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

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.

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 no controladas, asegurando una respuesta inmediata para la experiencia del usuario.

3. Objetivos

3.1 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.

3.2 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.

4. Marco Teórico

4.1 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.

4.2 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.

4.3 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.

5. Metodología de Desarrollo

El ciclo de vida del proyecto se estructuró en las siguientes fases técnicas:

5.1 Fase de Adquisición y Entrenamiento

Se utilizó la herramienta Google Teachable Machine para la captura y etiquetado de muestras. Se recopilaron cientos de imágenes por categoría para asegurar la robustez del modelo ante variaciones de posición y ángulo. Las clases definidas fueron:

1. **Clase A:** Piedra (Puño cerrado).
2. **Clase B:** Papel (Palma extendida).
3. **Clase C:** Tijera (Dedos índice y medio extendidos).
4. **Clase D:** Fondo (Ausencia de gesto para evitar falsos positivos).

5.2 Fase de Exportación e Integración

Tras validar una precisión superior al 85% en el conjunto de prueba, el modelo fue exportado en formato compatible con TensorFlow.js, generando los archivos de arquitectura (`model.json`) y pesos sinápticos (`weights.bin`). Estos archivos fueron alojados localmente en la estructura del proyecto.

5.3 Fase de Desarrollo de Software

Se codificó la aplicación web utilizando estándares modernos:

- **Estructura (HTML):** Diseño semántico dividido en paneles de información, visualización y métricas.
- **Estilo (CSS):** Implementación de diseño responsivo y estética *glassmorphism* para mejorar la usabilidad.

- **Lógica (JavaScript):** Implementación de funciones asíncronas (`async/await`) para la carga del modelo y gestión del bucle de predicción (`requestAnimationFrame`), asegurando un rendimiento de 60 cuadros por segundo.

6. Resultados

La implementación final resultó en una aplicación web funcional y estable. Las pruebas de rendimiento indican una latencia mínima en la detección de gestos.

A. Interfaz de Usuario: La aplicación despliega correctamente la información académica y los paneles de interacción. La integración gráfica opera fluidamente en navegadores basados en Chromium y Gecko. *(Insertar aquí captura de pantalla de la interfaz inicial)*

B. Precisión del Reconocimiento: El sistema demuestra capacidad para identificar los gestos con un nivel de confianza (probabilidad) adecuado, actualizando la etiqueta en tiempo real conforme el usuario cambia la posición de la mano. *(Insertar aquí captura de pantalla detectando un gesto específico)*

C. Funcionalidad del Juego: El algoritmo de comparación lógica determina correctamente las condiciones de victoria, derrota o empate, actualizando el historial de eventos y el marcador global sin errores de ejecución. *(Insertar aquí captura de pantalla mostrando el resultado de una partida)*

7. 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.

8. 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.