UNIVERSIDAD AUTÓNOMA DE CIUDAD JUÁREZ

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación

APLICACIÓN BASADA EN VISIÓN ARTIFICIAL PARA DETECTAR JUGADAS EN EL JUEGO DE BLACKJACK

Reporte Técnico de Investigación presentado por:

Edgar Vicente Castañeda Chávez 117921

Requisito para la obtención del título de:

INGENIERO EN SISTEMAS COMPUTACIONALES

Asesor:

Dr. Josué Domínguez Guerrero

Ciudad Juárez, Chihuahua 4 de junio de 2022

Ciudad Juárez, Chihuahua, a 4 de Junio de 2022

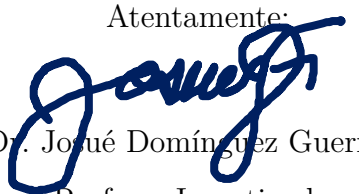
Asunto: Liberación de Asesoría

Mtro. Ismael Canales Valdiviezo
Jefe del Departamento de Ingeniería
Eléctrica y Computación
Presente.-

Por medio de la presente me permito comunicarle que, después de haber realizado las asesorías correspondientes al reporte técnico APLICACIÓN BASADA EN VISIÓN ARTIFICIAL PARA DETECTAR JUGADAS EN EL JUEGO DE BLACKJACK, del alumno Edgar Vicente Castañeda Chávez de la Licenciatura en Ingeniería en Sistemas Computacionales, considero que lo ha concluido satisfactoriamente, por lo que puede continuar con los trámites de titulación intracurricular.

Sin más por el momento, reciba un cordial saludo.

Atentamente:



Dr. Josué Domínguez Guerrero
Profesor Investigador

Ccp: Mtro. David Absalón Uchurtu Moreno
Coordinador del Programa de Sistemas Computacionales
Edgar Vicente Castañeda Chávez
Archivo

Ciudad Juárez, Chihuahua, a 4 de Junio de 2022

Asunto: Autorización de publicación

C. Edgar Vicente Castañeda Chávez

Presente.-

En virtud de que cumple satisfactoriamente los requisitos solicitados, informo a usted que se autoriza la publicación del documento de APLICACIÓN BASADA EN VISIÓN ARTIFICIAL PARA DETECTAR JUGADAS EN EL JUEGO DE BLACKJACK, para presentar los resultados del proyecto de titulación con el propósito de obtener el título de Licenciado en Ingeniería en Sistemas Computacionales.

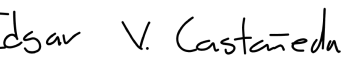
Sin otro particular, reciba un cordial saludo.

Dr. Everardo Santiago Ramírez

Profesor Titular de Seminario de Titulación II

Declaración de Originalidad

Yo, Edgar Vicente Castañeda Chávez declaro que el material contenido en esta publicación fue elaborado con la revisión de los documentos que se mencionan en el capítulo de Bibliografía, y que la solución obtenida es original y no ha sido copiada de ninguna otra fuente, ni ha sido usada para obtener otro título o reconocimiento en otra institución de educación superior.



Edgar Vicente Castañeda Chávez

Agradecimientos

Se agradece a mi asesor el Dr. Josué Domínguez Guerrero, sobre todo por su comprensión y ayuda. Gracias a todas las personas que directa o indirectamente me han ayudado con consejos y palabras de aliento.

Dedicatoria

A mi madre, mi principal fortaleza y ejemplo de perseverancia y tenacidad. A mi padre al que no fue posible despedirme, siempre lo tengo en mi mente.

Resumen

El presente proyecto de titulación comprende la realización de una aplicación basada en visión por computadora para detectar jugadas en el juego de Blackjack. El fin del proyecto es detectar los valores y jugadas en las partidas de este juego de mesa, para facilitar la labor de operadores de videovigilancia. La aplicación consiste en estar mostrando en pantalla los valores que tienen las cartas y jugadas, también guardar el vídeo para su posterior análisis en caso que sea necesario.

Palabras clave: Visión artificial, Blackjack, videovigilancia.

Índice general

Índice de figuras

Introducción

El análisis de patrones es una tarea que siempre ha tenido un factor humano, ya que es algo que no es tan fácil automatizar, pero debido a los avances que se han hecho en las ciencias de la computación y electrónica ya es posible tener sistemas que permiten el reconocimiento de imágenes. Las redes neuronales permiten crear soluciones para el reconocimiento de imágenes y patrones, debido a ello es un área que está en crecimiento. Los trabajos de reconocimiento de imágenes y patrones se han utilizado para reconocer cartas, pues tienen rasgos bien definidos como lo son la forma y las figuras que tienen grabadas.

La mayoría de los comercios actualmente no cuentan con sistemas de vigilancia automatizados, ya que el desarrollo en el reconocimiento de patrones, tratamiento y procesamiento de imágenes digitales apenas hace unos años que se ha empezado a implementar.

El negocio de los casinos es un campo que ha empezado a crecer los últimos años en México, y aunque hay referencias que en otros países van en lo último en avances tecnológicos en los casinos locales no es así, sino que se cuenta con lo mínimo necesario para realizar una videovigilancia.

Los juegos de azar han sido una parte del entretenimiento en la historia de la humanidad, en el que se puede usar desde una simple moneda hasta sofisticadas máquinas que incorporan computadoras que muestran imágenes.

En lo que compete a este proyecto los elementos que se analizarán son las cartas de una baraja francesa, la cual se compone de 52 elementos en la que aparecen cuatro figuras de

dos colores y los juegos que se hacen con ellas son muy variados como lo es el Póquer, Texas holdém y Blackjack.

En seguida se muestra la distribución de los capítulos de este documento.

Para el capítulo I que corresponde al planteamiento del problema, se llevó a cabo una investigación de los proyectos que han abordado el problema planteado, esta exploración corresponde a los antecedentes. También se establece el objetivo general y los objetivos específicos, así mismo la justificación dada a este problema. Además, se enumeran los alcances y limitaciones para el proyecto propuesto.

En el capítulo II se enuncia la teoría relacionada con el problema de detección de imágenes. Se describen conceptos relacionados con el problema y se da una explicación de ellos.

El capítulo III corresponde al desarrollo del proyecto, se describe el progreso del proyecto siguiendo la metodología de desarrollo en cascada. Se explica cómo se desarrolla el producto y las herramientas utilizadas.

Para el capítulo IV se muestran los resultados, en los que se hace la comparación de los resultados en diferentes escenarios.

Finalmente en el capítulo V se proporcionan las conclusiones del proyecto y se dan algunas recomendaciones a futuro.

Capítulo 1
Planteamiento del Problema

El monitoreo es una tarea relativamente nueva que se lleva a cabo mediante circuito cerrado de televisión (CCTV), en la que se cuenta con personal especializado que se dedica a observar cámaras colocadas de una forma estratégica, de modo que se pueda observar un área y ser grabada las 24 horas. El trabajo del monitorista es una tarea monótona, y por lo tanto proclive a que se pierdan detalles en la observación de las imágenes.

Los casinos son un ambiente en el que se adoptó el uso de sistemas de CCTV, para tener una vigilancia de sus instalaciones. Empero que sea una tecnología ampliamente adaptada, todavía no se tienen en la mayoría sistemas automatizados para detectar actividades específicas.

1.1. Antecedentes

Para el desarrollo de este proyecto se hace uso de técnicas de tratamiento y análisis de imágenes para automatizar las tareas de identificar jugadas que se realizan en un juego de cartas llamado Blackjack. En trabajos que guardan cierta similitud, se ha hecho uso del procesamiento de imágenes, ya que es necesario aislar ciertos parámetros básicos como lo son, identificar bordes o localizar formas individuales.

Las técnicas del procesamiento digital de imágenes consisten en la segmentación de imá-

genes, en la que se detectan los bordes, y también se hace un tratamiento de umbralización que consiste en convertir una imagen que tenga varios tonos a una en blanco y negro [1].

