BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA ELECTRÓNICA

MAESTRÍA EN INGENIERÍA ELECTRÓNICA, OPCIÓN INSTRUMENTACIÓN ELECTRÓNICA

Tesis para obtener el grado de

MAESTRO EN INGENIERÍA ELECTRÓNICA

SISTEMA PARA LA DETECCIÓN DE ACCESOS NO AUTORIZADOS EN UN ESTACIONAMIENTO

Presenta:

Elizabeth Xicotencatl Flores*

Asesores:

Dr. Aldrin Barreto Flores
Dr. Salvador Eugenio Ayala Raggi

*Becario CONACYT

Puebla, Pue., 15 de diciembre de 2021

Agradecimientos

Gracias a Dios por todo lo que tengo y a mis padres por lo que soy.

Agradezco a mis asesores, el Dr. Aldrin Barreto Flores y el Dr. Salvador Eugenio Ayala Raggi por su tiempo y paciencia durante toda la maestría.

A mis sinodales, el M.C. Ricardo Álvarez González, la M.C. Ana María Rodríguez Domínguez y la M.C. Selene Edith Maya Rueda por sus comentarios y sugerencias.

A la Benemérita Universidad Autónoma de Puebla por permitirme cursar la Maestría en Ingeniería Electrónica, Opción Instrumentación Electrónica y condonar la colegiatura durante mis estudios.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por la beca recibida para la realización de este proyecto, con número de CVU: 1004192.

Y por último, a mi familia, Magdaleno, Claudia y José por su incondicionalidad.

A todos ustedes, GRACIAS.

Resumen

En esta tesis se presenta un sistema de reconocimiento de matrículas, el cual propone un algoritmo para la localización y separación automática de los caracteres. Se evaluaron diferentes técnicas de procesamiento de imágenes digitales en el dominio del espacio, filtros, modelos de color, análisis de información estadística, extracción de características y arquitecturas de redes neuronales, siendo las de mejor resultado el modelo de intersección cortical con conexiones nulas, morfología, características geométricas y el perceptrón multicapa. El desempeño se evaluó en un conjunto de 660 imágenes traseras y frontales de automóviles obtenidas de diferentes sitios web compatibles con el formato asignado al estado de Puebla generando como resultado una precisión del 96.2 %. El tiempo promedio de ejecución en una Raspberry Pi 4 modelo B es de 1.8 segundos después de la captura de la imagen.

Abstract

This thesis presents a license plate recognition system, which proposes an algorithm for automatic location and separation of characters. Different digital image processing techniques were evaluated in the domain of space, filters, color models, statistical information analysis, feature extraction and neural network architectures. The best results were the intersecting cortical model with null connections, morphology , geometric characteristics and the multilayer perceptron. The performance was evaluated in a set of 660 rear and front images of automobiles obtained from different websites compatible with the format assigned to the state of Puebla, generating as a result an accuracy of 96.2 %. Average run time on a Raspberry Pi 4 model B is 1.8 seconds after image capture.

Abreviaturas

ALPR	Reconocimiento automático de matrículas vehiculares, *Automatic License Plate Recognition*
CCA	Análisis de componentes conectados, *Connected Component Analysis*
CNN	Redes neuronales convolucionales, *Convolutional Neural Networks*
DWT	Transformada Wavelet discreta, *Discrete Wavelet Transform*
EMNIST	Base de datos ampliada de NIST, *Extended Modified NIST*
ICM	Modelo de intersección cortical, *Intersecting Cortical Model*
ICMNN	Modelo de intersección cortical con conexiones nulas, *Intersecting Cortical Model with Null Connections*
K-NN	K vecinos más cercanos, *k-Nearest Neighbor*
MLP	Perceptrón multicapa, *Multilayer Perceptron*
MSE	Error cuadrático medio, *mean squared error*
NIST	Instituto Nacional de Estándares y Tecnología, *National Institute of Standards and Technology*
OCR	Reconocimiento óptico de caracteres, *Optical Character Recognition*
PCNN	Redes neuronales pulso acopladas, *Pulse Coupled Neural Networks*
RBFNN	Red neuronal de función de base radial, *Radial Basis Function Neural Network*
RCNN	Redes neuronales convolucionales basadas en regiones, *Region Based Convolutional Neural Networks*
SCW	Ventanas concéntricas deslizantes, *Sliding Concentric Windows*
SSD	Detector de disparo único, *Single Shot Detector*
SVM	Máquinas de vectores de soporte, *Support Vector Machines*

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

A través de los años el avance tecnológico le ha permitido al hombre optimizar diversos procesos utilizando inteligencia artificial, haciendo posible analizar gran cantidad de datos, identificar patrones y, con ello, formular predicciones de forma automática. Este principio ha servido para relacionar diversas áreas de interés como la óptica y la electrónica dando paso a disciplinas como la visión artificial, relacionada principalmente con técnicas de procesamiento de imágenes y aprendizaje automático. La extracción y manipulación de la información obtenida, ha logrado impulsar el desarrollo de sistemas capaces de reconocer rostros [1], restaurar imágenes [2], detectar espacios disponibles en estacionamientos [3], asistentes para diagnóstico de leucemia [4] e incluso de reconocimiento de dígitos manuscritos [5] que son almacenados para un uso posterior (registro, seguimiento, evaluación, etc.), tal es el caso de los *ALPR (Automatic License Plate Recognition)* diseñados para el reconocimiento de las matrículas de los automóviles, que debido a su amplio campo de aplicación son demandados cada vez más a nivel mundial.

El reconocimiento automático de matrículas disminuye la complejidad de gestionar tareas que solían resolverse manualmente, como por ejemplo, obtener la ubicación de automóviles robados, multar por exceso de velocidad y aquellas relacionadas con el monitoreo de estacionamientos. Generalmente este tipo de sistemas se conforman de tres procesos posteriores a la adaptación de la imagen: 1) obtención de la placa, 2) segmentación de los caracteres y su 3) clasificación individual. Las dos primeras etapas incorporan técnicas de procesamiento de imágenes en imágenes de dos dimensiones o en video, cuya evaluación se basa en la precisión y el tiempo de respuesta mientras que en la última suelen emplearse distintas configuraciones de redes neuronales. Todas las investigaciones previas realizadas en torno a este tema se han enfrentado a algún tipo de deficiencia en entornos no controlados (véase capítu-

lo 2) referentes a los cambios bruscos de iluminación que varía a cada hora del día durante la adquisición de las imágenes, orientación, diferentes rangos de distancia entre la cámara y el vehículo, número de objetos en la escena, factores ambientales, oclusión, deterioro de las placas que provoca que los caracteres no sean legibles, resolución de la cámara y el formato de las placas asignado por cada país o entidad federativa, por lo que se infiere que cada metodología es independiente y el desempeño está relacionado en gran medida a los recursos disponibles y las necesidades a cubrir durante su diseño.

En este trabajo se presenta un sistema de reconocimiento de matrículas, el cual propone un algoritmo para la localización y separación automática de los caracteres. Se evaluaron diferentes técnicas de procesamiento de imágenes digitales en el dominio del espacio, filtros, modelos de color, análisis de información estadística, extracción de características y arquitecturas de redes neuronales, siendo las de mejor resultado el modelo de intersección cortical con conexiones nulas, morfología, características geométricas y el perceptrón multicapa. El desempeño se evaluó en un conjunto de 660 imágenes traseras y frontales de automóviles obtenidas de diferentes sitios web compatibles con el formato asignado al estado de Puebla generando como resultado una precisión del 96.2 %. El tiempo promedio de ejecución en una Raspberry Pi 4 modelo B es de 1.8 segundos después de la captura de la imagen.

