Instituto Tecnológico y de Estudios Superiores de Monterrey

Escuela de Ingeniería y Ciencias

Ingeniería en Ciencia de Datos y Matemáticas

Momento de Retroalimentación: Reto Documentación

Inteligencia artificial avanzada para la ciencia de datos II

María Fernanda Torres Alcubilla A01285041

Morales Ramón Michelle Yareni A01552627

Paola Sofia Reyes Mancheno A00831314

Federico Medina García Corral A01721441

Supervisado por:
Iván Mauricio Amaya Contreras, Ph.D.
Edgar Covantes Osuna, Ph.D.
Hugo Terashima Marìn, Ph.D.

Monterrey, Nuevo León. Fecha, 1 de noviembre de 2023

1. Introducción

El presente escrito tiene como objetivo documentar nuestra propuesta de solución al proyecto "Medición de Asistencia y Participación en un Salón de Clases". Este proyecto tiene como finalidad desarrollar un sistema de reconocimiento facial que permita tomar asistencia de manera automatizada y medir la participación del alumnado en tiempo real dentro de un salón de clases, a través de modelos de redes neuronales. Esto conectado con una plataforma de acceso para docentes y estudiantes con funcionalidades específicas para cada una de estas credenciales, donde se muestre la información actualizada al final de las sesiones de clase en forma de dashboards. Con esto, tanto los docentes y estudiantes como directivos o empleados de la institución, puedan observar las estadísticas de las horas clase y tomar decisiones informadas.

En la documentación se presenta la metodología a seguir, donde se enumeran y describen los pasos. Se establecen las librerías utilizadas para la creación del modelo de reconocimiento facial y pose; se describe la interfaz, así como sus criterios de credenciales y conectividad con la nube; se establece Google Storage como herramienta para permitir la conectividad con el modelo y se documentan los distintos modelos generados, así como los siguientes pasos a seguir.

2. Metodología

A continuación se realizará la enumeración de los pasos a seguir para cumplir con el objetivo de este proyecto.

1. **Definición del escenario**: Para el escenario, se tomará en cuenta un espacio donde quepan por lo menos 6 estudiantes en donde se realizarán las pruebas del modelo. El espacio debe de estar bien iluminado, sin objetos que puedan intervenir entre la cámara y el estudiante al igual que debe de ser tomada con vista a las caras de los alumnos.

2. **Recopilación de información**: El video se hará como prueba, es decir, el equipo buscará simular un salón de clases en donde se pondrá a prueba el modelo.

3. **Programación de la detección de rostros**: El modelo debe de ser capaz de detectar los rostros de los alumnos al igual que sus poses, generando así una estadística sobre las asistencias

de las personas y la cantidad de veces que cada una de esta participó durante la clase (levantó la mano).

4. **Realización de Plataforma para consulta de asistencia**: Creación de una plataforma con StreamLit dirigida a estudiantes y profesores, donde según sus credenciales puedan recuperar información sobre asistencia y participación. Al crear la cuenta, sin importar el rol, se solicitará la carga de una foto para el entrenamiento del modelo.

3. Librerías

3.1. Face Recognition

La librería de *face_recognition* es una herramienta de Python que se utiliza para poder hacer reconocimiento facial tanto en imágenes como en videos. Dicha librería es una interfaz que se basa en la biblioteca de Aprendizaje Profundo o, popularmente conocido como Deep Learning, llamada dlib [Rosebrock, 2023]. Dicha librería tiene muchas características con las que se pueden elaborar distintos proyectos, pero las que se utilizaron para la elaboración de este son:

- **Detección de Rostros**: detecta los rostros presentes en una imagen.
- **Identificación de Rostros**: identifica cada uno de los rostros de manera individual.
- **Extracción de características**: extrae las principales características para poder obtener individualidades de cada rostro.
- **Correspondencia de rostros**: compara los rostros obtenidos en una imagen con los de un dataset previamente establecido.
- **Integración con cámaras y video**: permite utilizar el modelo en tiempo real para la detección e identificación de rostros.