En la detección de cartas se aplica el procesamiento de imágenes, pero para la identificación de las fichas de pago hay un reto especial para identificar su valor, ya que éstas solo se distinguen por su color [2]. Un detalle por destacar es el movimiento que se hace con las fichas, y es algo que se ha necesitado estudiar y dar una solución. Esto es el movimiento que hace el empleado al entregar o recoger fichas, por lo que se requiere que se tenga cierta uniformidad en la forma en que desplazan las fichas, así también en cómo las coloca. En el desplazamiento de las cartas y fichas que realiza el empleado de la mesa de juego, ya existen patentes que han sugerido soluciones al problema que se tiene cuando el personal reparte o recoge las fichas y cartas en los juegos, ya que es proclive a equivocarse al momento de cobrar o pagar una jugada ganadora [3].

En el reconocimiento de cartas ya se ha analizado el problema, pues ya hace años que se tiene la tecnología para su desarrollo, como lo fue introducir dispositivos de radiofrecuencia en las cartas y las fichas [4], además del uso del reconocimiento de cartas usando visión por computadora. Existen trabajos que se han basado en los resultados obtenidos por [5] en los que se han mejorado los resultados, pues se hace un análisis más profundo del método empleado de reconocimiento de cartas [6]. En estos trabajos se han usado diversas tecnologías y algoritmos que han analizado el problema de la orientación y clasificación en una baraja francesa para el juego de Texas Holdém en los cuales se emplearon cámaras que capturan todos los movimientos que se realizan en la mesa de juego [7]. Las herramientas usadas que han sido creadas por la disciplina de la visión artificial se han probado en diversos proyectos y se les ha dado solución usando algoritmos tal como lo es el *Scale-Invariant Feature Transform (SIFT)* [8].

1.2. Definición del problema

Al ser una tarea tan repetitiva y aburrida, las personas que se dedican a observar el juego de Black Jack pierden el interés en lo que está pasando en las pantallas de monitoreo, lo cual hace que no se percaten de los errores que están ocurriendo en el juego. Todo este proyecto está motivado porque se gasta mucho tiempo buscando errores que están grabados por vídeo, y también se omiten muchos otros porque el operador de monitores no los logra captar en el momento que ocurren.

1.3. Objetivo general

Desarrollar un asistente de monitoreo automático para mostrar en la pantalla los valores y jugadas que se llevan a cabo en el juego de Black Jack, éste consta de una interfaz donde se visualiza un vídeo de entrada, el cual puede que esté almacenado en algún medio o que sea directo de la cámara web.

1.3.1. Objetivos específicos

- Implementación de herramientas de reconocimiento de imágenes que usan redes neuronales para identificar cartas de la baraja francesa.
- Identificar jugada ganadora mediante algoritmos de redes neuronales convolucionales (CCN, por sus siglas en inglés).

1.4. Justificación

Las tecnologías creadas a partir de los campos de reconocimiento de patrones y reconocimiento de imágenes se pueden adecuar a los sistemas de videovigilancia. Actualmente,

muchos de los establecimientos solo cuentan con un sistema de grabación de vídeo, pero lo que se busca implementar ahora es que se detecten ciertos patrones de movimiento en las mesas de juego, uno en particular que es el Blackjack. Se tendrá un impacto tecnológico debido a que no existen sistemas en los casinos que automaticen las tareas de monitoreo, además podrán canalizar los recursos humanos hacia otras áreas que estén más descuidadas, como por ejemplo, ver las conductas de los clientes. Se tendrá un impacto económico, ya que se buscará reducir los errores al momento de pagar, y los clientes estarán más satisfechos.

1.5. Alcances y limitaciones

El proyecto abarca los siguientes aspectos.

- El proyecto está pensado para ser una simulación.
- Se aplican técnicas de inteligencia artificial, como lo son las redes neuronales convolucionales para reconocimiento de imágenes.

Los siguientes puntos quedan fuera del proyecto.

- Características técnicas de las cámaras usadas para la simulación.
- El tipo de formato de grabación que se tenga.
- El sistema de cómputo con que se cuente.

Capítulo 2
Marco teórico

La inteligencia artificial es un campo que en las últimas décadas ha crecido mucho, en este campo existe un subcampo que es el de aprendizaje automático que viene del inglés *machine learning*. Este campo se ocupa en desarrollar algoritmos, en los cuales se busca que aprendan al igual que lo hacen los humanos, pero usando términos matemáticos. *Deep learning* es otro término en inglés, al que ahora llamaremos aprendizaje profundo, el cual está adentro del aprendizaje automático y su principal elemento es la neurona artificial.

2.1. Inteligencia artificial

La inteligencia artificial está mal entendida en la época actual debido al enfoque que se ha dado en la literatura y cine, ya que se tiende a exagerar en cuanto a lo que se estudia en ella. Esto es debido a que, si bien muchos de los aspectos que entran en el campo de la inteligencia artificial se inspiran en procesos biológicos como el cerebro, no siempre se toma en cuenta que muchos aspectos que se dan por sentado como humanos no se pueden resolver mediante una computadora, pues la computadora está limitada por su arquitectura y solo puede trabajar con instrucciones bien definidas. Este mal entendido también se debe a que en la sociedad todavía no se tiene un concepto bien definido de inteligencia, ya que se involucran bastantes conceptos, entre los que se destacan los procesos de aprendizaje, razonamiento, entendimiento, visualización de relaciones, entre otras muchas más. Se puede

concluir que todos estos elementos tienen que ver con la habilidad de obtener, manipular y dar sentido a la información qué con la que se tiene interacción [9].

La inteligencia artificial es una disciplina relacionada con la teoría de la computación cuyo objetivo es emular algunas de las facultades intelectuales humanas en sistemas artificiales [10] y se construyó con base en conocimiento y teorías que ya existen en otros campos como las matemáticas, neurociencia y computación. Cada una de las disciplinas han agregado componentes para crear esta área, un claro ejemplo es que las redes neuronales artificiales fueron creadas por inspiración en el estudio del cerebro, pues se ha tratado de emular en cierto sentido a este órgano y de paso se busca que con ellas se pueda incrementar el entendimiento de las funciones de las redes neuronales biológicas[11].

Al tratar de simular el cerebro humano y su forma de trabajar, los algoritmos que se usan para resolver problemas en la inteligencia artificial se han inspirado en cierta forma en la naturaleza y está relacionada con lo que se percibe mediante nuestros sentidos [10]. Uno de los sentidos con los que contamos gran variedad de seres vivos es la vista, y este sentido se ha tratado de emular en el ambiente computacional, aunque el desarrollo de las cámaras digitales se ha llevado a cabo apenas en las últimas décadas. Cuando se analiza la inteligencia artificial se tiene en mente que los algoritmos desarrollados cumplan con alguno de estos objetivos:

Actuar como humano. En este punto se trata de emular la inteligencia, de tal modo que no se pueda distinguir si es algo que hace una máquina o un humano, lo cual es el famoso test de Turing. El fin de la inteligencia artificial es no tratar de recrear los procesos de inteligencia humana completamente, sino que solo servirá de guía para realizar una simulación que logre lo mismo que los humanos.

Pensar como humano. En esta parte ya se requiere inteligencia, y para emular este proceso y así pensar como humano podemos apoyarnos en las técnicas de:

- La introspección. Consiste en que se les dé seguimiento a los pensamientos y tratar de

emularlos.

- Pruebas psicológicas. Se hace una base de datos que se crea mediante la observación de sujetos que tengan comportamientos similares en circunstancias, objetivos, recursos y ambientes que comparten todos.

- Escaneo del cerebro. En esta técnica ya se requiere equipo para monitorizar las actividades cerebrales.

Pensar racionalmente. En esta parte se busca que una máquina pueda actuar acorde a datos guardados para interactuar con el ambiente basándose con los datos que se tienen inmediatamente. El objetivo sería resolver los problemas de forma lógica de ser posible, y en algunos casos solo se crea una base para encontrar la solución que posteriormente podría ser modificada para ser utilizada. En lo único en lo que se podría diferenciar un humano al método utilizado por una máquina es que en nuestras normas a seguir sí se suele tener algunos niveles de desviación, es decir, que con base en experiencias u otros factores se pueden cambiar nuestros comportamientos o decisiones.