1.1. Justificación

Los sistemas de identificación de placas vehiculares han evolucionado gracias a la aceptación de la comunidad científica de diferentes países (siendo China e India los mayores promotores) y a la gran cantidad de dispositivos comerciales disponibles donde el valor monetario define no solo la calidad del *hardware* sino también el porcentaje final de reconocimiento para ciertos escenarios ocasionando que exista escasa información referente al *software* empleado y que las bases de datos sean privadas lo que permite crear alternativas para resolver los diferentes problemas a los que se enfrentan donde el principal reto es la manipulación de imágenes en entornos versátiles.

En la presente tesis el sistema desarrollado pretende servir como apoyo para controlar los accesos de residentes y visitantes autorizados para aumentar el nivel de seguridad de cualquier estacionamiento ya sea público o privado, considerando para ello únicamente características de las placas del estado de Puebla.

1.2. Objetivos

Para el trabajo de tesis se fijaron los siguientes objetivos:

Objetivo General

Desarrollar un sistema para la detección de accesos no autorizados en un estacionamiento haciendo uso de inteligencia artificial.

Objetivos Específicos

1. Generar un banco de imágenes en un estacionamiento con atributos que permitan detectar un vehículo (placas del estado de Puebla).
2. Diseñar un algoritmo semi automático para la segmentación de la placa del vehículo.
3. Diseñar un algoritmo para separar los caracteres (letras y números).
4. Desarrollar una red neuronal para la identificación automática de la placa.
5. Generar una interfaz gráfica que incluya los elementos anteriores para supervisar el control de acceso a un estacionamiento.

1.3. Diagrama de la metodología empleada

El diagrama de bloques que se implementó para el control de accesos se observa en la Figura 1.1.

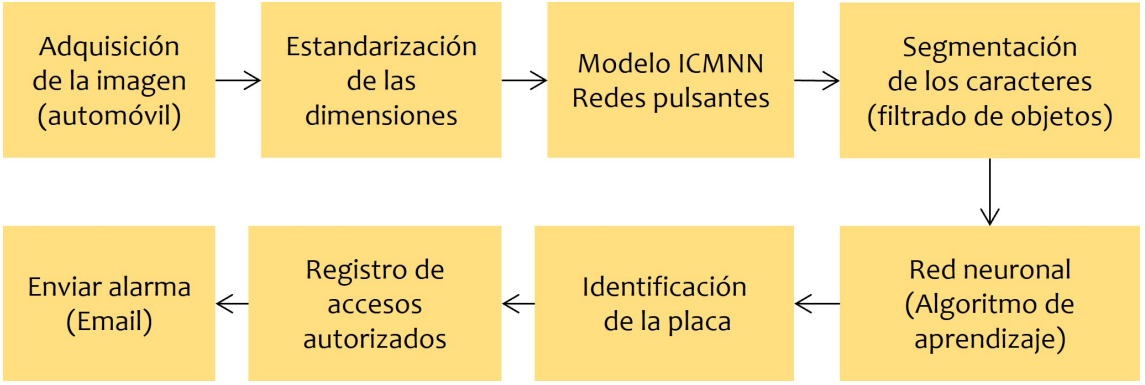


Figura 1.1: Metodología propuesta

En la Figura 1.2 se muestra el diagrama de flujo del sistema de detección de accesos no autorizados.

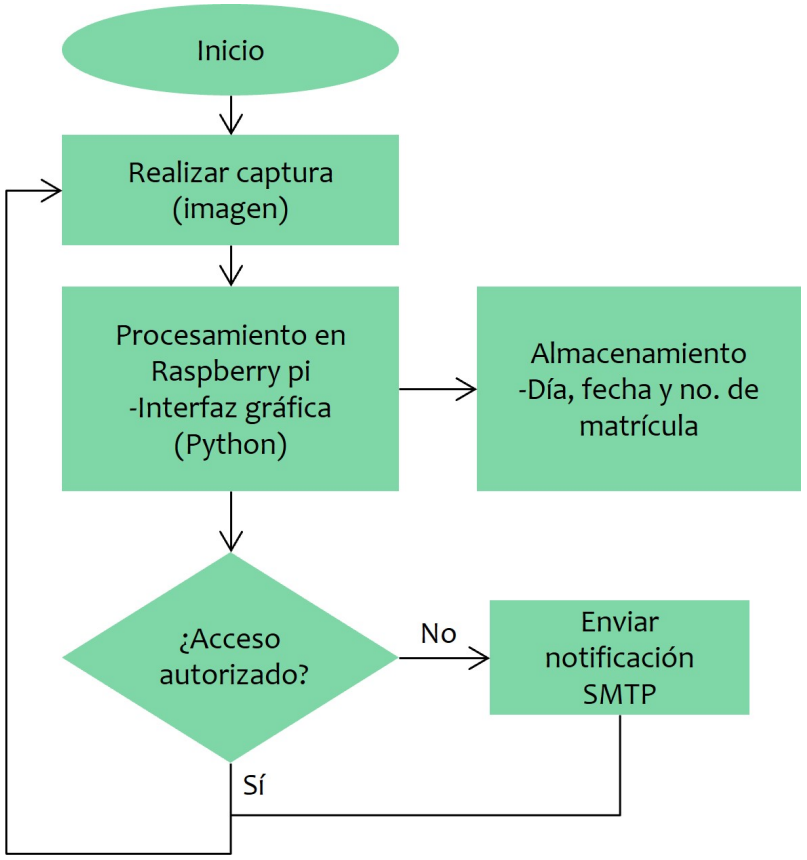


Figura 1.2: Diagrama del sistema implementado en Raspberry Pi.

1.4. Estructura del documento

La tesis está organizada de la siguiente manera. El **capítulo 2** contiene una revisión de trabajos similares de otros autores, en el **capítulo 3** se realiza una introducción a las técnicas utilizadas durante el desarrollo del sistema donde se desglosan definiciones y términos relacionados al procesamiento digital de imágenes y redes neuronales artificiales, en el **capítulo 4** se describen los procesos involucrados en el sistema propuesto y posteriormente en el **capítulo 5** se muestran los resultados obtenidos. En el **capítulo 6** se presentan las conclusiones y trabajo futuro. Por último se incluye la bibliografía utilizada como parte del trabajo de investigación.

Capítulo 2

Trabajo relacionado

En este capítulo se revisan algunos trabajos de diferentes autores para poner en contexto el trabajo realizado.

2.1. Localización de la placa

En este proceso de segmentación se busca aislar la región de la placa de todo el automóvil. A continuación se describen brevemente algunos métodos de detección de matrículas existentes en función de las características que se utilizaron.

2.1.1. Detección de bordes

Los métodos de detección de bordes se utilizan comúnmente para encontrar todos los rectángulos posibles en la imagen.

En [6], se obtuvieron las regiones candidatas restando la erosión de la imagen en escala de grises al resultado de aplicar el operador Sobel a la imagen original, mientras en que en [7] se ofrecen dos elementos estructurales para la extracción de los bordes horizontales y verticales utilizando este mismo operador. Algunos trabajos se han centrado en imágenes con ruido como en [8], donde la información de los bordes de la imagen se extrae evaluando la imagen por zonas de tal forma que se almacene la posición de los píxeles que presenten cambios bruscos de intensidad, logrando evitar la interferencia de ruido. Así mismo, en [9] se utiliza el algoritmo de Canny para obtener los bordes de las placas en imágenes borrosas, permitiendo tener una única respuesta para cada borde.

De igual forma, se han hecho esfuerzos por estandarizar la perspectiva de cada imagen fija y la detección de matrículas en video. Por ejemplo, en [10] se utilizan los bordes para corregir la inclinación teniendo como apoyo la transformada de Hough

y en [11] la clasificación se realiza con base en el valor de la proyección de los bordes de cada fotograma.