3.2. OpenCV

Open Source Computer Vision Library (*OpenCV*) es una librería open source que permite utilizarse en el campo de visión por computadora al igual que el procesamiento de imágenes [Team,

2023]. Esta librería proporciona una gran cantidad de herramientas y algoritmos para trabajar con imágenes y videos. Algunas de estas herramientas que se utilizaron fueron:

- **Visión por computadora (Computer Vision)**: se utiliza para procesar y analizar imágenes y videos para después extraer información de estos.
- **Detección de Objetos**: se incluyen algoritmos para la detección de objetos y el seguimiento de estos dentro de un video.
- **Procesamiento de Video**: este permite que el usuario pueda capturar videos desde una cámara al igual que ayuda a trabajar con flujos de video en tiempo real.

3.3. Numpy

Numpy es una de las librerías clásicas que se utilizan frecuentemente en Python. Esta librería permite el procesamiento de datos numéricos y científicos [Developers, 1995]. Dentro de esta librería hay muchas funciones que se utilizan para una gran variedad de tareas, sin embargo, para el proyecto solamente se utilizó para la creación y manipulación de matrices que representan a las imágenes de los videos que se procesan dentro del modelo.

3.4. TensorFlow

TensorFlow es una librería utilizada para crear modelos de Deep Learning ya sea directamente o mediante bibliotecas contenedoras que simplifican el proceso creado sobre este [Brownlee, 2022]. Dentro de esta, se incluye el procesamiento de imágenes y la visión por computadora. Este módulo, llamado *tf-pose-estimation*, se enfoca en la estimación de poses humanas a partir de imágenes. Esto se le conoce como "multipose", el cual es lo que se utiliza para el proyecto. Este es un modelo de Red Neuronal Profunda que ya previamente fue entrenado utilizando un gran número de datos de prueba y entrenamiento por parte de los creadores de este.

4. Interfaz

La primera parte de la interfaz es el ingreso de credenciales, para esto se utiliza la librería de *Streamlit*, junto con las herramientas de Google Storage y Firestore. Para esta parte se permite

realizar LogIn o SignIn, en el caso de tener una cuenta, se ingresan las credenciales correspondientes y te da la opción de dirigirte a la plataforma respectiva a tu rol dentro de la institución (estudiante o docente). Tanto el correo como la contraseña se verifican con las cuentas almacenadas de forma segura en Firestore, en el caso de no encontrar coincidencias, se muestra un error de Credenciales Inválidas.

Por otra parte, si necesitas crear una cuenta se te solicita tu correo institucional, contraseña y la carga de un archivo .jpeg de tu rostro para el entrenamiento del modelo, para poder crear la cuenta, se toman en cuenta los siguientes criterios:

- Cada correo solo puede tener una sola cuenta, por lo que si este ya se encuentra registrado se despliega el error correspondiente.
- No se permiten correos que no sean institucionales (@tec.mx).
- La contraseña debe ser mayor a 6 carácteres.
- Como el modelo Haar Cascade es bueno para la detección de rostros en general, se hizo uso de este para verificar si en la foto se encontraba un rostro. En el caso que no se detecte se solicita subir otro archivo, esto con el fin de tener confiabilidad en los datos de entrenamiento de nuestro modelo.

Al crear la cuenta, automáticamente se guarda la foto en el bucket de Cloud Storage nombrada como el localId de la cuenta, este se genera automáticamente por Firestore y nos brinda una mayor seguridad, ya que no se hace uso de información sensible. Por otra parte, en Firestore se guarda la cuenta (correo, contraseña y rol), donde el rol se obtiene con el prefijo del correo electrónico, si se tiene “a0” se identifica como estudiante y en lo contrario, como docente.

Cuando se ingresan credenciales correctas, se dirige a la plataforma de estudiantes o docentes. En la primera se tienen dos pestañas, Home y Mis Cursos y en la plataforma de docentes se agrega la pestaña de Estadísticas. La implementación del contenido de cada pestaña se encuentra en proceso de elaboración, sin embargo, se tiene como objetivo el despliegue de información general del perfil, de cada curso inscrito, dentro de estos las estadísticas generales o individuales de la cuenta.

