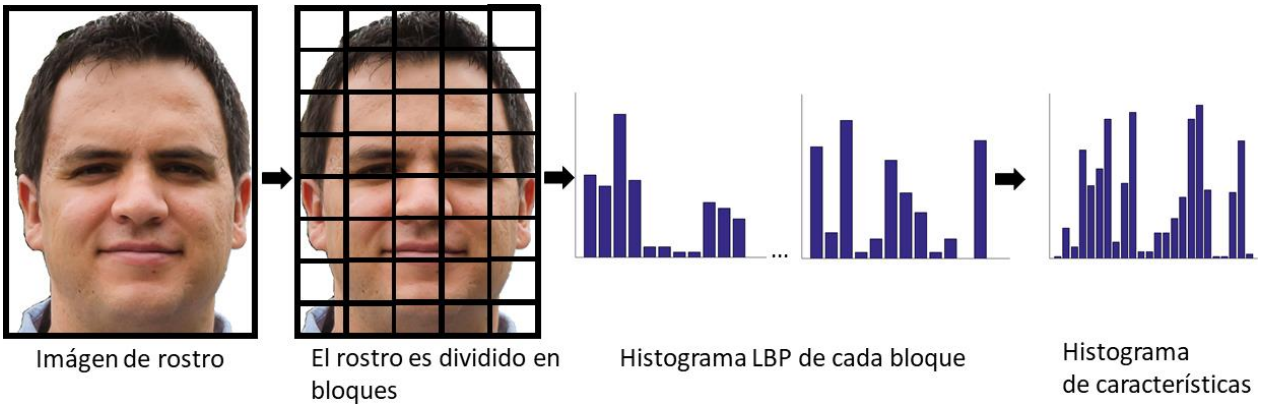


Figura 1. Cálculo de los histogramas mediante el algoritmo de LPBH. Fuente: elaboración propia.

Modelo 2: el segundo modelo toma como entradas secuencias de video de un segundo tomadas a 30 fps. Cada uno de los cuadros de esta secuencia se pasa por el modelo red neuronal convolucional multitarea (MTCNN, por sus siglas en inglés) [27], [28] del que se obtienen 5 puntos fiduciales del rostro de la persona: el centro de los ojos, la nariz y las comisuras de la boca. Adicionalmente, se obtienen los puntos superior izquierdo e inferior derecho del cuadro que enmarca el rostro (*bounding box*). Después, los puntos fiduciales se normalizan con respecto al *bounding box,* generando una secuencia para cada uno de los puntos fiduciales. Esta secuencia multivariada se utiliza para entrenar una red recurrente con arquitectura Long Short Term Memory (LSTM) con dimensión oculta h = 100, que clasifica entre las secuencias donde los sujetos prestan atención y las secuencias en las que no se presta atención. El procedimiento se puede ver en la Figura 2.

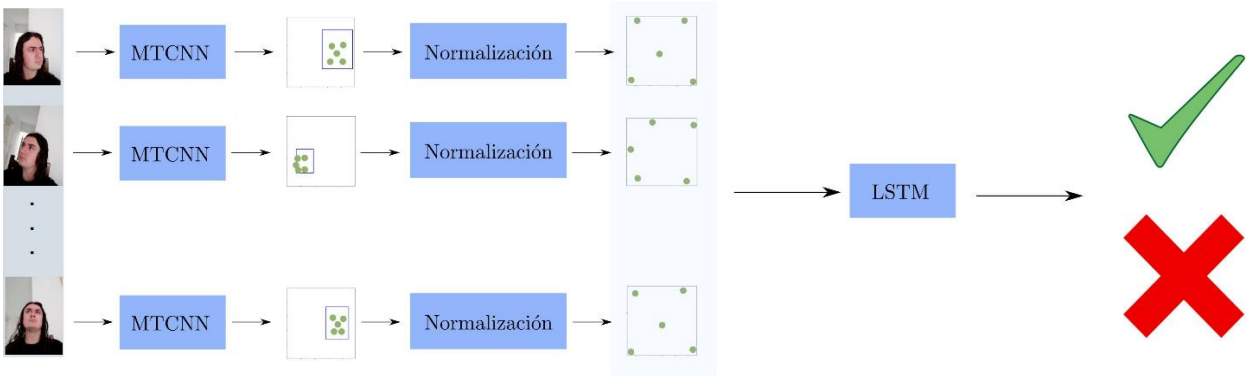


Figura 2. Esquema del modelo para detección de emociones. Fuente: elaboración propia.

2.3 Entrenamiento y prueba de los modelos

Los modelos se entrenaron dividiendo los datos en 80 % para entrenamiento y 20 % para prueba. De los datos de entrenamiento, se deja un 20 % adicional para validación con el fin de sintonizar los parámetros de los modelos. En el modelo 2, se utilizó Adam como algoritmo de optimización con una tasa de aprendizaje de 0.01. El tamaño del *batch* se ajustó en 16, se utilizó entropía cruzada como función de costo, y se entrenó durante 100 épocas. Los porcentajes de tasa de acierto de clasificación para cada uno de los modelos se muestran en la Tabla 1.

Tabla 1. Resultados de las tasas de acierto en la clasificación. Fuente: elaboración propia.

Modelo	Tasa de acierto
Modelo 1	95.0 %
Modelo 2	96.7 %

Finalmente, los modelos se probaron en tiempo real por parte de los miembros del semillero mientras realizaban las tareas que generaran emociones o requirieran atención. Si el modelo fallaba, se adquirían más imágenes de más participantes y se reentrenaba repitiendo nuevamente el proceso (ver Figura 3).