Actuar racionalmente. Se basa en analizar cómo se actúa en diversas situaciones, tomando en cuenta ciertas restricciones y así determinar cuáles son las mejores técnicas para resolver el problema [9].

2.2. Aprendizaje automático

Un campo que entra en la inteligencia artificial es el aprendizaje automático (*machine learning*) el cual consiste en dar ejemplos a un algoritmo para que resuelvan un problema dado [12].

El aprendizaje sucede como un resultado de un análisis de datos cada vez mayores, por lo que los algoritmos pueden no tener cambios, sin embargo, el código interno, los pesos y

sesgos usados sí lo hacen. Este procedimiento comprende muchas técnicas que permiten a la computadora aprender de datos proporcionados y así dar una solución o respuesta. Hace uso de análisis estadístico para encontrar analogías en los datos. El objetivo final es crear una simulación de aprendizaje humano mediante algoritmos que analicen conjuntos de datos muy grandes para que una aplicación se pueda adaptar a condiciones no precisas o inesperadas.

El aprendizaje automático tiene una fuerte carga matemática, pues es posible representar el entorno circundante usando funciones matemáticas que, aunque el algoritmo todavía no conoce, posteriormente podrá adivinar después de ver los datos, siempre en una forma de datos emparejados de entradas y salidas. Puesto en otras palabras, el algoritmo tiene funciones matemáticas que tienen ciertos parámetros que son modificables para ajustarse a la información que se ha tomado de los datos [9].

2.3. Aprendizaje profundo

Dentro del campo del aprendizaje automático hay otra área de investigación denominada aprendizaje profundo (*Deep learning*) que se caracteriza por el esfuerzo de crear un modelo de aprendizaje de varios niveles, en el que el nivel más profundo toma como entrada la salida del nivel anterior [13].

Trata de simular el cerebro humano y la forma cómo funciona, los datos los procesa usando unas células de cómputo a lo que llamamos neuronas, organizadas en secciones que llamamos capas. Las capas son un tipo de ajuste para optimizar el algoritmo y dependiendo de la red neuronal el número de capas puede ser muy grande y tiene la capacidad de aprendizaje especializado. El término de profundo tiene sentido cuando pensamos en que puede haber muchas capas que se utilizan en un algoritmo, pues al crearlo se hace un ensamble de diferentes neuronas en las que se relacionan en capas.

Una de las diferencias que se hacen del aprendizaje automático y del aprendizaje profun-

do es la generación de características, que son la creación de la información correcta para alimentar un algoritmo de aprendizaje automático, ya que en el aprendizaje profundo, gracias a que dispone de capas, éstas pueden definir mejor sus características [9].

2.4. Redes neuronales convolucionales

Las redes neuronales convolucionales han sido diseñadas específicamente para el reconocimiento de imágenes, y son un tipo especializado de redes neuronales para el procesamiento de datos conocido como topología en red. El nombre de redes convolucionales es porque se emplea una operación matemática llamada convolución [14]. La cual se lleva a cabo en dos funciones para generar un mapa de características, donde la primera función *I(t)* son los datos de entrada y *K(a)* es el filtro [15] (Véase la Ecuación 2.1).

$$\sum \alpha^N I(a) * K(t-a) \tag{2.1}$$

El término fue acuñado por primera vez en 1998 en Fukushima y posteriormente se le hicieron mejoras por parte de LeCun, Bottou y Bengio and Haffner. Gracias a esta investigación se desarrolló una arquitectura para el reconocimiento de caracteres hechos a mano al que denominaron LeNet-5 que ha sido muy usado para enseñar las bases de las redes neuronales convolucionales. Esta arquitectura se divide en tres tipos de capas, que son:

- Capa de convolución. El propósito que persiguen las capas de convolución es la detección de características como lo son las esquinas, líneas, puntos de color, entre otros tantos elementos visuales. Hay unos parámetros que se tienen que pasar en la mayoría de los marcos de trabajo de redes neuronales, que son el número y tamaño de filtro, porque estos son muy importantes, pues son los que se encargan de detectar las características. Algunos de los filtros que se suelen usar son los que se muestran en las siguientes matrices:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{2.2}$$

$$\begin{bmatrix} 0 & 1 & 0 \\ -1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.3}$$

- Capa de agrupamiento máximo. Esta capa lo que hace es que las dimensiones de la imagen se hagan más pequeñas, además de remover detalles y al no tener pesos el entrenamiento de las redes neuronales no le afectan.

- Capas densas. En este punto se conecta cada elemento o neurona con la capa anterior y el resultado se pasa a través de una función de activación que puede ser sigmoidea, hiperbólica o por Unidad de Rectificación Lineal (ReLU, por sus siglas en inglés). Estas funciones son usadas para obtener el resultado de un nodo con una respuesta de un sí o un no [16].

$$y = \frac{1}{1 + e^{-x}} \tag{2.4}$$

$$y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.5}$$

$$y = max(0, x) \tag{2.6}$$

2.5. Procesamiento de imágenes

En el procesamiento de imágenes se inicia con la etapa de adquisición de imágenes, en donde se requiere de un sensor de imágenes, cuyas señales producidas deben ser digitalizadas. La siguiente etapa es el preprocesamiento, que se realiza con el fin de detectar y eliminar las

fallas que puedan existir en la imagen para mejorarla. Las técnicas más utilizadas en esta etapa son:

a) Mejora del contraste

b) Eliminar el ruido

c) Restauración

En la siguiente etapa, que es la segmentación, la imagen se divide en sus partes constituyentes u objetos, con el fin de separar las partes necesarias de procesamiento del resto de la imagen que no interesan de acuerdo a la aplicación que se quiera dar. Las técnicas básicas en esta etapa son aquellas orientadas a:

a) El píxel

b) Los bordes

c) Las regiones

Sin embargo, las técnicas no son excluyentes, sino que se combinan de acuerdo al tipo de aplicación [1].

Capítulo 3
Desarrollo del Proyecto

El capítulo contiene la descripción de la implementación del proyecto, se realiza el desglose de la idea del diseño, se describen las tecnologías a utilizar y su instalación. Se realiza una descripción de las principales características del producto.

3.1. Producto propuesto

La solución se lleva a cabo aprovechando las utilidades de las redes neuronales convolucionales, una computadora con prestaciones de hardware estándar y herramientas de software open source. Debido a que el software utilizado es libre, el desarrollo es llevado a cabo en los sistemas operativos Windows y Linux, ya que todas las utilidades son compatibles en ambos ambientes.

La arquitectura de la herramienta YOLOv3 [17] está basada en el modelo de GoogLeNet para la clasificación de imágenes[18], que a su vez está basada en el algoritmo de redes neuronales convolucionales para reconocimiento visual, desarrollado en 1989. YOLOv3 cuenta con una amplia documentación, y aunque tiene pocas clases predefinidas, esto es, objetos que puede identificar sin un entrenamiento previo, tiene la ventaja que es fácil crear modelos personalizados para detectar nuevas clases.

El hardware utilizado para la ejecución y pruebas de este proyecto es muy elemental,

para esta solución es necesaria una cámara para captura de vídeo, una computadora con un procesador de arquitectura de 64-bit, 16 GB de memoria RAM DDR4 y almacenamiento de al menos 1 TB, esto es lo mínimo necesario para que el software se ejecute satisfactoriamente. Adicionalmente, se sugiere que cuente con una tarjeta dedicada al procesamiento de vídeo, ya que con esto el software no será dependiente solamente de las capacidades del CPU.

La solución propuesta tiene la capacidad de reconocer cartas individuales que se van colocando en la mesa de juego. En pantalla se muestran los valores individuales de cada carta, posteriormente se hace uso de los datos obtenidos para identificar jugadas ganadoras.

3.2. Descripción de la metodología

La metodología seleccionada para el desarrollo del proyecto es la de cascada. Esto debido a su forma secuencial que permite ir concatenando fases como se muestra en la Figura 3.1.