2.1.2. Basados en color

Algunos autores se inclinan hacia este tipo de método porque cada país tiene colores específicos de diseño que incluso varían en estilos (de doble o triple línea) dependiendo del estado, siendo una característica que las vuelve únicas.

En [12], se propone ajustar el rango de valores de cada componente del espacio de color HSV para la detección de placas azules y amarillas. En [13] se utilizan las funciones de membresía para dividir la imagen, creando vectores de características que también incluyen información de la textura. En [14], se utiliza un enfoque de búsqueda de franjas periódicas para encontrar el tono de cada píxel para agruparlos y después verificar que su forma y relación de aspecto coincidan con las de la placa.

En [15], se sugiere utilizar el espacio de color HSI en lugar de la imagen en escala de grises para evitar los falsos positivos en placas azules y amarillas. En [16], se binariza la imagen original para utilizar los bordes y el análisis de componentes conectados y en [17], se utiliza el valor promedio del canal H del espacio de color HSV para identificar los placas amarillas, rojas, verdes y azules en automóviles de procedencia china.

2.1.3. Basados en textura

Este tipo de método depende de la presencia de caracteres en la matrícula, lo que da como resultado un cambio significativo en el contraste entre el color de los caracteres y el color de fondo de la matrícula.

En [18], el algoritmo AdaBoost permitió localizar la placa en secuencias de video en cualquier parte de la imagen con diferente escala e inclinación utilizando características de Haar. Sin embargo se especifica que presenta problemas con sesgos de más de 45 grados. En [19], se utilizó el filtro Gabor para resaltar regiones con información de alta textura en imágenes con alta resolución cambiando sus parámetros a través de las pruebas, resultando ser flexible a otro tipo de aplicaciones. En [20], la contribución que se presenta es en relación a la combinación de la técnica de ventanas concéntricas deslizantes (SCW) y diferentes operadores morfológicos para localizar más de una placa, teniendo un tiempo de respuesta de 0.1 segundos.

En [21], se aplica la función sub-banda HL de la transformada Wavelet (DWT) para resaltar significativamente los bordes verticales de la matrícula y suprimir el ruido. Pese a la alta precisión obtenida, solo se puede utilizar en imágenes de alta resolución o con capturas a corta distancia.

2.1.4. Características globales de la imagen

En este apartado se encuentran métodos que también utilizan técnicas de procesamiento de imágenes pero que no pueden categorizarse en los grupos anteriores.

En [22], se utilizó un modelo matemático basado en una cadena de Markov bidimensional de valores discretos con dos estados para reducir el costo computacional centrándose en los planos de bits de los dígitos. En [23] se optó por definir tres umbrales. El primero para eliminar los píxeles que no pertenecían a las esquinas de la placa, el segundo para encontrar los candidatos y el tercero para ampliar las regiones y analizar los clusters resultantes. En [24], se plantea un algoritmo secuencial sencillo de implementar pero vulnerable a muchos aspectos además del ruido ya que se deben definir las estructuras de los operadores y el umbral óptimo, valores esenciales que definen el desempeño final. En [25], se calcula la dimensión y el intervalo fractal de la región de la matrícula para después establecer en cero los valores en escala de grises del fondo, reportando una tasa de pérdida de cero en placas chinas.

En [26], se implementó el algoritmo de dilatación isotrópica utilizando la transformada de la distancia euclidiana de la imagen binaria, siendo una alternativa viable para detectar vehículos en movimiento.

2.1.5. Basados en aprendizaje profundo

El aprendizaje profundo es un tipo de aprendizaje automático en el que los diseñadores forman grandes redes neuronales artificiales similares a las que se encuentran en el cerebro humano. Estos modelos computacionales se especializan en la detección de determinadas características de los datos que se procesan en cada nivel de la red.

En [27], el sistema de reconocimiento consta de tres módulos: de rectificación, residual y de secuencia, que son responsables de la rectificación, extracción de características y reconocimiento de caracteres respectivamente. Su desempeño es elevado en escenas complejas como lluvia, nieve, niebla, obstrucciones, matrículas distorsionadas y escenas borrosas pero se limita a 125fps. En [28], se diseñó una red neuronal convolucional basada en regiones (RCNN) capaz de identificar y clasificar todos los objetos detectados. En términos de precisión este tipo de configuración supera notablemente a las técnicas de procesamiento de imágenes pero se necesita un número de ejemplos considerable para el entrenamiento, al igual que en [29], donde la red convolucional necesitó 182 muestras por dígito (0-9) y 28 por letra (A-Z) para alcanzar un 98.2 % de reconocimiento.

Como se pudo observar, cada uno de los métodos propuestos ha sido probado y evaluado en una base de datos específica por lo que podrían adaptarse o no a otro

Tabla 2.1: Comparación de diferentes métodos de localización de placas

Técnica	Ventajas	Desventajas	Ref.
Detección de bordes	Algoritmos simples que pueden utilizarse en aplicaciones en tiempo real	Sensibles al ruido, no se recomienda en imágenes demasiado borrosas o en escenas que involucren muchos objetos	[6-11]
Basados en color	Pueden detectar colores específicos	Se limita a ciertas condiciones de iluminación. Algunos espacios de color se ven afectados por el ruido	[12-17]
Basados en textura	Método robusto ante placas deformadas	Se deben considerar imágenes en alta resolución o capturas a corta distancia. El costo computacional aumenta por el exceso de bordes	[18-21]
Características globales de la imagen	La localización es independiente de la posición	Puede presentar errores al detectar objetos	[22-26]
Basados en aprendizaje profundo	Eficientes en entornos no controlados	Alto consumo de recursos, algoritmos complejos	[27-29]

tipo de aplicaciones. En la Tabla 2.1 se muestran las ventajas y desventajas de cada enfoque según lo reportado.

2.2. Reconocimiento de la matrícula

Después de analizar cada región obtenida en la etapa de localización y descartar los no candidatos, en el proceso siguiente cada segmento obtenido es representado como un carácter alfanumérico. La Tabla 2.2 resume las técnicas empleadas por diversos autores para sistemas ALPR, en ella se incluye el número de referencia, país, año, los algoritmos utilizados en cada etapa de reconocimiento, el número de muestras, la resolución de las imágenes, el porcentaje de reconocimiento final (precisión), tiempo de procesamiento y el hardware o software empleado. A continuación se describe lo reportado por cada artículo.

En 2006, Christos Nikolaos E. *et al.* [30] presentaron un sistema ALPR que se centra en utilizar el método de ventanas concéntricas (SCWs). En él básicamente se definen n ventanas que recorrerán toda la imagen proporcionando mediciones estadísticas y si se supera cierto umbral se considera que el píxel central de las ventanas pertenece a una placa candidata. El bajo rendimiento reportado es dependiente de las condiciones de adquisición del conjunto de imágenes. En [31], K.Chang, S.Ryoo

y H.Lim proponen un sistema de seguimiento vehicular. Después de identificar la matrícula del automóvil en la secuencia de video, se calcula su centroide y la distancia para centrar el vehículo en la escena. En el aparto de resultados se especifica que las imágenes que utilizaron no presentan cambios bruscos de iluminación. En el artículo de V.Laxmi y H. Rohil [32] se muestra el proceso de extracción de características usando el filtro Wavelet. La mayor ventaja que presenta es que es invariante a la rotación aunque el desempeño se ve afectado por la distancia de captura entre la cámara y el automóvil. También se añade el uso de una red neuronal del tipo *back propagation* como alternativa a template matching para la detección de la placa.