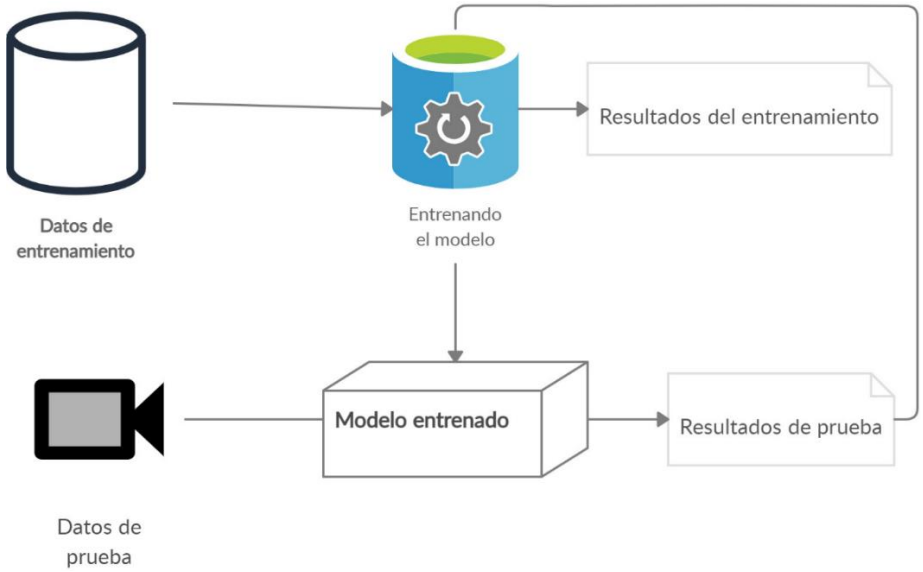


Figura 3. Representación del entrenamiento de los modelos. Fuente: elaboración propia.

2.4 Planificación y desarrollo del aplicativo web

Para el desarrollo y la implementación de la aplicación web, encargada de la integración de los dos modelos, se siguió con la arquitectura de programación MVC [29], que permite desarrollar aplicativos webs de forma más organizada y sencilla por capas. Además, se trabajó con Flask en un proyecto, utilizando dicha arquitectura y otro que no, encontrando que si se trabaja con base en el MVC se pueden obtener mayores velocidades en la carga de información sin afectar el tamaño o peso total de los archivos utilizados para la aplicación.

En el MVC, la capa vista almacena las diferentes vistas de la aplicación con las que interactúa el usuario, que son por lo general archivos HTML con estéticas y funcionalidades agregadas gracias a CSS y Javascript, respectivamente. La capa modelo contiene la información y códigos necesarios para el funcionamiento de los 2 modelos de visión artificial, y finalmente el controlador, aquel que regula la relación entre los modelos y las vistas será llevado por Flask (ver Figura 4), sirviendo como intermediario con rutas que adquieren la información mediante protocolos HTTP (Hypertext Transfer Protocol) y métodos GET.