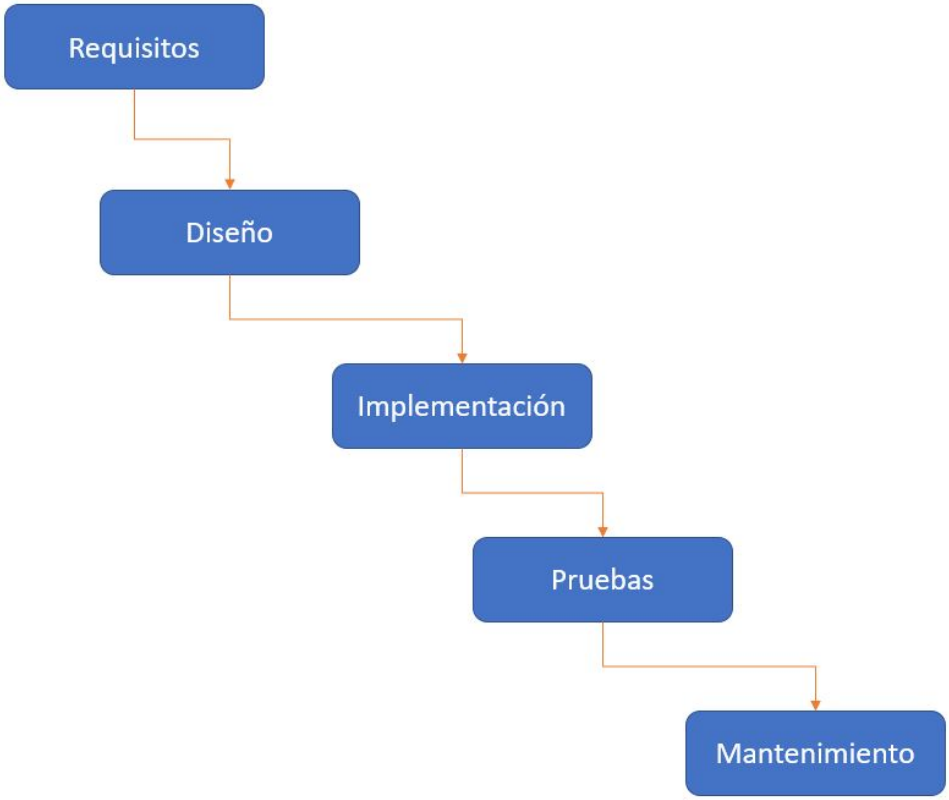


Figura 3.1: Metodología en cascada para el desarrollo de la aplicación.

Los pasos seguidos por el método en cascada se enumeran como sigue:

- Los requisitos del sistema vienen dados de un previo análisis del problema, por lo que se es necesario tener una buena comprensión de este.
- El diseño es la etapa en la que se hace un esbozo de lo que se ha solicitado del software, en la que se representa la arquitectura del software y cómo es que está representada la interfaz.
- En la implementación se transcribe el diseño ideado para que se pueda ejecutar en la computadora.

- Las pruebas es la validación de los procesos lógicos del programa, buscando que se eliminen la mayoría de los errores.
- El mantenimiento viene dado como un mecanismo de seguir depurando el software, por ejemplo, que algún requerimiento presentase algún cambio, o arreglar algún error que no se detectó en la etapa de pruebas.

3.3. Requisitos

Observando cómo es que se desarrolla el juego de Blackjack es que se logran aislar los requerimientos del producto, también con la interacción que se tuvo con el personal de vigilancia se pudo observar cuál es el equipo que se utiliza y cuáles son los ambientes en los que se ejecutan sus programas. Con la información recabada se desglosan los siguientes requisitos.

Requisitos funcionales:

- El sistema proporciona el número individual de cada carta en la mesa.
- Las jugadas se evalúan mediante la suma que se asignó a cada carta.

Requisitos no funcionales:

- El valor mostrado de las cartas solo será mostrado si cuenta con una precisión arriba del 85 %.
- Los equipos utilizados son estaciones de trabajo de torre.
- El tiempo de actualización del vídeo es de 30 cuadros por segundo.
- Los ambientes en los que se utiliza el software es mayormente Windows.

3.4. Diseño

El software tiene el propósito de estarse ejecutando continuamente, mostrando los valores individuales de cada carta y su valor total como jugada de forma constante, todo lo que se presenta en la interfaz también se está guardando en disco por si se desea realizar una revisión posteriormente.

En pantalla se están mostrando los valores de cada carta, se hace una agrupación de los valores por secciones de la pantalla; las cuales representan las jugadas: Posteriormente, se suman los valores asignados por sección y se visualiza la suma total; la cual representa la jugada, así como se muestra en la Figura 3.2.

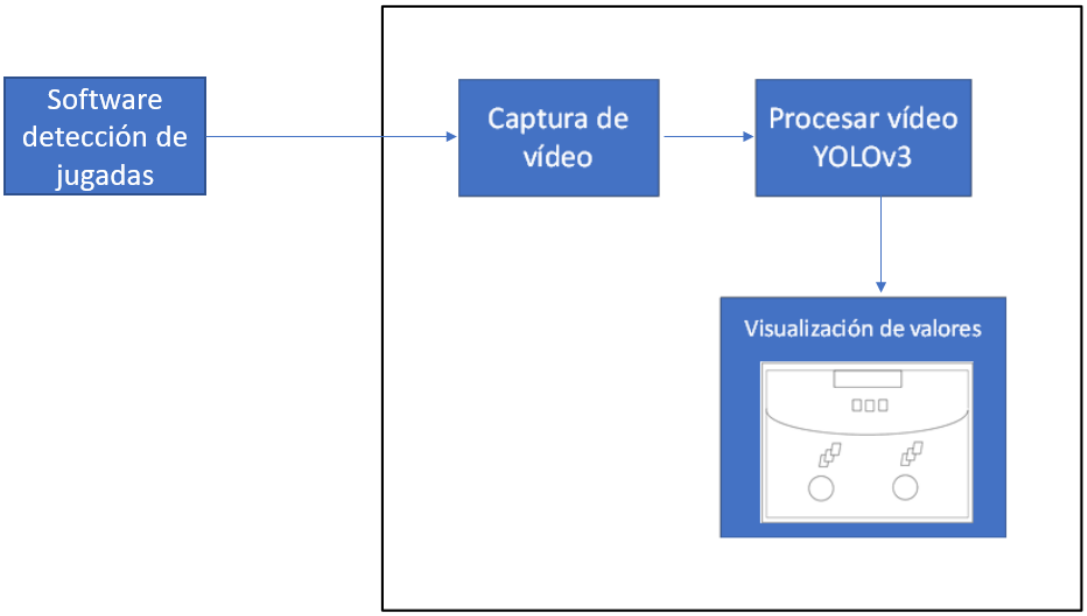


Figura 3.2: Diagrama conceptual para la identificación de cartas.

El modelo utilizado para detectar los valores de las cartas es YOLOv3, este modelo se puede implementar en diversos lenguajes de programación. Para este proyecto se implementa en el lenguaje de programación Python. Hay ciertas bibliotecas externas que es necesario

instalar, como lo son OpenCV, PyTorch, Pillow y otras que ya están integradas en este lenguaje.

La finalidad del juego es tener una puntuación que más se acerque al 21 sin pasarse. El juego se lleva a cabo en una mesa de hasta seis jugadores en la que el crupier compite contra cada jugador. Al inicio de la partida se reparten a los jugadores dos cartas que muestran sus valores, a partir de ahora denominadas boca arriba; el crupier tiene una carta que no muestra los valores, la cual se dice que está boca abajo y la otra carta boca arriba. Los términos previamente mencionados son parte del vocabulario del juego y pueden no ser conocidos para el público que conoce los juegos de cartas.

Las reglas básicas son:

- Los jugadores se sientan y colocan sus apuestas en sus respectivas casillas.
- Los jugadores reciben 2 cartas boca arriba.
- El crupier recibe dos cartas, una boca arriba.
- Si una jugada es mayor a 21 se marca en rojo indicando que es perdedora.
- Si la jugada es 21 se muestra en verde indicando que es ganadora.
- Si la jugada es menor a la que tiene el crupier, se marca como perdedora.
- Si la jugada es mayor a la que tiene el crupier, se marca como ganadora.