En [33], I. Abbas y A.Hashim utilizan sensores IR para obtener las imágenes de las placas y almacenarlas en un ordenador (PC). Posteriormente utilizan la técnica de coincidencia de plantillas para la identificación de caracteres. Sin embargo el método no es el apropiado por el costo computacional implícito cuando la cantidad de datos aumenta. En 2017, K.Bachchan, A.Gorai y P.Gupta [34] presentaron una propuesta de reconocimiento de matrículas de la India invariante a la rotación y a las condiciones de iluminación. El patrón binario local permite identificar la textura de los caracteres alfanuméricos y después reconocer la ubicación de la placa por su histograma comparándola con plantillas. En el documento reportado por S.Montazzolli y C.Jung [35] se utiliza la red *You Only Look Once* (YOLO) en diferentes versiones para el reconocimiento de la matrícula. Se especifica que las imágenes deben tener una alta resolución y el automóvil no debe ocupar toda la escena (en promedio debe abarcar el 0.26 % de la imagen) debido a que los objetos tienen áreas definidas. Durante el proceso de identificación también se acentuó la confusión entre números y letras que son similares (O/Q, 0/D, 1/I, etc...). En [36], P.Dhar *et al.* crearon un conjunto de imágenes con placas pertenecientes a Bangladesh con 14 clases de caracteres. Las etapas de localización y segmentación surgen de la unión de varias técnicas como la utilización de filtros morfológicos para eliminar áreas relacionadas a detalles pequeños en la imagen y el filtrado por área y relación de aspecto. Los resultados experimentales señalan que es una buena propuesta para ambientes controlados.

En 2020, Z.Selmi *et al.* [37] proponen un sistema ALPR basado en redes neuronales convolucionales. La red consiste en 21 capas pooling donde las características resultantes representan 1/8 de la imagen original e introducen como función de activación principal "swish"propuesta por Google Brain. El sistema fue comparado con la metodología de otros autores y cuatro conjuntos de imágenes pernecientes a Taiwan, China, Estados Unidos y Túnez. En el trabajo realizado por W.Wang *et al.* [38] también se propone una red neuronal convolucional pero multitarea. En la etapa de localización se utiliza MTLPR como alternativa a las configuraciones de

las redes You Only Look at Once (YOLO), Faster R-CNN y Single Shot Detector (SSD). Además, el proceso de segmentación se omite sustituyéndolo por estructuras similares a las que utilizan las redes neuronales convolucionales recurrentes. En el apartado de resultados la red muestra ser robusta en imágenes con escenarios complejos. El artículo presentado por C.Henry, Y.Ahn y W.Lee [39] propone un sistema capaz de reconocer matrículas de diferentes países extrayendo una cadena ordenada de los caracteres. La estrategia principal que siguieron fue analizar los diseños de las placas, incluyendo el de países como Austria, Japón, Canadá, Perú, Sudáfrica y Nueva Zelanda. El sistema fue evaluado en placas de Korea del Sur, Taiwan, Estados Unidos, Grecia y Croacia. En [40], V. Pustokhina *et al.* proponen el uso del algoritmo de Bernsen y el análisis de componentes conectados para la detección de la placa, k-means y Krill Herd para la segmentación de caracteres. X.He y P.Hao [41] plantean un sistema de reconocimiento de matrículas chinas en tiempo real que además funciona correctamente con placas distorsionadas por la posición de la cámara.

En 2021, Q.Huang, Z.Cai y T.Lan [42] presentan ALPRNet, una única red neuronal convolucional que realiza el proceso de detección y reconocimiento simultáneamente considerando cada dígito como un objeto. El sistema se implementó en PyTorch1.0 y ResNet-50. y fue evaluado en 3 tipos de placas (Mainland China, Hong Kong, y Macao) y 72 clases de caracteres (26 ingleses, 10 numéricos y 36 Chinos).

Tabla 2.2: Metodología empleada por diversos autores

Ref.	**País y año**	**Etapas de reconocimiento** L: Localización S: Segmentación I: Identificación	**Conjunto de datos** N: No. imágenes R: Resolución	**Rendimiento y recursos** P: Precisión (%) T: Tiempo
[30]	Grecia (2006)	L: SCWs, segmentación local de Sauvola, etiquetado de regiones S: orientación, relación de aspecto, número de filas, desviación estándar I: red neuronal probabilística de dos capas	N: 1334 R: 1024x768	P: 89.1 T: 276ms SONY XCD/X700
[31]	Corea del sur (2011)	L: bordes verticales, umbralización y longitud S: umbralización por región de acuerdo con el color de la placa I: coincidencia de plantillas, NF	N: 250 R: 1300x1030	P: 92.4

[32]	India (2014)	L & S: filtro Wavelet, Sobel, regiones delimitadoras I: reconocimiento óptico de caracteres	N: 30 R: 1200x1600	Matlab R2007a
[33]	Bagdad (2016)	L: dilatación, proyecciones, Sobel, relleno de huecos, apertura morfológica S: etiquetado de componentes conectados, método de Otsu, transformada de Hough I: correlación de coincidencia de plantillas	N: 60 R:1000x1000	P: 97.74 T: 6s MATLAB R2013a, cámara A4tech USB, sensor IR GP2Y0A21YK0F
[34]	India (2017)	L: patrón binario local, coincidencia del histograma S: cuadros delimitadores I: características del histograma, KNN	N: 300 R: 450x140	P: 92.74 cámara Moto G 3G (13 megapixeles)
[35]	Brasil (2017)	L: YOLO - segmentación vista frontal S & I: FAST-YOLO	N: 2000 R: 1920x1080	P: 93 T: 13s Titan X Pascal GPU
[36]	Bangla-desh (2018)	L: operador Prewitt, dilatación, verificación de forma utilizando DtBs S: corrección de inclinación, binarización, dilatación, filtrado por área I: red neuronal convolucional (CNN)	N: 1400	P: 99.6 T: 140s Matlab 2017a
[37]	Túnez (2019)	L & S & I: DELP-DAR, CNN	N: 412,502 R: 1028x728	P: 99.3 opencv 3.1, caffe software, NVIDIA GeForce GPU
[38]	China (2019)	L: MTLPR- RNA en casada, supresión no máxima S & I: redes neuronales recurrentes, clasificación temporal conexionista	N: 500,000	P: 98 Tesla K80 GPU, Intel i7-6700
[39]	Corea del sur (2020)	L: YOLOv3 S & I: YOLOv3-SSP, detección de diseño	N: 9463 R: 1920x1080	P: 99 T: 41.5 ms NVIDIA GTX Titan X Pascal

[40]	Rusia (2020)	L: IBA, CCA S: OKM, Krill Herd I: CNN	N: 59800	P: 98 TensorFlow, Pillow, OpenCV, PyTesseract
[41]	China (2020)	L: CNN S & I: VGG, RCNN, ResNet	N: 250,000 R: 1160x720	P: 98.3 Pytorch, Intel Xeon E5-2680v4 processor, Nvidia Tesla K80*8 GPU
[42]	China (2021)	L & S & I: ALPRNet	N: 1376 R: 1024x1024	P: 99.43 Nvidia RTX2070 GPU, Intel Corel i7-6700 CPU

En la Tabla 2.3 se muestran las ventajas y limitaciones de cada técnica de acuerdo a lo revisado en este apartado omitiendo las redes convolucionales por tratarse de un tipo de método basado en aprendizaje profundo, caso que ya ha sido abordado en la localización de la placa.