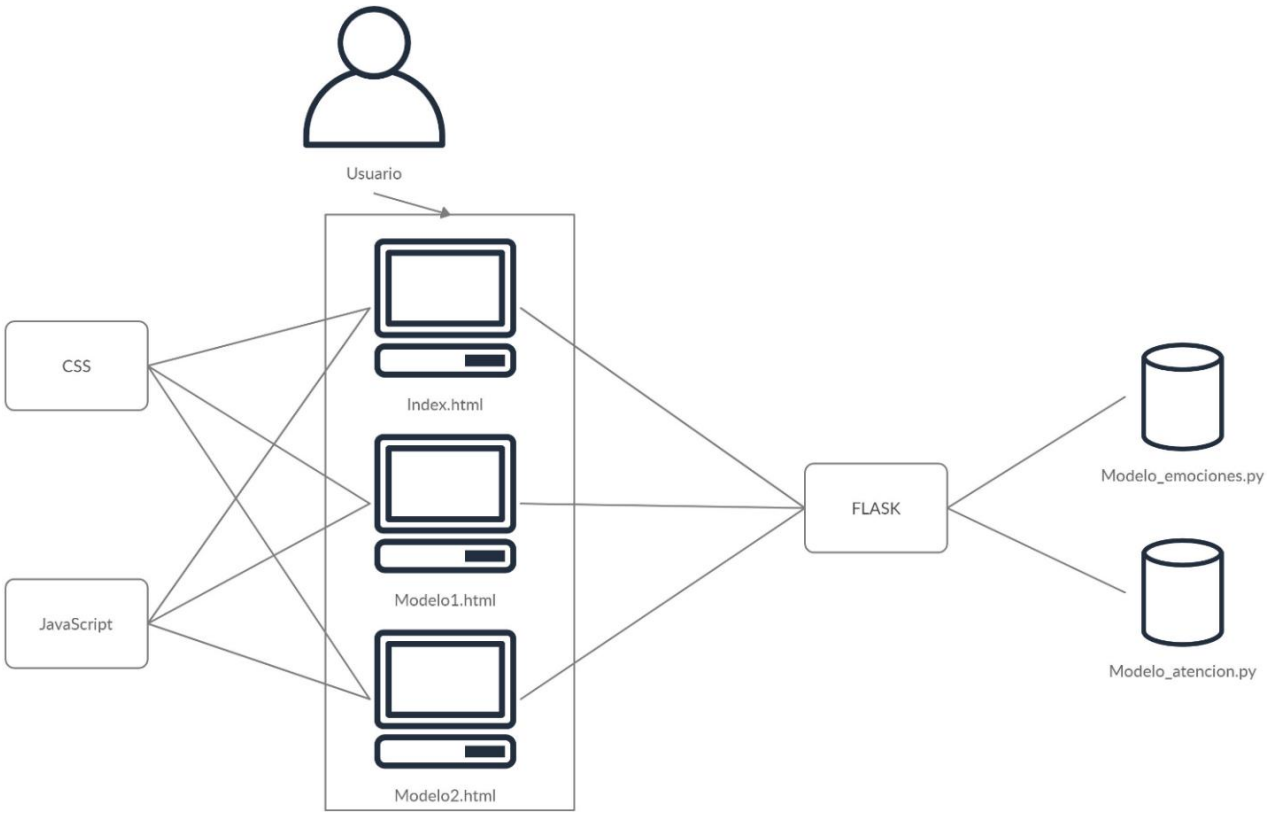


Figura 4. Representación del aplicativo web basado en el MVC. Fuente: elaboración propia.

Adicionalmente, se seleccionó Flask, por ser un *microframework* que cuenta con una estructura sencilla, su curva de aprendizaje es baja y no utiliza estructuras complejas para el manejo de los objetos. Flask tiene ventajas frente a otros *frameworks* como Django, debido a su optimización en las tareas sencillas, siendo una buena alternativa para este tipo de proyectos.

Para el proyecto se realizaron 3 vistas:

-Index.html, la cual muestra la página principal del proyecto con información acerca del mismo y las opciones para acceder a las otras 2 vistas.

-Modelo1.html, muestra información sobre el proyecto de detección de emociones y al final de esta se mostrará en tiempo real el *frame* capturado por la cámara y su respectiva clasificación de emoción.
-Modelo2.html, muestra la información sobre el proyecto de detección de atención y al final de esta se mostrará en tiempo real el *frame* capturado por la cámara y su respectiva clasificación de atención.

El entorno de desarrollo integrado (IDE, por sus siglas en inglés) utilizado para la programación fue Visual Studio Code debido a su poco peso y fácil instalación de *plug-in* o lenguajes de programación dentro del mismo, además, cuenta con un panel de control integrado que facilita el trabajo al momento de ejecutar la aplicación. Flask, internamente, utiliza el motor de plantillas Jinja y el kit de herramientas Werkzeug WSGI [30]. Es necesario almacenar los archivos HTML en una carpeta llamada *templates*, la cual posteriormente, sirve para que Flask pueda renderizar los archivos para su visualización dentro del buscador web al momento de ser llamados por alguna de las rutas.

Continuando con las vistas Flask, dentro de su estructura establece que archivos de CSS, JavaScript e imágenes que dan estilo a los archivos de HTML, se deben almacenar dentro de una carpeta llamada '*static'*.

Una vez realizadas las vistas, sus estilos y funcionalidades se prosiguió a hacer la capa de controlador y el enrutamiento del servidor a las diferentes instancias de la aplicación. Se creó la carpeta llamada '*Controlador'*, donde se almacena el archivo camera.py encargado de prender la cámara del computador del usuario y extraer información *frame* a *frame*. También están los archivos Load_data_Atencion.py y Load_data_emociones.py encargados de obtener los *frames* de camera.py, compararlos con el entrenamiento de los modelos y procesarlos para devolver una predicción pertinente al modelo que se esté utilizando.

Seguido de lo anterior, se creó una carpeta llamada *Modelo* que contenía en su interior 2 carpetas que almacenan el modelo de detección de emociones y el modelo de detección de atención, a su vez cada una contiene archivos necesarios para su funcionamiento como XML, imágenes de comparación, código de arquitectura de la red neuronal convolucional, entre otros.

Para el enrutamiento se creó el archivo llamado servidor.py, el cual debe ir afuera de las carpetas del MVC y será el encargado de obtener los requerimientos del usuario y enrutar ese requerimiento a través de las diferentes instancias de la aplicación para devolverle una vista con dicho requerimiento.

En las Figuras 5 y 6 se presentan los diagramas de secuencia diseñados para cada uno de los modelos, detección de emociones y detección de atención, presentando la interacción entre los *frames* desde que el usuario entra a la vista del modelo hasta mostrarle el resultado dentro de esa misma vista.