Estas reglas se ilustran en forma de diagrama de flujo tal como se muestra en la Figura 3.3

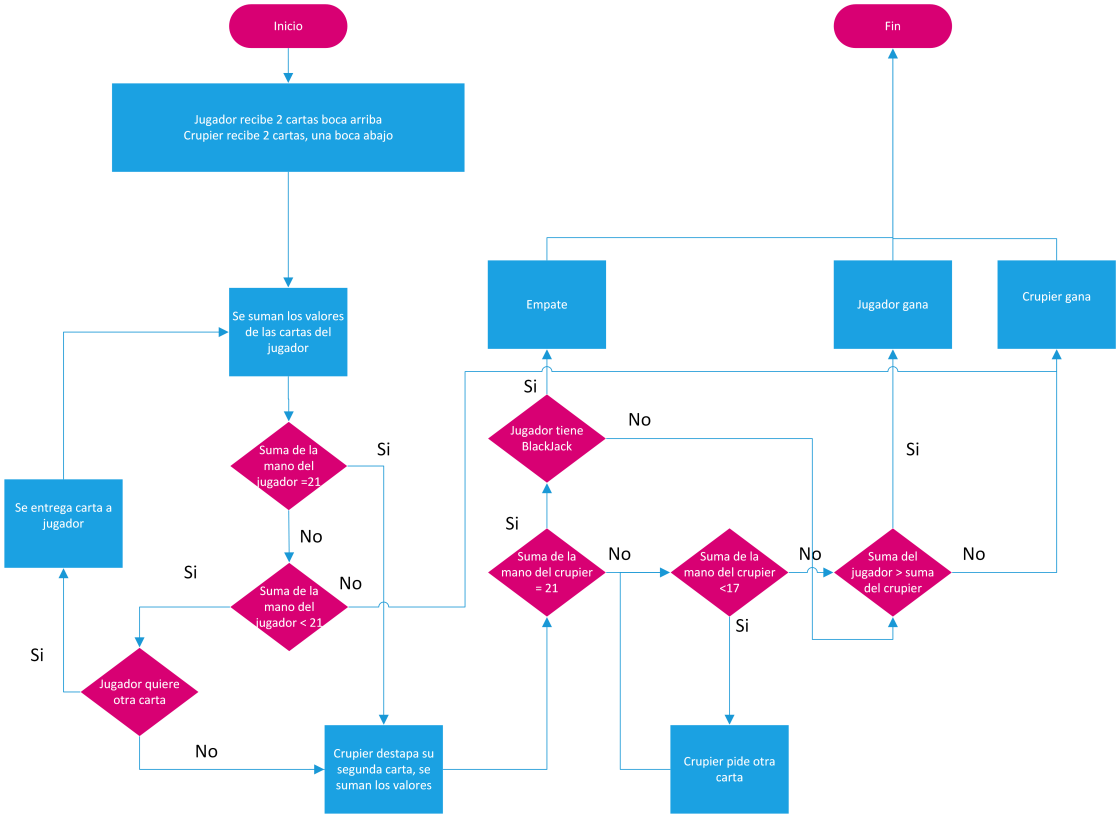


Figura 3.3: Diagrama de flujo juego de BlackJack.

3.5. Implementación

En esta sección se describe cómo se implementa la aplicación de detección de jugadas. Esta es llevada a cabo en tres partes, el etiquetado de imágenes, el entrenamiento del modelo y el despliegue de la interfaz con la que interactuará el usuario.

3.5.1. Etiquetado del conjunto de imágenes

Previo al entrenamiento del modelo para detección de cartas, se tiene que hacer una recolección de imágenes de cartas. La herramienta utilizada para etiquetar las imágenes es

labelImg, la cual es *open source* y la interfaz se muestra en la Figura 3.4.

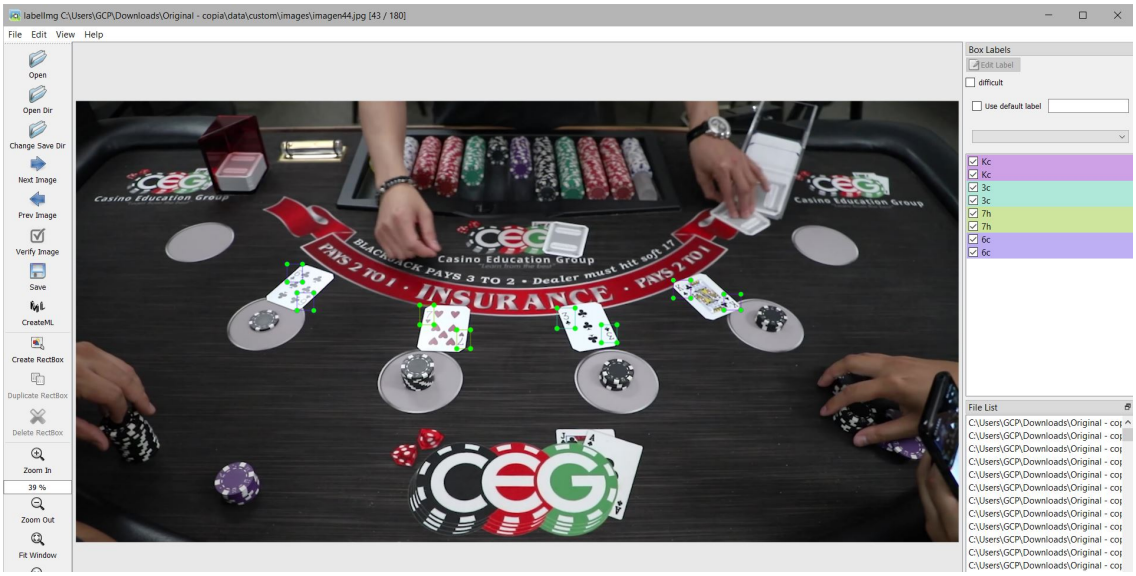


Figura 3.4: Etiquetado de imágenes usando labelImag.

La ruta donde están las imágenes tiene que ser seleccionada mediante el icono *Open Dir*, la ruta donde están las imágenes alojadas para este proyecto es en $.\backslash data\backslash custom\backslash images$.

Haciendo clic en el icono *Change Save Dir* se va a seleccionar la ruta donde se guardan los archivos generados por el software, en este caso estos archivos se guardan en la ruta $.\backslash data\backslash custom\backslash labels$.

Para cada imagen se tienen que seleccionar manualmente los elementos que se quieren identificar, en este caso es el valor que corresponde a cada carta, a estos elementos son denominados clases, tal como se muestra en la Figura 3.5.

Figura 3.5: Identificación de clases del software LabelImg.

El archivo generado se llama *classes*, el cual contiene las clases para cada una de las imágenes etiquetadas y las que vienen por defecto en el modelo YOLOv3, las rutas donde se guardan las imágenes y las etiquetas que corresponden en cada etiqueta se muestran en la Figura 3.6.

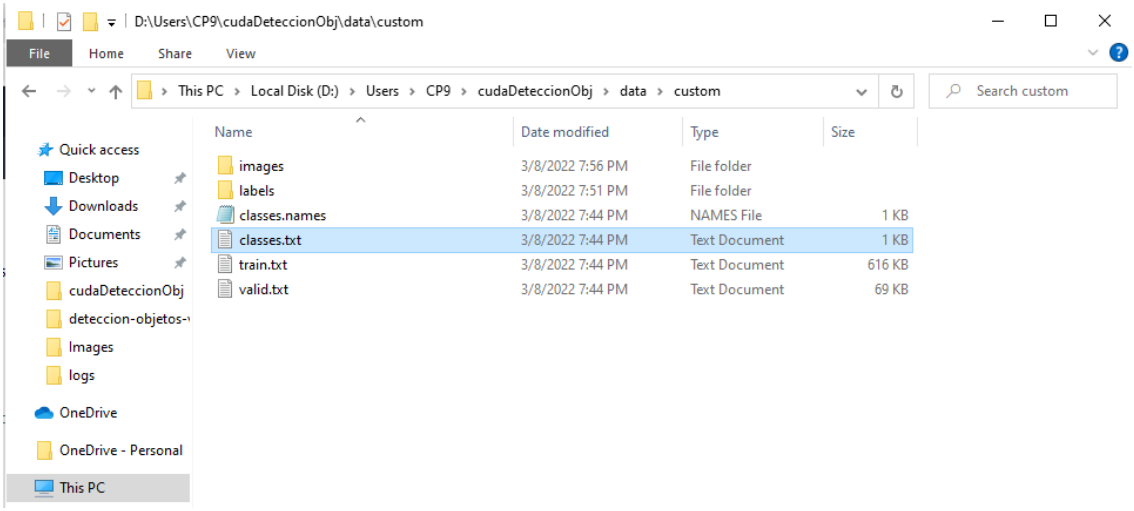


Figura 3.6: Ruta del archivo *classes*.

