Universidad Autónoma de Baja California
Facultad de Contaduría y Administración

Inteligencia de Negocios

Materia: Big Data

“Proyecto Final - Teachable Machine”

-Alumno: Ramírez Cardenas Luis Armando

-Matricula: 2200607

-Grupo: 961

-Fecha: 27/11/2025

-Profesor: Adrián Rodríguez Aguiñaga

Reporte de Proyecto Final

1. Introducción

En el contexto de la computación moderna, el procesamiento de grandes volúmenes de datos y la inteligencia artificial constituyen herramientas fundamentales para la automatización de procesos complejos. Particularmente, la visión por computadora ha evolucionado significativamente, permitiendo la interpretación automatizada de imágenes digitales para facilitar interfaces hombre-máquina más naturales y eficientes.

El presente informe técnico documenta el ciclo de desarrollo de una aplicación web que integra modelos de aprendizaje automático (*Machine Learning*) para la clasificación de imágenes en tiempo real. Se ha seleccionado la dinámica del juego "Piedra, Papel o Tijera" como caso de estudio para validar la capacidad de una Red Neuronal Convolucional (CNN) para identificar patrones visuales específicos y ejecutar lógica condicional basada en dicha identificación.

La implementación se sustenta en la utilización de bibliotecas de JavaScript de alto rendimiento, específicamente TensorFlow.js, lo cual permite la ejecución de la inferencia del modelo directamente en el navegador del cliente. Este enfoque descentralizado optimiza el uso de recursos y reduce la latencia de red, prescindiendo de servidores externos para el procesamiento de imágenes.

2. Planteamiento del Problema

El desarrollo de interfaces web que interactúen mediante estímulos visuales en tiempo real presenta desafíos técnicos, principalmente asociados a la latencia de respuesta y al consumo de recursos computacionales. Los métodos tradicionales de entrada (teclado y ratón) limitan la naturalidad de la interacción en aplicaciones modernas.

La problemática abordada en este proyecto consiste en la integración efectiva de un modelo de visión computacional pre-entrenado en una arquitectura web estándar. El sistema debe ser capaz de distinguir con precisión entre tres configuraciones manuales morfológicamente distintas bajo condiciones de iluminación variables, asegurando una respuesta inmediata para la experiencia del usuario sin depender de procesamiento en la nube (*backend*).

3. Justificación

La pertinencia de este proyecto reside en la creciente demanda de interfaces hombre-máquina más naturales e intuitivas (NUI, por sus siglas en inglés). En un contexto donde la computación ubicua y el "Internet de las Cosas" (IoT) cobran mayor relevancia, la capacidad de controlar sistemas mediante gestos, sin contacto físico, representa un avance significativo en términos de accesibilidad e higiene digital.

Desde una perspectiva académica, este desarrollo justifica su implementación al servir como una demostración práctica de cómo los paradigmas de **Big Data** no solo se limitan al análisis estadístico de bases de datos, sino que también abarcan el procesamiento de flujos de datos no estructurados, como el video en tiempo real. La utilización de redes neuronales en el navegador democratiza el acceso a la inteligencia artificial, permitiendo que desarrollos complejos sean accesibles a cualquier usuario con conexión a internet, reduciendo la brecha tecnológica.

4. Objetivos

4.1 Objetivo General

Diseñar, entrenar e implementar un sistema de software basado en tecnologías web que utilice redes neuronales para la clasificación de gestos manuales en tiempo real, permitiendo la interacción lógica con un algoritmo automatizado.