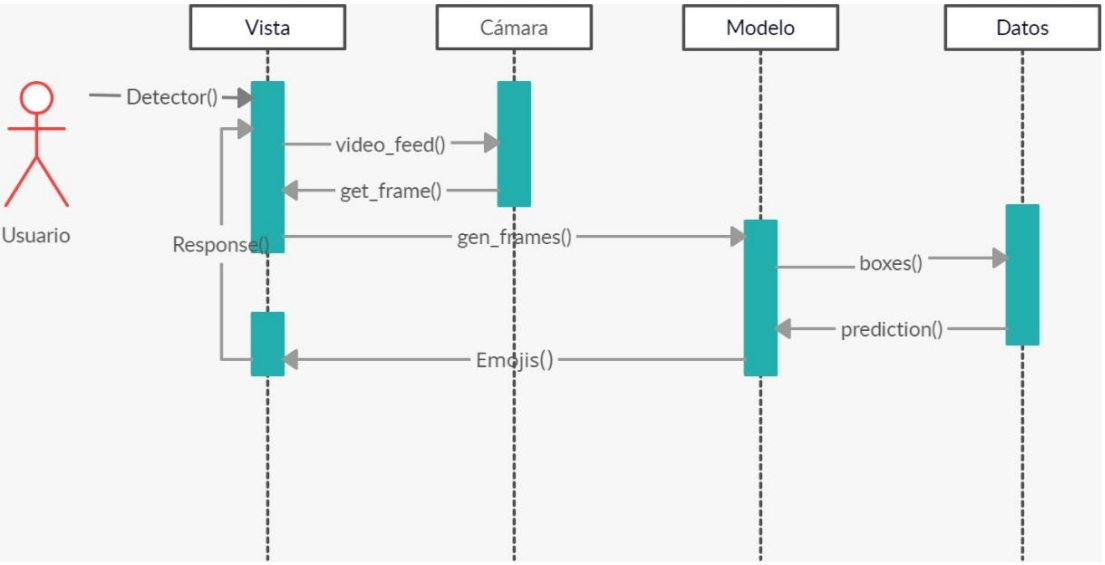


Figura 5. Diagrama de secuencia para el modelo de emociones. Fuente: elaboración propia.

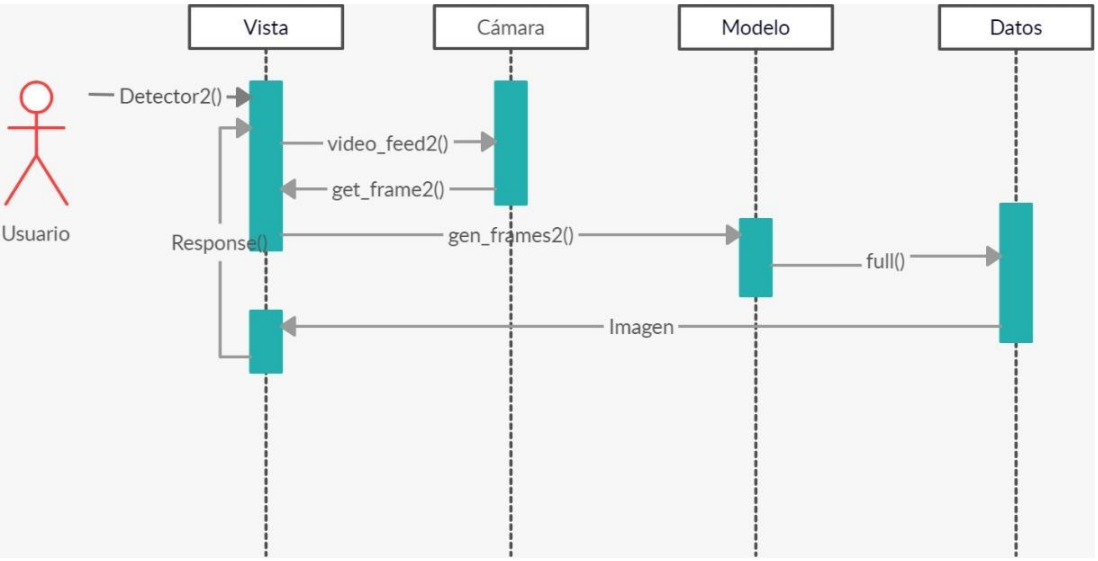


Figura 6. Diagrama de secuencia para el modelo de atención. Fuente: elaboración propia.

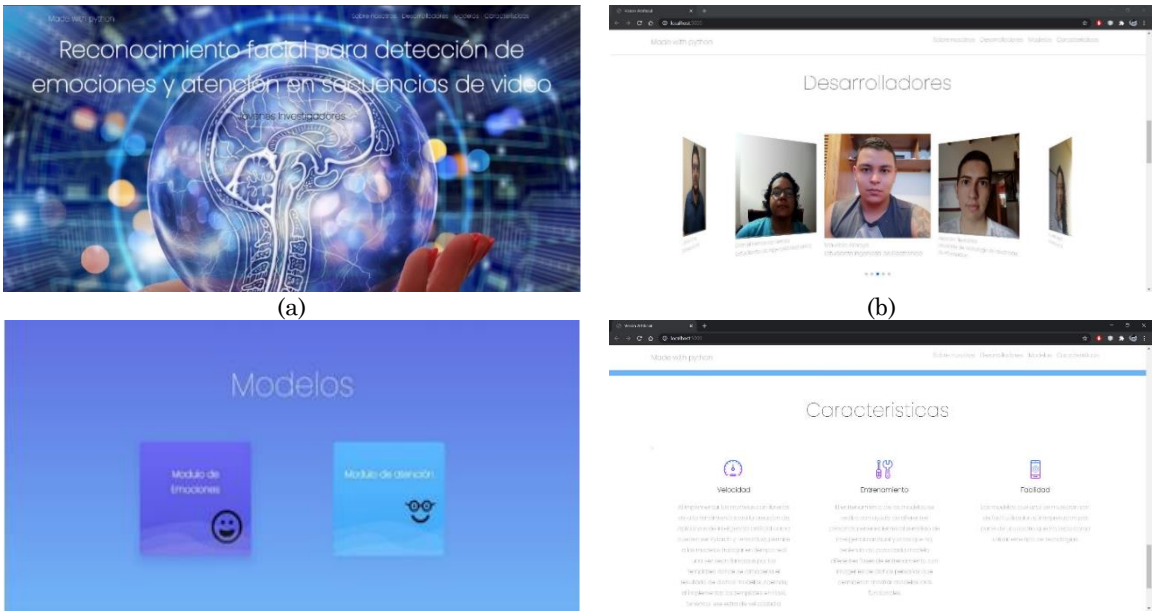
3. RESULTADOS Y DISCUSIÓN

En esta sección se presenta el funcionamiento de la aplicación web y la interacción con los dos modelos propuestos.