3.5.2. Entrenamiento del modelo

Para un primer entrenamiento se hace uso del siguiente script de Python, el cual es llamado desde una terminal de comandos. Este script realiza un entrenamiento inicial para el modelo de YOLOv3. Si nunca se ha entrenado el modelo, esta es una parte obligatoria, el código que se muestra abajo indica cómo es que se tiene que ejecutar el script.

```
1 python train.py --model_def config/yolov3-custom.cfg --data_config config/
    custom.data --pretrained_weights weights/darknet53.conv.74 --batch_size
     2
```

Los argumentos pasados son todos los elementos que siguen a la instrucción *python train.py* y tienen como prefijo – – estos son argumentos que tiene por defecto el modelo: *model_def, data_config, pretrained_weights y batch_size*. El único argumento que se puede modificar es – –*batch_size* , este valor depende de la computadora con la que se realizará el entrenamiento, ya que entre más grande sea el tamaño del lote más recursos le serán solicitados a la computadora.

Se puede empezar el entrenamiento a partir de un modelo ya existente, en caso de que no se esté todavía de acuerdo con los resultados obtenidos. En este caso solo es necesario especificar en el argumento *checkpoint_model* y *weights_path*.

```
parser.add_argument("--weights_path", type=str, default="weights/yolov3.weights", help="Ruta de los pesos del archivo")
```

```
parser.add_argument("--checkpoint_model", type=str, help="Ruta donde se guardan el último modelo entrenado")
```

Al entrenar el modelo, se van mostrando las estadísticas del entrenamiento, lo que principalmente importa es la precisión de la clasificación (*cls_acc*) y la pérdida total (*total_loss*), el lote (*batch*) que viene dado por el cociente del total de imágenes que se tiene como muestra y el tamaño del lote (– –*batch_size*), las épocas (*epoch*) es un parámetro que también se puede elegir, el cual indica la cantidad de veces que se repite el entrenamiento, así como se muestra en la Figura 3.7.

```
---- [Epoch 0/12, Batch 0/1125] ----
+------------+--------------+--------------+--------------+
| Metrics    | YOLO Layer 0 | YOLO Layer 1 | YOLO Layer 2 |
+------------+--------------+--------------+--------------+
| grid_size  | 14           | 28           | 56           |
| loss       | 0.240960     | 0.082568     | 0.077112     |
| x          | 0.010989     | 0.001088     | 0.001815     |
| y          | 0.010644     | 0.000867     | 0.002121     |
| w          | 0.002789     | 0.001643     | 0.000853     |
| h          | 0.001246     | 0.001450     | 0.000992     |
| conf       | 0.197265     | 0.071401     | 0.069333     |
| cls        | 0.018027     | 0.006118     | 0.001998     |
| cls_acc    | 80.70%       | 91.67%       | 98.33%       |
| recall50   | 0.789474     | 0.900000     | 0.966667     |
| recall75   | 0.789474     | 0.900000     | 0.966667     |
| precision  | 0.775862     | 0.830769     | 0.716049     |
| conf_obj   | 0.966843     | 0.973613     | 0.971055     |
| conf_noobj | 0.000180     | 0.000128     | 0.000122     |
+------------+--------------+--------------+--------------+
Total loss 0.4006396532058716
---- ETA 23:36:27.637691
```

Figura 3.7: Estadísticas del entrenamiento.

3.5.3. Interfaz de usuario

La aplicación es lanzada utilizando un script de Windows, el cual se usa para activar el ambiente de Python. Después de activar el ambiente, entonces se llama el script de Python que invoca la ventana que muestra nuestra aplicación. En la figura 3.8 se muestra el contenido del script de Windows.

Figura 3.8: Script de Windows para lanzar la aplicación

Debido al tamaño de las imágenes y a que las cartas se encuentran muy dispersas, es necesario fraccionar la imagen en varias regiones, estas regiones están divididas de acuerdo con el número de participantes del juego, como se muestra en la figura 3.9. Estas zonas se nombraron regiones de interés, las cuales son definidas proporcionalmente a la imagen seleccionada. De esta forma se reduce el área donde el algoritmo hace las detecciones, ya que si las imágenes de las cartas están muy diseminadas suele haber fallos en la detección.

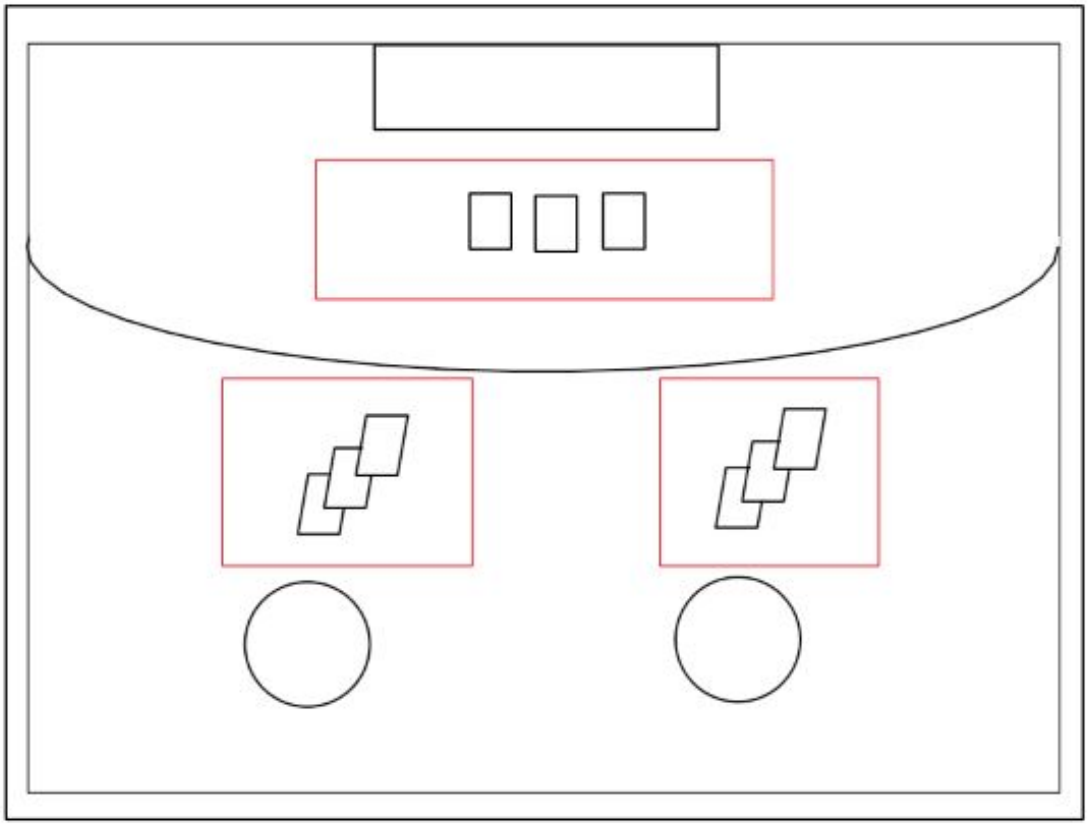


Figura 3.9: Regiones de interés.