Tabla 2.3: Comparación de diferentes clasificadores

Técnica	Ventajas	Desventajas	Ref.
Coincidencia de plantillas	El proceso es ágil si se limita la cantidad de plantillas o se definen puntos clave	Presenta errores en ligeras desviaciones en la forma, el tamaño y la orientación de la imagen	[31] [33] [43–47]
Clustering	Eficiente en grandes conjuntos de datos	El tiempo de procesamiento depende del número de clusters	[48–50]
K-NN	Algoritmo simple de implementar	La precisión puede verse afectada por el ruido	[34] [51] [52]
Redes neuronales probabilísticas	Son relativamente insensibles a valores con ruido	Alto consumo de recursos	[30] [53]

En síntesis, los sistemas de reconocimiento de matrículas se dividen en tres procesos principales: localización, segmentación y reconocimiento aunque en algunas ocasiones se suele añadir una etapa de preprocesamiento para ajustar el contraste o suavizar las imágenes. En la Figura 2.1 se muestra un mapa conceptual de los

métodos y técnicas convencionales involucradas en este tipo de sistemas según el estado del arte.

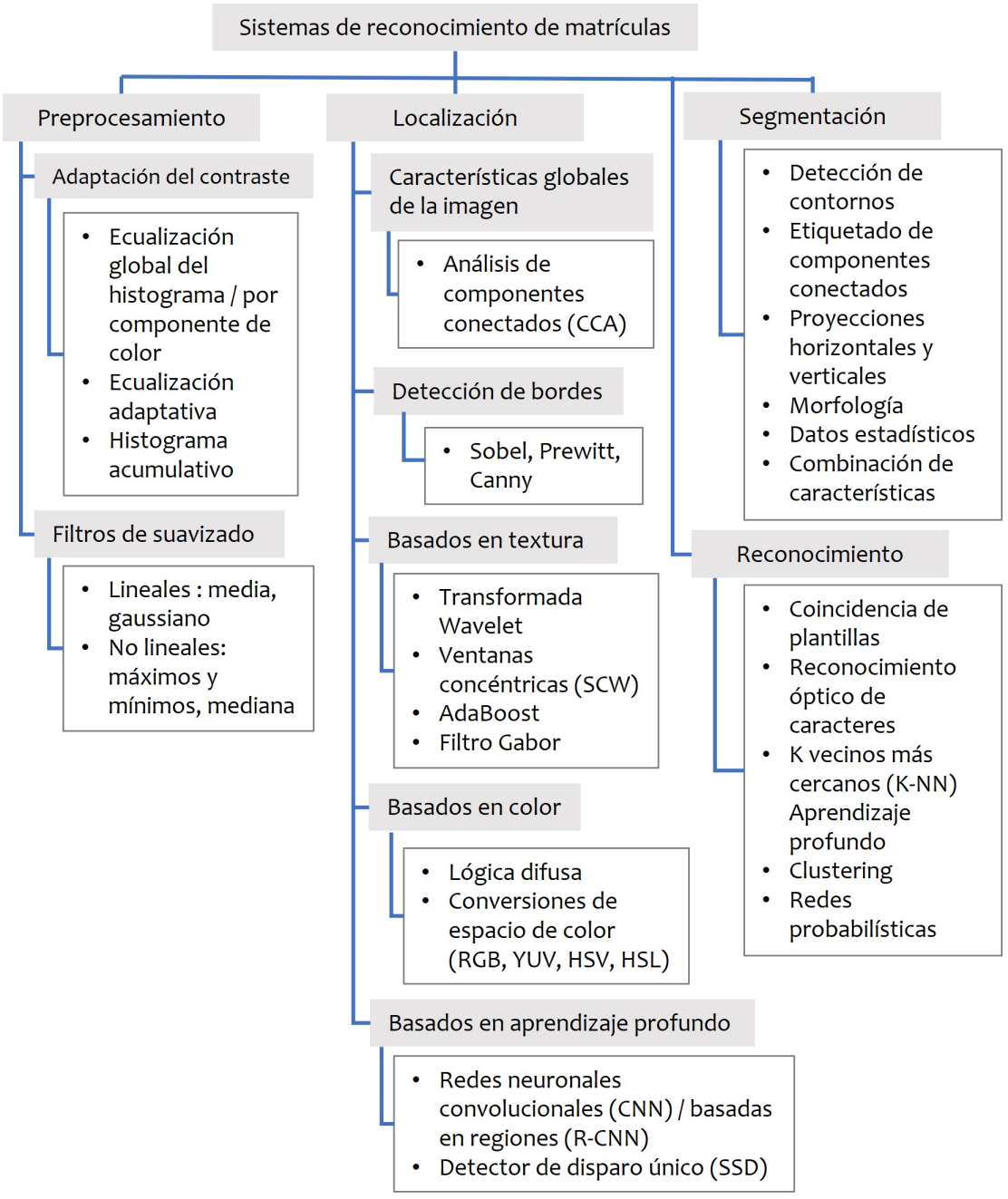


Figura 2.1: Taxonomía de los ALPR

En el siguiente capítulo se describirán los fundamentos de las técnicas aplicadas al sistema propuesto de reconocimiento de matrículas y otro tipo de alternativas que también fueron consideradas durante su implementación, pero que se descartaron por mostrar un bajo desempeño en el conjunto de datos de prueba.

Capítulo 3

Marco Teórico

El análisis de imágenes digitales comprende una amplia variedad de técnicas y algoritmos debido a las múltiples implementaciones que se han realizado en función del software y hardware disponible, sumado al creciente uso de inteligencia artificial cuyo fin es diseñar una máquina que desempeñe las mismas funciones cognitivas que un ser humano. En este capítulo se realiza una introducción al procesamiento digital de imágenes y diversas configuraciones de redes neuronales, así como una serie de conceptos que serán de utilidad para los siguientes capítulos.

3.1. Procesamiento básico de imágenes

Mediante procesos complejos, el cerebro humano es capaz de recoger información visual y procesarla para distinguir los elementos que forman una escena (personas, animales y objetos). En contraste, las computadoras realizan este proceso de una forma completamente diferente debido a que solo las interpretan como una matriz de números.

Una imagen digital puede definirse como una función bidimensional I(x,y) que cuantifica la intensidad de luz [54] o nivel de gris obtenido por el indexado de las coordenadas x e y. La unidad mínima que las conforma es el píxel y su manipulación puede enfocarse desde dos perspectivas:

- Alteración píxel a píxel
- Operaciones basadas en múltiples puntos - vecindad.

En las operaciones píxel a píxel sólo es tomado en cuenta el valor del píxel involucrado y se obtienen a partir de la ecuación I'(x,y)=f[I(x,y)]. La función f podrá ser o no independiente de las coordenadas de la imagen (homogéneas/no homogéneas) y lineal

o no lineal dependiendo del operador elegido. Ejemplos de este tipo de operadores son cambios de contraste y de iluminación, inversión o complemento, segmentación por umbral y transformación de color.

Contrario a lo anterior, en las alteraciones por vecindad se involucra la relación que tiene un píxel de manera posicional con los píxeles mas cercanos a él por lo que el valor de salida es la suma promediada de sus vecinos. En la Figura 3.1 se muestra la vecindad 8 definida para un píxel p en la posición (x,y). Los filtros entran en esta categoría [55] y son el resultado de la combinación de una función y una máscara, *kernel*, que se "mueve" a través de la imagen. Esta máscara H(i,j) tiene su propio sistema de coordenadas y sus valores definen los pesos con los cuales los correspondientes valores de intensidad de la imagen son multiplicados. De manera formal, los valores de los píxeles en la nueva imagen I'(x,y) son calculados utilizando la ecuación 3.1.

$$I'(x,y) \leftarrow \sum_{(i,j)\in R(x,y)} I(x+i, y+j) \cdot H(i,j), \tag{3.1}$$

Generalmente se utilizan máscaras de 3x3 pero las dimensiones dependerán del resultado que se busca debido a la influencia de cada vecino.