En la aplicación web se implementaron dos modelos de visión artificial, específicamente de reconocimiento de rostros en tipo real para el monitoreo de estudiantes. El primer modelo clasifica la emoción (neutro, enojo y felicidad) que tiene el estudiante, el segundo identifica si el estudiante está o no está prestando atención, asimismo si el estudiante abandonó o está presente.

La aplicación se desarrolló en Flask con la arquitectura MVC. Al ingresar a la aplicación el usuario entra a la vista index.html, donde se encuentra una pantalla de inicio (Figura 7a), un apartado mostrando los desarrolladores del aplicativo que se realizó en Javascript con la librería Swiper (Figura 7b), permitiendo mejorar el diseño y aplicar dinamismo a las

imágenes, un apartado con 2 botones (Figura 7c) para que el usuario vaya al modelo que desee y, finalmente, un apartado de características descritas con texto y algunos iconos (Figura 7d).



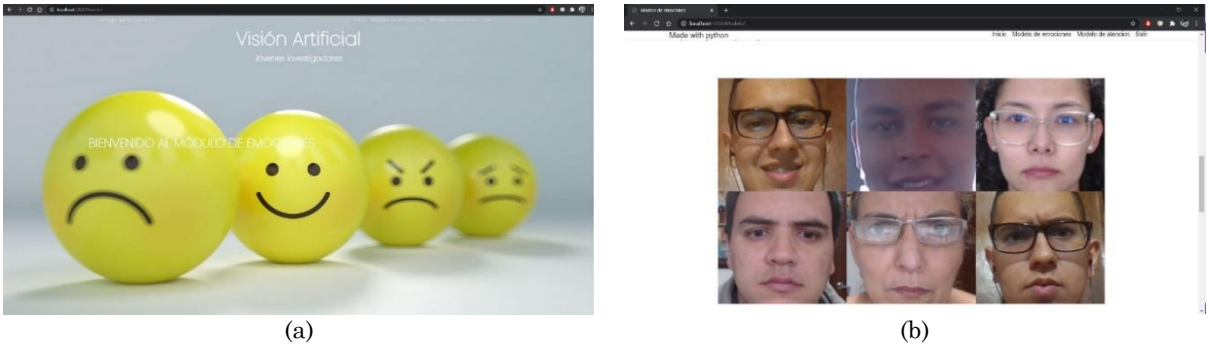
(a) (b) (c) (d)

Figura 7. Vistas de la aplicación web (a)Inicio, (b) Desarrolladores, (c) Modelos y (d) Características. Fuente: elaboración propia.

Al ejecutar la aplicación y abrirla en el navegador de Google Chrome no se observaron problemas de visualización, ni de velocidad de ejecución al navegar por los diferentes apartados de esta pestaña. Se realizaron 4 pruebas tomando el tiempo de carga para la visualización de la aplicación y se encontró un promedio de 9.47 segundos. Tampoco presentó problemas de enrutamiento pudiendo abrir alguna otra pestaña de los modelos como si fueran la pestaña principal.

3.1 Modelo 1: Detección de emociones

Para la pestaña del modelo de emociones, Modelo1.html, se observa un apartado de bienvenida (Figura 8a), un apartado de galería con algunas imágenes de ejemplo utilizadas para el entrenamiento del modelo (Figura 8b) y finalmente un apartado con un recuadro que muestra la captura que hace la cámara en tiempo real y la identificación de emociones.



(a) (b)

Figura 8. Vista del Modelo 1: detección de emociones de los estudiantes (a) Bienvenida, (b) Imágenes. Fuente: elaboración propia.

Al momento de acceder a esta pestaña desde la página principal anteriormente vista, se observó la rápida carga de la información visual (imágenes) y de texto. Sin embargo, en el apartado de la pestaña donde se muestra en tiempo real la captura de la cámara y su clasificación de emociones, se nota un pequeño retraso de 1 a 3 segundos para que se encienda la cámara y empiece a procesar los primeros *frames*. Se evidencia el buen funcionamiento del modelo, siendo este capaz de reconocer las expresiones faciales neutral, feliz y enojado para las que fue inicialmente programado; sin embargo, al ser un modelo con implementación simple, suele tener algunos problemas en entornos de iluminación deficiente. El enrutamiento desde la página principal hacia la página del modelo de emociones se realizó de manera satisfactoria sin ningún fallo o código de HTTP que pudiera indicar algún problema. En la Figura 9a se presenta la emoción neutral, en la Figura 9b la emoción de felicidad y finalmente en la Figura 9c la emoción de enojo.