Para seleccionar el área de interés en la que va a trabajar el algoritmo se utilizan las siguientes líneas del script *deteccion_video.py*. Hay otras tres regiones de interés creadas, solo los puntos cambian porque las regiones de interés están ubicadas en diferentes lugares de cada imagen.

```
1            punto1=(int(tamanio[1]*.30), int(frame.shape[0]/5))
2            punto2=(int(tamanio[1]*.63),int(frame.shape[0]*.4))
3            ROI=frame[punto1[1]:punto2[1],punto1[0]:punto2[0]]
4            resize_ROI=cv2.resize(ROI,(416,416))
5            cv2.rectangle(frame,punto1,punto2,(0,255,0),2)
```

La línea uno y dos son los puntos que utiliza la función *rectangle* para dibujar el área

de interés, *punto1=(x,y)* es la coordenada donde está la esquina superior del rectángulo, el *punto2=(x,y)* indica la esquina inferior del punto, como se muestra en la figura 3.10.

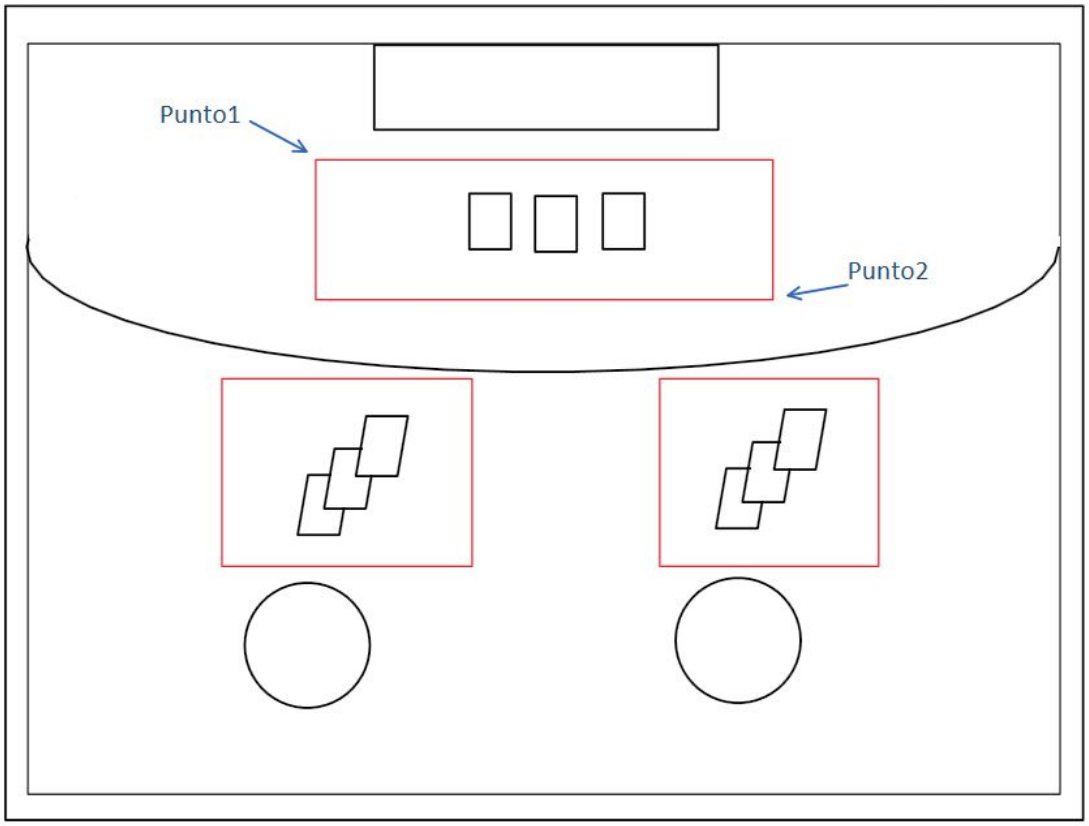


Figura 3.10: Ubicación de los puntos en la región de interés.

La variable de la línea tres utiliza los valores de las variables del punto1 y punto2 para hacer la selección del área de interés, y esta se guarda en la variable ROI. Se cambia la dimensión del área de interés en la línea 4 y finalmente se dibuja el rectángulo que es el que indica el área de interés en la interfaz. La pantalla queda dividida, como se muestra en la figura 3.11

Figura 3.11: Regiones de interés.

Capítulo 4
Resultados y Discusiones

En este apartado se muestran los resultados de la ejecución del Sistema de detección de jugadas en un ambiente experimental. Hay dos escenarios que se consideran al hacer las pruebas, y ambos entornos están relacionados en la forma que es entrenado el modelo para detección de cartas. Para ambos modelos es necesario realizar la tarea de etiquetado y entrenamiento, como se describió en el capítulo 3.

4.1. Configuración de los datos y escenarios de prueba

Para realizar las pruebas se eligió un vídeo en particular, en el cual las cartas son muy similares al conjunto de imágenes que se descargaron. También se consideró que no haya variaciones en la posición, foco e iluminación en el vídeo. El ambiente es definido como el lugar donde se va a implementar el producto, los datos se toman extrayendo los fotogramas de un vídeo previamente grabado. Gracias a que son vídeos grabados, la iluminación, disposición de los jugadores y posición de las cartas siempre tienen características muy similares, estas características son de gran utilidad, ya que con esto el conjunto de imágenes que se logran obtener son muy estables.

4.1.1. Ejecución usando modelo entrenado con imágenes descargadas de internet

Este modelo fue entrenado con un total de 19,999 imágenes que fueron descargadas del sitio https://www.kaggle.com/datasets/andy8744/playing-cards-object-detection-dataset. Ya que los vídeos que se suben a la plataforma de YouTube están realizados en un ambiente controlado, esto es que la iluminación y posicionamiento de la cámara son estables, se opta por este medio para realizar las descargas de los vídeos para hacer las pruebas del software, la descarga de los vídeos se hacen del canal Perfect Pair! , ya que cuenta con un catálogo grande de vídeos y su formato de grabación es muy estable.

El resultado obtenido usando el modelo entrenado con las 19,000 imágenes y un vídeo descargado de internet no son satisfactorios cuando no se realiza una segmentación de los fotogramas, ya que no detecta valores de ninguna de las cartas, como se muestra en la Figura 4.1.

Figura 4.1: Detección no satisfactoria

La mesa dividida en sectores ayuda al algoritmo de YOLOv3 a que las detecciones sean más certeras. Las regiones son definidas proporcionalmente al tamaño de la mesa, ya que la cámara está fija no es necesario hacer ajustes. Con la asignación de regiones de interés son logradas detecciones exitosas en pantalla, como se muestra en la figura 4.2.

Figura 4.2: Detección exitosa

4.1.2. Ejecución usando modelo entrenado con imágenes obtenidas del ambiente final

En el caso del modelo que se entrenó con las mismas cartas del vídeo, éste si detecta los valores como se muestra en la Figura 4.3. Aunque al haberse probado con un conjunto muy pequeño de cartas, muchos de estos valores están equivocados, a este tipo de anomalía es comúnmente llamada como valor con falso-positivo.

Figura 4.3: Detección con falsos-positivos

4.2. Resultados de ejecución en Windows y Linux

La ejecución del producto puede ser tanto en equipos que cuenten con hardware dedicado al procesamiento de vídeo o equipos que no tengan este hardware. Debido a restricciones del ambiente Windows, el producto solo pudo ser ejecutado usando la CPU del equipo, a diferencia de Linux, que sí, permite instalar el software necesario para poder utilizar la tarjeta de vídeo.

El equipo en situaciones normales de funcionamiento muestra los valores que se muestran en la Figura 4.4.