5. Cloud Storage

La plataforma de servicios en la nube que se utilizará para montar la arquitectura de datos para este proyecto, será Google Cloud. Esto debido a que se maneja información a tiempo real, el porcentaje del tiempo activo de los servicios es del 99.99 %, con lo que dicha plataforma permitiría un funcionamiento correcto del proyecto. De igual manera, un punto muy importante para la escalabilidad, es la posibilidad de optimizar los servicios ya sea para un mejor performance o para reducir costos; lo que beneficia al proyecto tanto en un principio, cuando la información que se maneja es baja, como cuando esta se decida escalar a nivel institucional. Además, ofrece una alta disponibilidad y escalabilidad, así como medidas de seguridad avanzadas y flexibilidad en los costos.

Los servicios a utilizar, son los siguientes:

- **SQL:** Cloud SQL opera mediante el uso de instancias, donde cada instancia se ejecuta en una máquina virtual (VM) alojada en un servidor de Google Cloud. Cada VM está configurada para ejecutar el programa de base de datos junto con agentes de servicio que brindan funciones de apoyo, como registro y supervisión. En este servicio, se alojará la base de datos con la asistencia de lxs estudiantes, así como su participación.

- **Dataflow:** Dataflow es un servicio administrado que ejecuta una amplia variedad de patrones de procesamiento de datos. Permite desarrollar canalizaciones de transmisión de datos de forma simplificada y rápida con una latencia de datos más baja.

- **Cloud Functions:** Este servicio permite desarrollar aplicaciones dentro de la infraestructura de Google sin necesidad de servidores físicos. Con Cloud Functions nos aseguraremos de que las subidas de contenido a Cloud Storage, cambios en los registros de bases de datos o solicitudes a puntos finales HTTP se ejecuten y gestionen automáticamente.

- **Cloud Storage:** Este servicio de Google Cloud, funciona como almacenamiento para datos No SQL. Está creada para utilizarla para data lakes y Big Data, pues tiene gran capacidad de almacenamiento, así como un manejo de transfrerencia de información rápido y confiable.

Estos servicios se utilizarán en la arquitectura que se encuentra en la Gráfica 1, la cual funciona de la siguiente manera. Primero, se registrará a lxs alumnxs con sus localID obtenido de Firestore

y una fotografía a través de Streamlit, que se estará ejecutando con Cloud Functions. Al utilizar la librería de Streamlit, el archivo que se utiliza funciona como Back end y Front end al mismo tiempo; es decir, no existe esta división como tal. La información tomada de la plataforma, se guardará en Cloud Storage, en carpetas nombradas con los ID correspondientes de lxs alumnxs para posteriormente entrenar el modelo de Machine Learning dentro de Cloud Functions.

El video en tiempo real será procesado por Google Dataflow, el cual pasará nuevamente por Cloud Functions para tomar la asistencia y participación. Esto será hasta que se termine la clase o se interrumpa la grabación. Luego, esta nueva información se almacenará en Cloud SQL el cual tendrá una conexión a Streamlit para que cada vez que haya una actualización en la base de datos se actualicen también las gráficas de las estadísticas de la clase. Finalmente se mostrarán los resultados al ususario en donde podrá acceder a las asistencias y participaciones de las personas. [1]

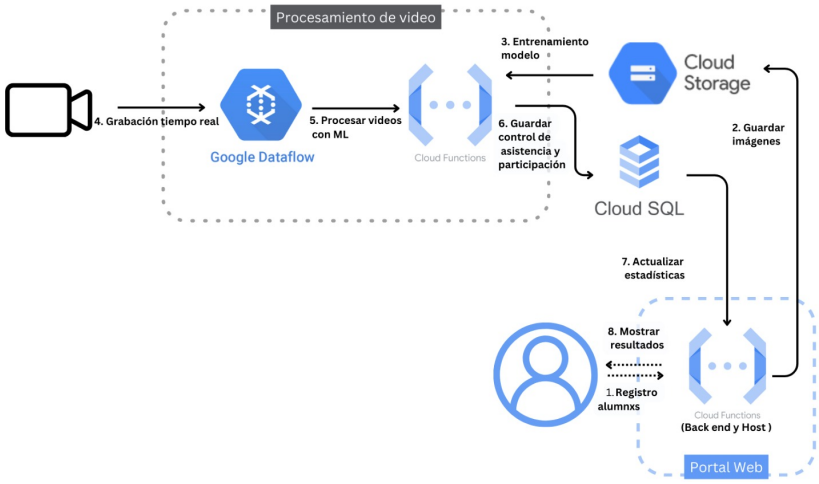

Figura 1: Esquema del procesamiento de datos e implementación del modelo.