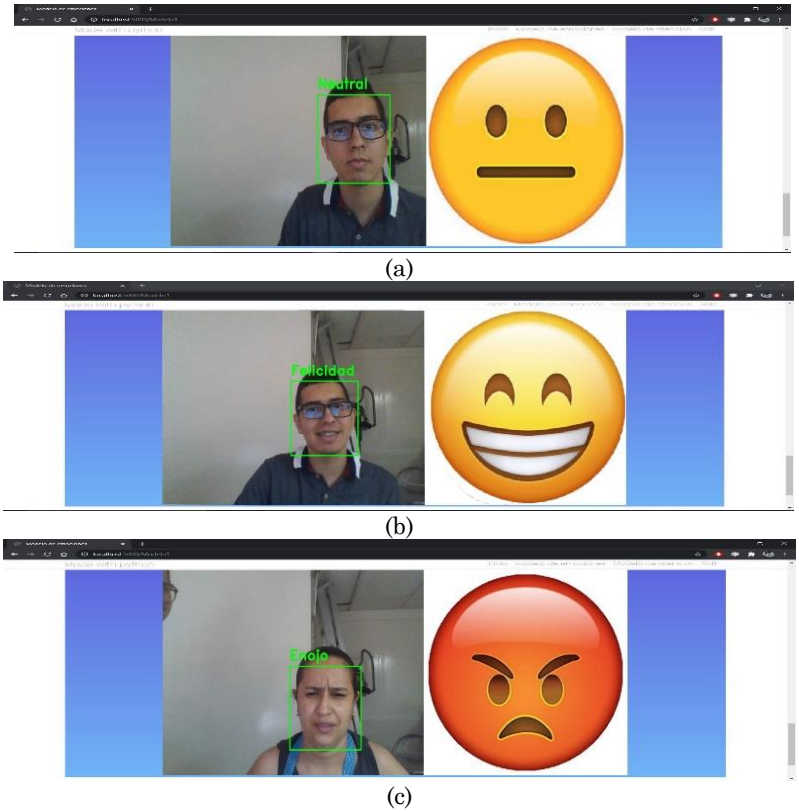


Figura 9. Funcionamiento del Modelo 1: detección de emociones de los estudiantes (a) Neutral, (b) Felicidad, (c) Enojo. Fuente: elaboración propia.

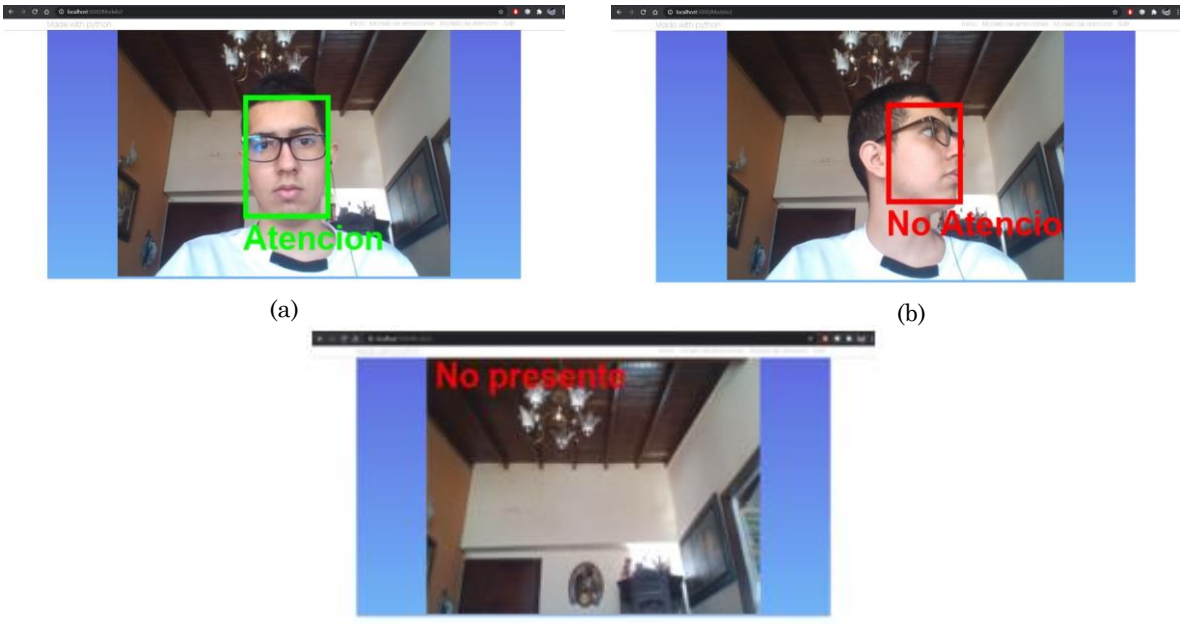
3.2 Modelo 2: Detección de atención

Igualmente, en la pestaña del modelo de atención, Modelo2.html, se observa la página de bienvenida, como se muestra en la Figura 10.

Figura 10. Vista del Modelo 1: detección de emociones de los estudiantes. Fuente: elaboración propia.

Asimismo, como el modelo 1, el funcionamiento del modelo 2 presenta buenos resultados de clasificación (como se presentó en la Tabla 1). En la Figura 11 se presenta el funcionamiento del modelo para cada una de las etiquetas utilizadas, en la Figura 11a, el estado de atención, en la Figura 11b el estado de no atención y finalmente en la Figura 11c el estado cuando No está presente.

Esta aplicación web, que integra los dos modelos, permite su utilización en ambientes virtuales para hacer seguimiento a los estudiantes en sus niveles de atención y la emoción que les causa la presentación de algunos materiales de aprendizaje.



(a) (b)

(c)

Figura 11. Funcionamiento del Modelo 2: detección de atención de los estudiantes (a) Atención, (b) No atención y (c) No presente.
Fuente: elaboración propia

4. CONCLUSIONES

En este trabajo se presenta el desarrollo de una aplicación web que integra dos modelos de reconocimiento de rostros; el primero de ellos para la detección de emociones expresadas ante la cámara; el otro, para la detección de atención a la pantalla del computador aplicables a entornos virtuales de aprendizaje. Para la composición de los modelos se utilizó el *framework* Flask y se puede concluir que su uso permite la integración de algoritmos de inteligencia artificial ofreciendo gran rendimiento. Además, Flask permite diseñar una interfaz agradable con características de usabilidad que integra los algoritmos y así el usuario final no tiene que estar en contacto ni tener conocimientos de Python, sino que consume la aplicación final. Por otro lado, se evidenciaron las ventajas de Flask al manejar el MVC, lo que permite un desarrollo más ágil y la reutilización de código. En este trabajo se separaron las capas así: en el Modelo se desarrollaron los algoritmos de detección de rostros, el controlador se encargó de la comunicación entre el modelo y la vista y en la capa de vista se desarrolló la interfaz o el *frontend* de la aplicación. Cuando se realizaban cambios en el modelo, por ejemplo, la actualización de los algoritmos de reconocimiento de rostros, la vista no se modificaba, lo que permitió que varios desarrolladores trabajaran en el mismo proyecto diferente capa, sin afectar el trabajo del otro.