4.2 Objetivos Específicos

- **Estructuración del Dataset:** Generar un conjunto de datos visuales representativo y balanceado para las clases *Piedra*, *Papel*, *Tijera* y una clase de control (*Ruido/Fondo*).
- **Entrenamiento del Modelo:** Configurar una Red Neuronal Convolucional utilizando la plataforma Teachable Machine, optimizando los hiperparámetros para maximizar la precisión de la inferencia.
- **Implementación Frontend:** Desarrollar la interfaz de usuario utilizando HTML5 y CSS3, integrando la biblioteca TensorFlow.js para la gestión de la cámara web y la carga del modelo.
- **Desarrollo de Lógica:** Codificar el algoritmo de decisión en JavaScript para la gestión del flujo de estados, incluyendo temporizadores, comparación de vectores de probabilidad y actualización de métricas.

5. Alcances del Proyecto

El sistema desarrollado abarca la implementación completa del ciclo de vida de un producto de software basado en IA, desde la recolección de datos hasta su despliegue. Los alcances definidos son:

- **Compatibilidad Web:** El software es capaz de operar en la mayoría de los navegadores modernos (Chrome, Firefox, Edge) sin necesidad de instalación de plugins adicionales.
- **Tiempo Real:** El reconocimiento se realiza con una latencia mínima, permitiendo una actualización de estado constante (60 cuadros por segundo).
- **Clasificación Múltiple:** El modelo distingue y clasifica correctamente cuatro estados específicos: Piedra, Papel, Tijera y Estado de Reposo (Nada/Fondo).

6. Marco Teórico

6.1 Aprendizaje Supervisado

Este proyecto se fundamenta en el paradigma del aprendizaje supervisado, un enfoque donde el algoritmo es entrenado utilizando un conjunto de datos que incluye tanto las entradas (imágenes) como las salidas deseadas (etiquetas correctas). El modelo ajusta sus pesos internos para minimizar el error entre su predicción y la etiqueta real.

6.2 Redes Neuronales Convolucionales (CNN)

Las CNN son un tipo especializado de red neuronal artificial diseñada para procesar datos con topología matricial, como las imágenes. A diferencia de las redes densas tradicionales, las CNN utilizan operaciones de convolución para extraer características jerárquicas (bordes, texturas, formas complejas) de la imagen de entrada, haciéndolas el estándar industrial para tareas de clasificación visual.

6.3 Computación en el Cliente (Edge AI) con TensorFlow.js

TensorFlow.js es una biblioteca de código abierto para computación numérica y aprendizaje automático. Su implementación permite que el proceso de inferencia ocurra localmente utilizando la aceleración gráfica (WebGL)

del dispositivo del usuario. Esto garantiza la privacidad de los datos, ya que las imágenes capturadas por la cámara nunca abandonan el dispositivo local.

7. Explicación del Modelo de Teachable Machine

Para el núcleo de inteligencia artificial del proyecto, se utilizó la plataforma **Google Teachable Machine**. Esta herramienta facilitó el entrenamiento de una Red Neuronal Convolucional optimizada para la web.

Estructura del Entrenamiento: El modelo fue entrenado bajo la modalidad de clasificación de imágenes, definiendo cuatro clases distintas para asegurar la precisión de la inferencia y evitar ambigüedades:

1. **Clase "Piedra":** Muestras visuales de la mano con el puño cerrado.
2. **Clase "Papel":** Muestras de la mano con la palma totalmente extendida.
3. **Clase "Tijera":** Muestras con los dedos índice y medio extendidos.
4. **Clase "Fondo/Nada" (Control):** Esta clase es crítica para la estabilidad del sistema. Se capturaron imágenes del escenario vacío y movimientos aleatorios para enseñar al modelo a "no hacer nada" cuando no se detecta una intención de juego clara, reduciendo drásticamente los falsos positivos.

Una vez validada la precisión del modelo, este fue exportado a formato **TensorFlow.js**, generando los archivos de arquitectura (`model.json`) y los pesos sinápticos binarios (`weights.bin`), los cuales contienen la lógica matemática aprendida por la red.

8. Proceso de Desarrollo y Decisiones Técnicas

El ciclo de desarrollo del software se centró en la integración del modelo exportado con una aplicación web responsiva.