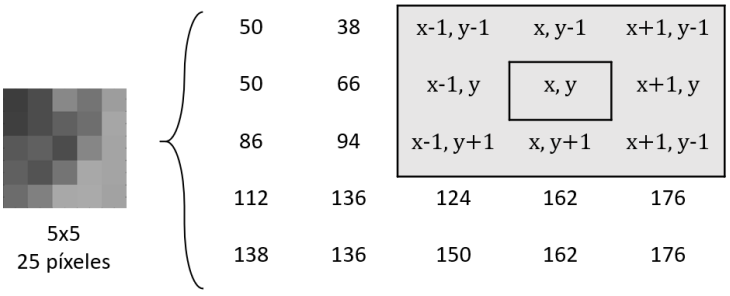


Figura 3.1: Definición de la vecindad de un píxel, 8-vecinos

Entre los filtros más conocidos se encuentran:

- Difuminado/Suavizado. Son filtros pasa-bajas que eliminan ruido y detalles en la imagen que no tienen relevancia, ejemplos de ello son:
  - Media. El valor del píxel se obtiene con la media aritmética de los píxeles de la máscara.
  - Mediana. La salida se obtiene ordenando de menor a mayor intensidad los píxeles de la máscara de referencia para después tomar el valor que está en medio. Suele utilizarse para eliminar el tipo de ruido sal y pimienta por ser menos sensible a valores extremos.

  - Gaussiano. Los vecinos más cercanos al píxel actual tienen más peso que los externos. Estos pesos se calculan con una campana de Gauss dependiente de la distancia al píxel.

- Operaciones morfológicas. Se suelen aplicar sobre imágenes binarias para simplificar formas o describir objetos. En este tipo de filtro se encuentran los operadores de erosión, dilatación, apertura, cierre y hit-or-miss.

- Detección de bordes. Algunos de los métodos más utilizados son Prewitt, Sobel, Roberts, Kirsch, Frei-Chen, Laplaciano y Canny.

3.1.1. Imágenes en escala de grises y a color

La asignación de color a cada píxel se realiza por medio de bits [56]. Si a una imagen se le asigna un bit por píxel esta tendrá sólo dos colores (1, 0), por este motivo generalmente se utiliza un rango de 8 bits para ampliar los niveles de intensidad que se pueden apreciar en una imagen en escala de grises.

Entre los espacios de color más utilizados están el rojo, verde y azul - RGB, donde cada píxel está representado por tres números correspondientes a cada componente. En este caso el conjunto de valores (0,0,0) es el negro absoluto, el (255,255,255) el blanco absoluto, el (255,0,0) el rojo puro, el (0,255,0) el verde puro y el (0,0,255) el azul puro. La combinación de distintos valores generará otros colores como el (255,255,51) que es un tono amarillo, siendo 255^3 el número de combinaciones posibles. Además del RGB existen otros modos de color como el HSV, HSI, Lab, CMY, YCbCr y HLS. Algunos de ellos difieren de la idea de hacer la visualización de colores más fiel a la realidad, sino que son abstracciones matemáticas que hacen posible el tratamiento de ciertas propiedades de la imagen como lo es el modelo Lab que representa todos los valores del espectro visible.

3.1.2. Histogramas

En el área de procesamiento digital de imágenes, los histogramas describen la frecuencia con la que se presentan los valores de intensidad de los píxeles permitiendo interpretar las complicaciones [54] que se puedan presentar para su manipulación. El histograma de una imagen I(u,v) en escala de grises con intensidades en el intervalo [0, *K-1*] se define como:

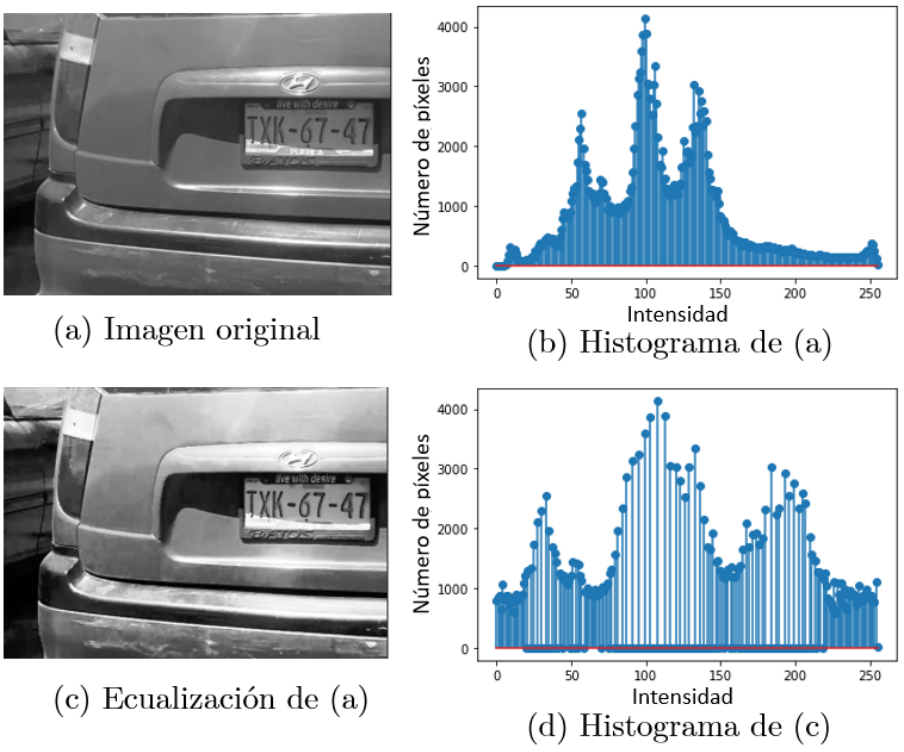
$$n_k = h(r_k) \tag{3.2}$$

donde r_k es la k-ésima intensidad de gris, n_k es el número de píxeles en la imagen con r_k nivel de gris y $h(r_k)$ es el histograma de la imagen.

Es importante mencionar que es imposible reconstruir una imagen utilizando solamente el histograma debido a que este no proporciona información acerca del

origen de los píxeles que los conforman pero en contraste proporciona características como:

- Iluminación. Los niveles de gris se agrupan en una región específica.
- Contraste. Diferencia entre el máximo y mínimo valor de intensidad de los valores presentes en la imagen.
- Dinámica. Número de píxeles diferentes que son utilizados en la imagen.



(a) Imagen original (b) Histograma de (a)

(c) Ecualización de (a) (d) Histograma de (c)

Figura 3.2: Imágenes de intensidad de 8 bits con su respectivo histograma. *K=256*

En la Figura 3.2 se muestra un tipo de transformación para aumentar el contraste de la imagen conocida como ecualización del histograma. Tiene como objetivo ampliar el rango de intensidad de los niveles de gris para tener una distribución uniforme y se puede implementar con la ecuación 3.3.

$$f_{ac} = (p - p_{low})(\frac{p_{max} - p_{min}}{p_{high} - p_{low}}) \tag{3.3}$$

donde el valor del píxel p se interpola linealmente desde el intervalo $[\ p_{low}, p_{high}\ ]$ al $[\ p_{min}, p_{max}\ ]$.

En ocasiones, el histograma también es de utilidad para el proceso de binarización. Para ello se considera un valor umbral dentro del histograma tal que a los valores menores a él se les asigna un 0 y a los mayores que él un 1.

3.1.3. Lab - ecualización adaptativa

En este espacio de color también conocido como CIELAB, L representa la brillantez y a y b las tonalidades rojo-verde y azul-amarillo. Como se observa en la Figura 3.3, la ecualización del canal L permite mejorar el contraste en imágenes oscuras en comparación con la ecualización global y la normalización del histograma.