Como trabajo futuro se plantea el uso de la aplicación web en un ambiente virtual de aprendizaje real, para ayudar al profesor a tomar acciones que mejoren la experiencia y el proceso de enseñanza–aprendizaje.

5. AGRADECIMIENTOS

Este trabajo se desarrolló bajo el proyecto de investigación “Implementación y aplicación de una metodología de detección de rostros y expresiones faciales en tiempo real, aplicando técnicas de inteligencia artificial, para el seguimiento de emociones en los estudiantes de la facultad de ingeniería”, con código P20227 financiado por la convocatoria para la formación de banco de elegibles de proyectos de Ciencia, Tecnología, Innovación y Creación para los grupos de Investigación del ITM - 2019. Además, apoyado por el semillero de inteligencia artificial del grupo de investigación en Máquinas Inteligentes y Reconocimiento de Patrones (MIRP), sus miembros y familiares quienes aportaron fotos para el entrenamiento y validación de los modelos.

CONFLICTOS DE INTERÉS DE LOS AUTORES

Los autores declaran no tener ningún conflicto de interés.

CONTRIBUCIÓN DE LOS AUTORES

Alejandro Piedrahíta-Carvajal: conceptualización de los modelos; diseño e implementación del aplicativo web; ejecución y validación de experimentos; preparación del documento borrador original; escritura del documento final.

Paula A. Rodríguez-Marín: tutora del joven investigador Alejandro Piedrahita, conceptualización de los modelos, escritura del documento final.

Daniel F. Terraza-Arciniegas: conceptualización de los modelos; validación de experimentos y código computacional del modelo de atención; revisión y edición del documento final.

Mauricio Amaya-Gómez: conceptualización de los modelos; validación de experimentos y código computacional del modelo de emociones; revisión y edición del documento final.

Leonardo Duque-Muñoz: tutor del joven investigador Mauricio Amaya, conceptualización de los modelos, escritura del documento final.

Juan D. Martínez-Vargas: tutor del joven investigador Daniel Terraza, conceptualización de los modelos; validación de experimentos y código computacional del modelo de atención; revisión, edición y aprobación del documento final.

6. REFERENCIAS

[1] A. Gegenfurtner; S. Narciss; L. K. Fryer; S. Järvelä; J. M. Harackiewicz, "Editorial: Affective Learning in Digital Education," *Front. Psychol.*, vol. 11, pp. 2020–2022, Jan. 2020. https://doi.org/10.3389/fpsyg.2020.630966

[2] A. Puente Ferreras, *Psicología contemporánea básica y aplicada*. Ed, Piramide. 2011. URL

[3] D. Hazarika; S. Poria; R. Zimmermann; R. Mihalcea, "Conversational transfer learning for emotion recognition," *Inf. Fusion*, vol. 65, pp. 1–12, Jan. 2021. https://doi.org/10.1016/j.inffus.2020.06.005

[4] N. Ibañez, "Las emociones en el aula," *Estud. Peagogicos*, vol. 1, no. 28, pp. 31–45, 2002. URL

[5] A. Fernández-Castillo; M. E. Gutiérrez Rojas, "Atención selectiva, ansiedad, sintomatología depresiva y rendimiento académico en adolescentes," *Electron. J. Res. Educ. Psychol.*, vol. 7, no. 1, pp. 49–76, Apr. 2009. URL

[6] G. Caicedo Delgado, "La enseñanza en ingeniería," *Tecnológicas*, no. 31, pp. 9–11, Nov. 2013. https://doi.org/10.22430/22565337.95

[7] V. Londoño-Osorio; J. Marín-Pineda; E. I. Arango-Zuluaga, "Introduction to Artificial Vision through Laboratory Guides Using Matlab," *TecnoLógicas*, pp. 591- 603, 2013. https://doi.org/10.22430/22565337.350

[8] M. M. Bundele; R. Banerjee, "Detection of fatigue of vehicular driver using skin conductance and oximetry pulse: a neural network approach," in *11th International Conference on Information Integration and web-based applications & services*, Lumpur 2009, pp. 739–744. https://doi.org/10.1145/1806338.1806478

[9] C. Li; C. Xu; Z. Feng, "Analysis of physiological for emotion recognition with the IRS model," *Neurocomputing*, vol. 178, pp. 103–111, Feb. 2016. https://doi.org/10.1016/j.neucom.2015.07.112

[10] S. K. D'Mello; S. D. Craig; A. C. Graesser, "Multimethod assessment of affective experience and expression during deep learning," *Int. J. Learn. Technol.*, vol. 4, no. 3/4, Oct. 2009, https://doi.org/10.1504/ijlt.2009.028805

[11] S. K. D'Mello; A. Graesser, "Multimodal semi-automated affect detection from conversational cues, gross body language, and facial features," *User Model. User-adapt. Interact.*, vol. 20, no. 2, pp. 147–187, May. 2010. https://doi.org/10.1007/s11257-010-9074-4