A. Arquitectura Técnica

Se optó por ejecutar el modelo directamente en el navegador (*Client-side*). Esta decisión técnica ofrece dos ventajas fundamentales:

- **Latencia Cero:** Al no enviar las imágenes a un servidor externo para su procesamiento, la detección es inmediata, lo cual es crucial para la fluidez del juego.
- **Privacidad:** Las imágenes capturadas por la cámara web nunca abandonan el dispositivo del usuario.

B. Desarrollo del Frontend (Interfaz de Usuario)

Se diseñó una interfaz intuitiva utilizando HTML5 y CSS3. Se implementó un estilo visual moderno (*Glassmorphism*) y un fondo animado mediante CSS puro para mejorar la experiencia de usuario sin comprometer el rendimiento del navegador. La interfaz se dividió en tres paneles lógicos: información académica, área de interacción (video) y métricas de juego.

C. Lógica de Negocio (JavaScript)

Se desarrolló un algoritmo en JavaScript encargado de orquestar el funcionamiento de la aplicación:

- **Gestión Asíncrona:** Uso de funciones `async/await` para la carga eficiente del modelo y la inicialización de la cámara web.
- **Bucle de Predicción:** Implementación de `requestAnimationFrame` para analizar el flujo de video fotograma a fotograma.

- **Máquina de Estados:** Control de los estados del juego (Espera, Conteo Regresivo, Análisis, Resultado). El algoritmo compara la clase predicha por la IA (con mayor probabilidad) contra una selección pseudoaleatoria de la CPU para determinar la victoria, derrota o empate.

9. Resultados Obtenidos

La implementación final resultó en una aplicación web estable y funcional. Las pruebas de rendimiento demostraron que el sistema es capaz de identificar los gestos definidos con un nivel de confianza superior al **85%** en condiciones de iluminación estándar.

Interfaz de Usuario: La aplicación despliega correctamente la información académica y los paneles de interacción, manteniendo una estética profesional. *(Insertar captura de pantalla de la interfaz general)*

Funcionamiento del Reconocimiento: Al presentarse un gesto ante la cámara, el sistema actualiza la etiqueta de predicción en tiempo real. *(Insertar captura de pantalla detectando un gesto)*

Mecánica de Juego: El ciclo de juego se ejecuta sin interrupciones. El algoritmo determina correctamente al ganador, actualiza el marcador global y agrega el evento al historial de jugadas. *(Insertar captura de pantalla mostrando el resultado de una partida)*

10. Conclusión

El desarrollo de este proyecto ha permitido corroborar la viabilidad técnica de integrar modelos de inteligencia artificial en aplicaciones web ligeras. Se ha demostrado que herramientas como TensorFlow.js democratizan el acceso a capacidades avanzadas de visión computacional, permitiendo implementaciones complejas sin infraestructura de servidor dedicada (*backend*).

Se concluye que la calidad y variedad del conjunto de datos de entrenamiento (*dataset*) es el factor crítico para la precisión del sistema; la inclusión de una clase de "Fondo" resultó indispensable para la estabilidad del modelo. Asimismo, la arquitectura de procesamiento en el lado del cliente demostró ser eficiente para aplicaciones que requieren interacción en tiempo real.

Este trabajo satisface los requerimientos académicos de la asignatura de Big Data, evidenciando la aplicación práctica de conceptos teóricos en un producto de software tangible y funcional.

11. Referencias Bibliográficas

- Abadi, M., et al. (2016). *TensorFlow: A System for Large-Scale Machine Learning*. OSDI.
- Google. (2023). *Teachable Machine: Train a computer to recognize your own images*. Recuperado de: https://teachablemachine.withgoogle.com/
- TensorFlow. (2024). *TensorFlow.js Guide*. Recuperado de: https://www.tensorflow.org/js
- MDN Web Docs. (2024). *MediaStream Recording API*. Mozilla Developer Network.