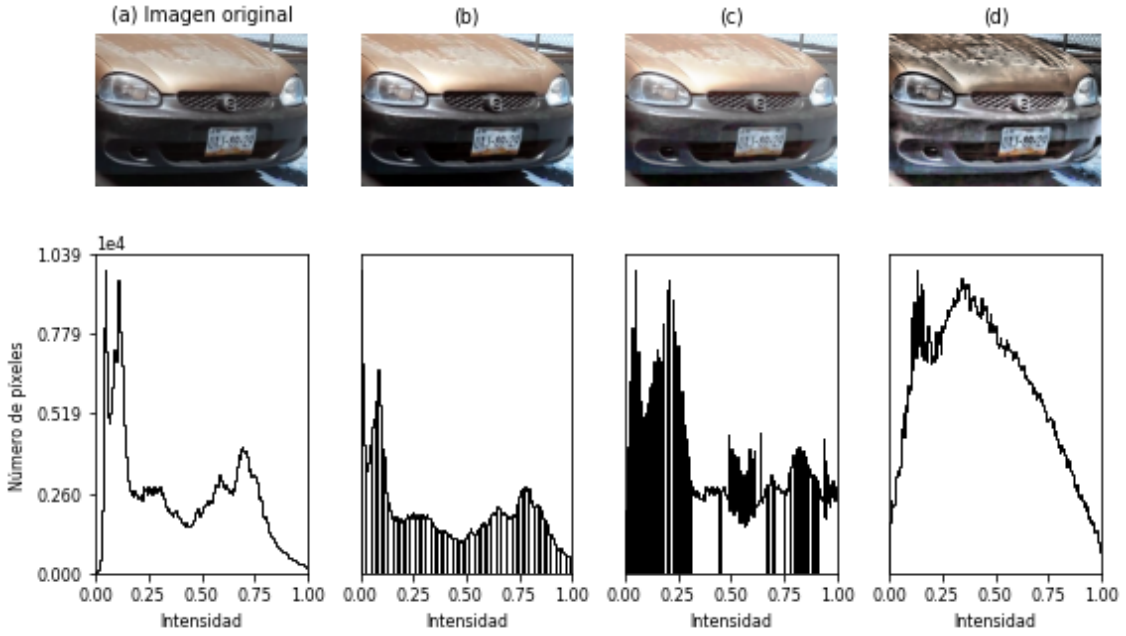


Figura 3.3: Mejora de contraste de (a) a través de la (b) normalización, (c) ecualización del histograma y (d) ecualización del canal L

La conversión de RGB a CIELAB de realiza con la siguiente matriz de transformación:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.4}$$

El modelo CIEXYZ es necesario para obtener las nuevas componentes de RGB relativas a un punto blanco definido como C=(X,Y,Z). De ello se obtiene:

$$\begin{aligned} L &= 118 f_1(Y/Y_n) - 16 \\ a &= 500[f_1(X/X_n) - f_1(Y/Y_n)] \\ b &= 200[f_1(Y/Y_n) - f_1(Z/Z_n)] \end{aligned} \tag{3.5}$$

donde

$$f_1(p) = \begin{cases} p^{(1/3)}, & si \;\; p > 0.008856 \\ 7.787 \cdot p + \frac{16}{116}, & si \;\; p \leq 0.008856 \end{cases} \tag{3.6}$$

Los valores triestímulo CIE XYZ del punto blanco de referencia se definieron como $X_n = 0.950456$, $Y_n = 1.0$ y $Z_n = 1.088754$. Los valores resultantes de L

están dentro del intervalo [0,100] y los de a y b entre [-127,127], por esta razón se recomienda escalar los píxeles al tipo uint8 con:

$$L \leftarrow L(255/100); \quad a \leftarrow a + 128; \quad b \leftarrow b + 128 \tag{3.7}$$

Para aumentar el contraste localmente, se ecualizó el canal L aplicando la siguiente fórmula con un kernel de 20x20:

$$f_{ac} = (p - p_{low})(\frac{p_{max} - p_{min}}{p_{high} - p_{low}}) \tag{3.8}$$

donde el valor del píxel p es interpolado linealmente del intervalo $[p_{low}, p_{high}]$ a $[p_{min}, p_{max}]$. Hecho esto, los canales se vuelven a transformar para continuar con su procesamiento considerando las ecuaciones:

$$\begin{aligned} X &= X_n[f_2(\frac{a}{500} + \frac{L+16}{166})] \\ Y &= Y_n f_2(\frac{L+16}{166}) \\ Z &= Z_n f_2(\frac{L+16}{166} - \frac{b}{200}) \end{aligned} \tag{3.9}$$

$$f_2(p) = \begin{cases} p^3, & si \;\; p^3 > 0.008856 \\ \frac{p-(16/116)}{7.787}, & si \;\; p^3 \leq 0.008856 \end{cases} \tag{3.10}$$

3.1.4. Etiquetado de objetos

El etiquetado de objetos se utiliza para dividir la imagen binaria en regiones conocidas como componentes conectados considerando las vecindades de los píxeles. Como ejemplo, en la Figura 3.4 se muestran dos componentes conectados en una imagen etiquetada. El algoritmo consiste inicialmente en asignar un índice temporal

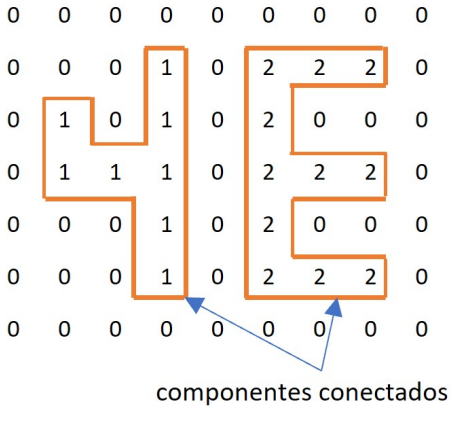


Figura 3.4: Resultado del etiquetado de los objetos

a un píxel I(x,y)=1 e incrementarlo en base a sus 4 u 8-vecinos para después resolver iterativamente colisiones que se presentan cuando un objeto tiene asignada más de una etiqueta. La implementación se realiza bajo los siguientes puntos:

1. Inicializar la matriz de componentes conectados con las dimensiones de la imagen binaria.
2. Iterar a través de cada píxel por filas y columnas
3. Asignar una etiqueta a cada píxel = 1 (objeto)
4. Si uno de los píxeles vecinos tiene una etiqueta entonces la etiqueta se incrementa
5. Si varios de los píxeles vecinos tienen una etiqueta asignada entonces se le asigna una de ellas al nuevo píxel.
6. Las etiquetas no usadas son colisiones
7. Ubicar la etiqueta más pequeña en la colisión de tal forma que sustituya las etiquetas con mayor valor.