[12] A. Kapoor; R. W. Picard, "Multimodal affect recognition in learning environments," *Proceedings of the 13th ACM International Conference on Multimedia, MM 2005*. pp. 677–682, Nov. 2005. https://doi.org/10.1145/1101149.1101300

[13] B. Mcdaniel; S. D'Mello; B. King; P. Chipman; K. Tapp; A. Graesser, "Facial Features for Affective State Detection in Learning Environments," in *UC Merced Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 29, no. 29, pp. 467-472, 2007. URL

[14] S. Craig; A. Graesser; J. Sullins; B. Gholson, "Affect and learning: An exploratory look into the role of affect in learning with AutoTutor," *J. Educ. Media*, vol. 29, no. 3, pp. 241–250, Jul. 2010. https://doi.org/10.1080/1358165042000283101

[15] R. Pekrun; T. Goetz; A. C. Frenzel; P. Barchfeld; R. P. Perry, "Measuring emotions in students' learning and performance: The Achievement Emotions Questionnaire (AEQ)," *Contemp. Educ. Psychol.*, vol. 36, no. 1, pp. 36–48, Jan. 2011. https://doi.org/10.1016/j.cedpsych.2010.10.002

[16] C. Jonathan; J. P.-L. Tan; E. Koh; I. S. Caleon; S. H. Tay, "Engagement as flourishing: The contribution of positive emotions and coping to adolescents' engagement at school and with learning," *Psychology in the Schools*, vol. 45, no. 5, pp. 419–431, 2017. https://doi.org/10.1002/pits.20306

[17] T. E. Oliphant, "Python for scientific computing," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 10–20, Jun. 2007. https://doi.org/10.1109/MCSE.2007.58

[18] I. Challenger-Pérez; Y. Díaz-Ricardo; R. A. Becerra-García, "El lenguaje de programación Python," *Ciencias Holguín,* vol. 20, no. 2, pp. 1–13, Abr. 2014. https://www.redalyc.org/pdf/1815/181531232001.pdf

[19] M. Anggo; La Arapu, "Face Recognition Using Fisherface Method," en *2nd International Conference on Statistics, Mathematics, Teaching, and Research 2017,* Makassar, Indonesia, 2017, pp. 998–1001, 2018. https://doi.org/10.1088/1742-6596/1028/1/012119

[20] W. Shen; R. Khanna, "Prolog to Face Recognition: Eigenface, Elastic Matching, and Neural Nets," *Proc. IEEE*, vol. 85, no. 9, p. 1422, Sep. 1997. https://doi.org/10.1109/JPROC.1997.628711

[21] N. N. Mohammed; M. I. Khaleel; M. Latif; Z. Khalid, "Face Recognition Based on PCA with Weighted and Normalized Mahalanobis distance," en *International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, Bangkok, 2018, pp. 267–267, doi: https://doi.org/10.1109/iciibms.2018.8549971

[22] I. William; D. R. Ignatius Moses Setiadi; E. H. Rachmawanto; H. A. Santoso; C. A. Sari, "Face Recognition using FaceNet (Survey, Performance Test, and Comparison)," en *Proc. 2019 4th Int. Conf. Informatics Comput. ICIC*, Semarang, 2019. https://doi.org/10.1109/ICIC47613.2019.8985786

[23] E. Winarno; I. H. Al Amin; H. Februariyanti; P. W. Adi; W. Hadikurniawati; M. T. Anwar, "Attendance System Based on Face Recognition System Using CNN-PCA Method and Real-Time Camera," en *2019 2nd Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI*, pp. 301–304, Yogyakarta, 2019. https://doi.org/10.1109/ISRITI48646.2019.9034596

[24] C. Li; Z. Q; N. Jia; J. Wu, "Human face detection algorithm via Haar cascade classifier combined with three additional classifiers," en *ICEMI 2017 - Proc. IEEE 13th Int. Conf. Electron. Meas. Instruments*, pp. 483–487, Yangzhou, 2017. https://doi.org/10.1109/ICEMI.2017.8265863

[25] P. Viola; M. Jones, "Rapid object detection using a boosted cascade of simple features," en *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Kauai, 2001. https://doi.org/10.1109/cvpr.2001.990517

[26] L. Chen; Y. Hong Wang; Y. Ding Wang; D. Huang, "Face recognition with local binary patterns," en *Proc. 2009 Int. Conf. Mach. Learn. Cybern.*, Baoding, 2009, vol. 4, pp. 2433–2439. https://doi.org/10.1109/ICMLC.2009.5212189

[27] J. Li; T. Qiu; C. Wen; K. Xie; F. Q. Wen, "Robust face recognition using the deep C2D-CNN model based on decision-level fusion," *Sensors (Switzerland)*, vol. 18, no. 7, pp. 1–27, Jun. 2018. https://doi.org/10.3390/s18072080

[28] K. Zhang; Z. Zhang; Z. Li; Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, Aug. 2016. https://doi.org/10.1109/LSP.2016.2603342

[29] M. R. Mufid; A. Basofi; M. U. H. Al Rasyid; I. F. Rochimansyah; A. Rokhim, "Design an MVC Model using Python for Flask Framework Development," en *2019 International Electronics Symposium (IES).,* Surabaya, 2019, pp. 214–219. https://doi.org/10.1109/ELECSYM.2019.8901656

[30] F. A. Aslam; H. N. Mohammed; J. M. M. Munir; M. A. Gulamgaus, "Efficient Way Of Web Development Using Python And Flask," *Int. J. Adv. Res. Comput.,* vol. 6, no. 2, pp. 54–57, Mar. 2015. URL