6. Modelos Generados

6.1. Modelos Anteriores

Anteriormente, para la realización del proyecto, se utilizó un modelo de detección y reconocimiento de rostros con Haar Cascade. Este modelo al ser entrenado funcionaba muy bien, de hecho cometía pocos errores a la hora de detectar caras e identificarlas. Sin embargo, se optó por dejar

atrás dicho modelo gracias a que el modelo funcionaba a través de las características Haar al igual que un clasificador en cascada, la cual utiliza un enfoque en patrones rectangulares de intensidad de pixeles, lo cual no reconocía rostros utilizando diferentes características faciales como se quería. Esto implica que Haar Cascade es un muy buen modelo para la detección de rostros, pero no para la identificación de características faciales más específicas, lo cual limita mucho la flexibilidad del modelo. Igualmente, el modelo con Haar Cascade no utliliza una Red Neuronal Profunda, lo cual fue una de las razones por la que se optó por cambiar de modelo que pudiera utilizar ambas características para el reconocimiento de rostros, lo cual es mejor para el proyecto en elaboración.

Por otro lado, para la detección de pose se utilizó MediaPipe, la cual es una librería desarrollada por Google para el reconocimiento de gestos. Dicha librería nos fue muy útil para poder desarrollar el modelo de detección de pose ya que funcionaba muy bien, hasta que se comenzaron a realizar pruebas en donde se requería que el modelo reconozca gestos de múltiples personas. En este momento, nos percatamos de que el modelo no funcionaba adecuadamente cuando se incluía a más de una persona en la imagen, por lo que se optó por cambiar de modelo.

6.2. Modelos Actuales

Para el modelo actual, se está utilizando face_recognition y tensorflow. Como ya se explicó en la Sección de Liberías, ambas librerías nos sirven para la detección de objetos, reconocimiento facial y detección de gestos. Con este, hemos logrado crear un modelo que detecte con gran precisión a múltiples personas dentro de una imágen al igual que los gestos que hacen en un momento dado. Para esto, se tiene una base de datos con una imagen para cada una de las personas que estarán involucradas en las pruebas en donde se registraron sus identificadores. Luego, el modelo ingresa a la base de datos y detecta a las personas que están en la imagen para después seleccionarlas y seguirlas durante la clase, para así detectar la asistencia y las participaciones de estas.

6.3. Pendientes

Hasta el momento, no se ha logrado obtener con claridad el conteo de asistencia dentro del modelo ya que hay instancias donde la imágen no detecta a los participantes y los evalúa como si no estuvieron presentes en el salón. Para esto, se buscará hacer que el modelo cuente como asistencia a un estudiante cuando por lo menos aparece una vez en la imagen del salón de clases. Por otro lado,

el modelo para la detección de pose debe de ser refinado para poder detectar en el momento en que algún estudiante participe utilizando diferentes poses o establecer una pose específica para contar la participación. Finalmente, es necesario unir el modelo con la interfaz ya que hay algunos detalles que no nos han permitido avanzar mucho con la interfaz pero creemos que pueden ser arreglados en poco tiempo para ya enfocarnos directamente en el modelo y en el perfeccionamiento de este.

Referencias

Brownlee, J. (2022, julio). Introduction to the Python Deep Learning library TensorFlow. https://machinelearningmastery.com/introduction-python-deep-learning-library-tensorflow/

Developers, N. (1995). NumPy: The absolute basics for beginners. https://numpy.org/doc/stable/user/absolute_beginners.html#

Rosebrock, A. (2023, mayo). Face detection with dlib (Hog and CNN). https://pyimagesearch.com/2021/04/19/face-detection-with-dlib-hog-and-cnn/

Team, G. L. (2023, agosto). OpenCV tutorial: A guide to learn opencv in python. https://www.mygreatlearning.com/blog/opencv-tutorial-in-python/#:~:text=OpenCV%20is%20a%20Python%20library,learning%20more%20about%20the%20library.