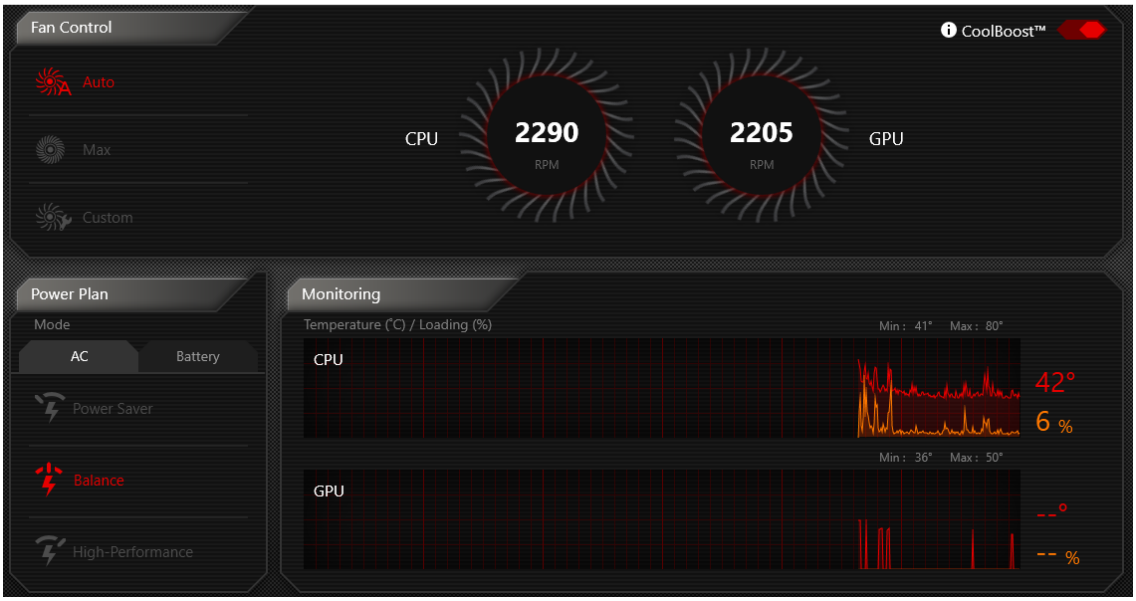


Figura 4.4: Temperatura normal del equipo

Una vez que se ejecuta el software se puede observar que el equipo muestra valores más altos a lo normal, tal como la Figura 4.5. Estos valores pueden ser sostenidos por el equipo si se cuenta con una buena refrigeración, de lo contrario se recomienda que se opte por un equipo que tenga instalado algún sistema basado en Linux.

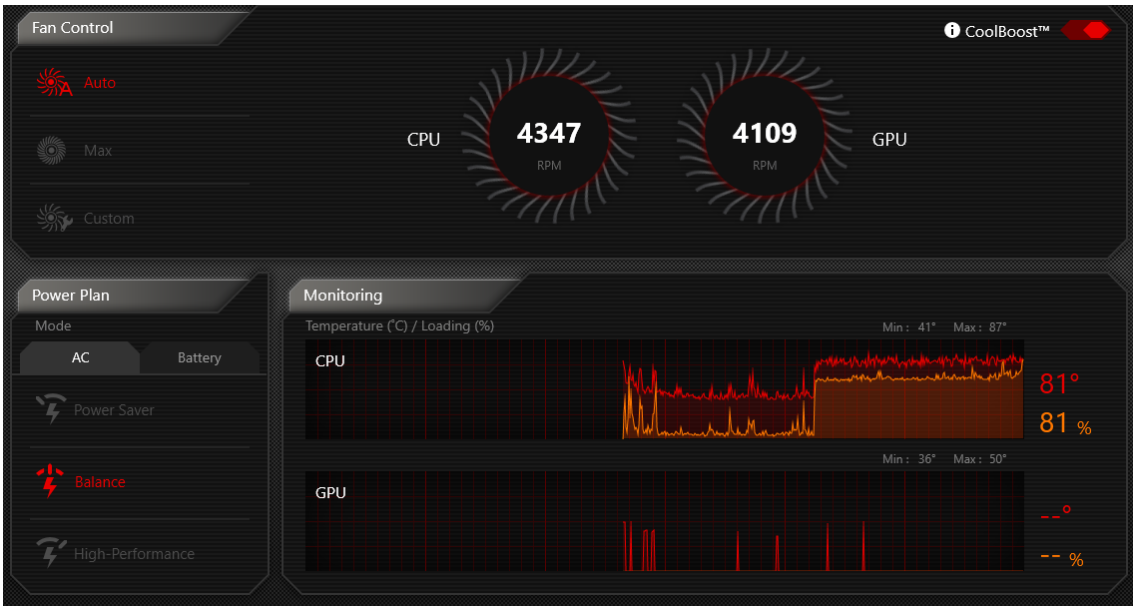


Figura 4.5: Temperatura al ejecutarse el software

Capítulo 5
Conclusiones

En este capítulo se muestra lo más relevante en cuanto al desarrollo del producto basado en lo que se tiene planteado como objetivo. Además, se agregan algunas recomendaciones a futuro para este producto.

5.1. Con respecto al objetivo general

En este documento se muestra cómo se llevó a cabo la implementación del sistema de reconocimiento de jugadas, el cual logró exitosamente su cometido, tal como se puede verificar en los capítulos 3 y 4. Con esto se concluye que el producto es viable y puede ser implementado, esto fundamentado con los resultados que se presentan en el capítulo 4 Resultados y discusiones. Lo que se concluye con el proyecto se enumera a continuación:

1. La detección de múltiples cartas (establecer qué tipo de cartas se refiere) es posible con la biblioteca YOLOv3.

2. Las alertas en tiempo real son proclives a omitirse por parte del usuario.

3. El proyecto es un apoyo directo para los trabajadores del casino que se dedican a revisar las jugadas.

4. El apoyo como material grabado facilita la búsqueda de errores, ya que no es necesario

estar haciendo las operaciones mentalmente, sino que ya aparece el valor de cada jugada.

5.2. Recomendaciones para trabajo a futuro

Existen modelos con mejor eficiencia, pero se optó por YOLOv3, debido a que este tiene más documentación a diferencia del modelo YOLOv5. En cuanto a la programación, se recomienda cambiar del lenguaje de programación Python a C++, con esto se puede lograr una mayor eficiencia en la detección de los elementos. También se considerará el análisis del pago de las jugadas en posteriores actualizaciones al software.

Bibliografía

[1] N. L. S. Palomino and U. N. R. Concha, "Técnicas de segmentación en procesamiento digital de imágenes," *Revista de investigación de Sistemas e Informática*, vol. 6, no. 2, pp. 9–16, 2009.

[2] P. Martins, L. P. Reis, and L. Teófilo, "Poker vision: playing cards and chips identification based on image processing," in *Iberian Conference on Pattern Recognition and Image Analysis*, pp. 436–443, Springer, 2011.

[3] Y. Shigeta, "Table game management system and game token," Nov. 14 2019. US Patent App. 16/506,346.

[4] K. Zutis and J. Hoey, "Whos counting? real-time blackjack monitoring for card counting detection," in *International Conference on Computer Vision Systems*, pp. 354–363, Springer, 2009.

[5] G. Hollinger, N. Ward, and E. C. Everbach, "Introducing computers to blackjack: Implementation of a card recognition system using computer vision techniques," *Colby College, Waterville*, 2003.

[6] C. Zheng and R. Green, "Playing card recognition using rotational invariant template matching," in *Proc. of Image and Vision Computing New Zealand*, pp. 276–281, Citeseer, 2007.

[7] D. Brinks and H. White, "Texas holdem hand recognition and analysis," 2007.

[8] N. Velasco, D. J. M. Chipantasi, V. H. Andaluz, and S. Dominguez, "Anglo-american playing cards identification based on counting each symbols and scale invariant feature transform (sift)," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 1-5, pp. 93–96, 2017.

[9] L. Massaron and J. P. Mueller, *Machine learning for dummies*. HOEPLI EDITORE, 2019.

[10] R. Benítez, G. Escudero, S. Kanaan, and D. M. Rodó, *Inteligencia artificial avanzada*. Editorial UOC, 2014.

[11] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[12] H. Daumé III, "A course in machine learning," *Publisher, ciml. info*, vol. 5, p. 69, 2012.

[13] G. Zaccone and M. R. Karim, *Deep Learning with TensorFlow: Explore neural networks and build intelligent systems with Python*. Packt Publishing Ltd, 2018.

[14] Y. Bengio, I. Goodfellow, and A. Courville, *Deep learning*, vol. 1. MIT press, 2017.

[15] W. Pedrycz and S.-M. Chen, *Deep Learning: Algorithms and Applications*. Springer, 2020.

[16] J. Heaton, *AIFH, Volume 3: Deep Learning and Neural Networks*. 2015.

[17] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.