3.2. Rotación y homografía

Las transformaciones geométricas que a continuación se describen se aplican con el fin de cambiar la orientación y la perspectiva en una imagen. Para rotarla, cada

Algoritmo 1 : Rotación

Importar imagen I
Definir ángulo θ y convertirlo a radianes
Obtener el seno de θ
Obtener el coseno de θ
Inicializar I' con las dimensiones de I
    filas h, columnas w, canales c
    h = |h·coseno| +|w·seno|
    w = |w·coseno| + |h·seno|
Obtener el centro de I y I' {cx,cy, cx2,cy2}
Para i=1:h & j=1:w
    Ubicar las coordenadas del píxel con respecto al centro de I
        x = w-1-j-cx; y = h-1-i-cy
    Ubicar las coordenadas del píxel con respecto a I'
        x2 = x·coseno + y·seno
        y2= -x·seno + y·coseno
    Ajustar el centro de I'
        x2 = cx2-x2; y2 = cy2-y2
    Mover los píxeles a su nueva posición
        I'[y2,x2,:]= I[i,j,:]

píxel f(x,y) se gira θ grados considerando un punto de origen y calculando la nueva

posición g(x',y') con base en:

$$\begin{aligned} x' &= x \cdot cos\theta - y \cdot sin\theta \\ y' &= x \cdot sin\theta + y \cdot cos\theta \end{aligned} \tag{3.11}$$

Las fórmulas utilizadas en el punto 7 del Algoritmo 1 se utilizan para girar la imagen en sentido contrario a las manecillas del reloj. En algunos casos el cambio de orientación no es suficiente y se necesita corregir la distorsión de la perspectiva. Para ello se considera el mapeo de dos sistemas de coordenadas:

$$\begin{bmatrix} h \cdot x' \\ h \cdot y' \\ h \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.12}$$

de lo anterior se obtiene:

$$\begin{aligned} x' &= \frac{a_1 \cdot x + a_2 \cdot y + a_3}{c_1 \cdot x + c_2 \cdot y + 1} \\ y' &= \frac{b_1 \cdot x + b_2 \cdot y + b_3}{c_1 \cdot x + c_2 \cdot y + 1} \end{aligned} \tag{3.13}$$

sustituyendo cuatro puntos de ambos sistemas de coordenadas (x_1,$y_1 \leftrightarrow x'_1$,y'_1) en la ecuación anterior en su forma matricial, se puede conocer el valor de los coeficientes:

$$\begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 \cdot x'1 & -y_1 \cdot x'1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 \cdot y'1 & -y_1 \cdot y'1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 \cdot x'2 & -y_2 \cdot x'2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 \cdot y'2 & -y_2 \cdot y'2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3 \cdot x'3 & -y_3 \cdot x'3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3 \cdot y'3 & -y_3 \cdot y'3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4 \cdot x'4 & -y_4 \cdot x'4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -x_4 \cdot y'4 & -y_4 \cdot y'4 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \\ b_3 \\ c_1 \\ c_2 \end{bmatrix} \tag{3.14}$$

Sustituyendo los valores obtenidos en la ecuación 3.11 se obtiene H y las nuevas posiciones quedan definidas por I'(i,j,:) = $H \cdot I$(i,j,:). I' representa la imagen resultante, I la imagen original e i y j las filas y columnas. En la Figura 3.5 se muestra un

(a) Imagen original (b) Homografía

Figura 3.5: Homografía entre I y I'

ejemplo de transformación proyectiva. Los puntos azules simbolizan las posiciones (x,y) para calcular el mapeo entre los dos sistemas de coordenadas. En la segunda imagen estos puntos se omiten porque representan las esquinas.

3.3. Aprendizaje automático

En esta era tecnológica es posible pasar desapercibido la gran cantidad de aplicaciones que han surgido gracias a las investigaciones en el campo del aprendizaje automático y que representan un papel importante en nuestra vida diaria como lo es iniciar una búsqueda en Google, detectar spam en una cuenta de correo o las recomendaciones de YouTube al término de un video. En concreto, los algoritmos detrás de estas tareas aprenden a través de la experiencia sustituyendo el trabajo de los humanos al analizar datos.

El término de aprendizaje automático, *machine learning*, fue acuñado en 1950 por Alan Turing en su artículo *"Computing Machinery and Intelligence"* [57]. Es un subcampo de la inteligencia artificial que busca crear modelos a partir de la identificación de patrones en un conjunto datos y así predecir eventos. Este aprendizaje puede dividirse en tres grupos diferentes:

1. Supervisado. Se proporciona un conjunto de datos (también llamadas muestras, ejemplos, instancias u observaciones) con las respuestas correctas (etiquetas) y en base a ellas el algoritmo se generaliza para responder correctamente a todas las entradas posibles. Clasificación y regresión son dos tipos de aprendizaje supervisado, en el primero se asignan categorías mientras que en el segundo se realizan predicciones.

2. No supervisado. Las etiquetas son omitidas y el algoritmo intenta encontrar similitudes entre las entradas para poder categorizar los ejemplos en diferentes grupos. Un subcampo del aprendizaje no supervisado es la reducción de dimensionalidad, frecuentemente utilizado para comprimir información.

3. Reforzado. El agente debe explorar un espacio desconocido para determinar las acciones a ejecutar mediante prueba y error. Aprenderá a través de recompensas o sanciones.

Dentro del aprendizaje automático existen diferentes técnicas que cubren diferentes problemáticas (Figura 3.6), sin embargo en la última década la comunidad científica se ha visto influenciada por el uso de las redes neuronales en virtud de que pueden procesar información cada vez más compleja logrando entrar al campo del aprendizaje profundo, *deep learning*, aprovechando con ello la cuarta revolución industrial relacionada con el manejo de los datos.

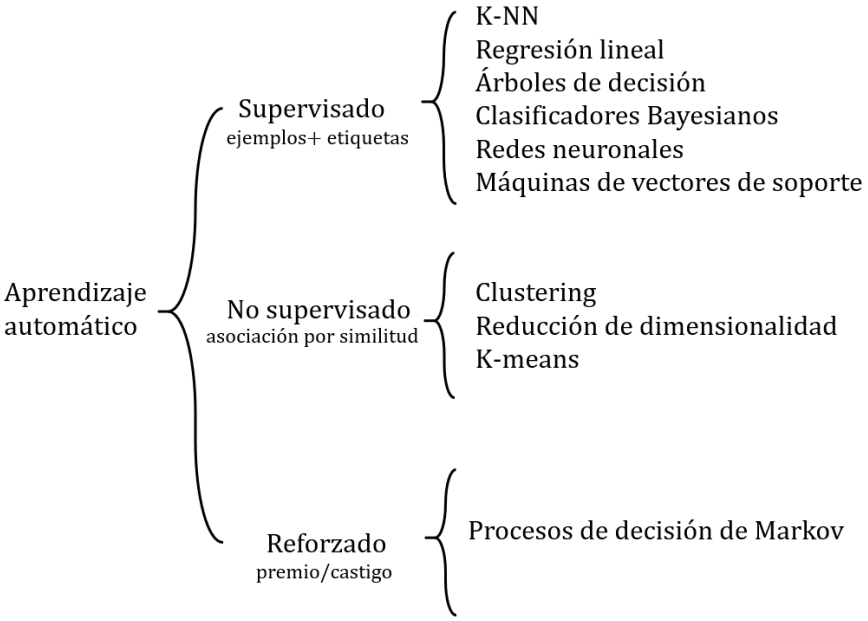


Figura 3.6: Clasificación de métodos por tipo de aprendizaje

3.3.1. K-Nearest Neighbors

Esta técnica fue propuesta por Thomas Cover y Peter Hart en 1991 - *"An introduction to kernel and nearest-neighbor nonparametric regression"* [58]. Es un algoritmo que no hace suposiciones por la distribución de los datos y la clasificación se basa en la búsqueda de características similares con cada uno de los elementos que se disponen de un conjunto de entrenamiento.

Algoritmo 2 : K-NN

Entrada: conjunto de entrenamiento y validación $\{X, Y\}$,
número de vecinos $\{k\}$
Salida: etiquetas del conjunto de validación

1: Asociar una clase a cada vector de entrenamiento $\{X_i\}$
2: D = lista vacía
3: Para cada muestra de validación $\{Y_i\}$:
Calcular la distancia a cada X_i
Generalmente se utiliza la distancia Euclidiana:
d = $\sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2}$
4: Añadir a D los valores de d y después ordenarlas ascendentemente
5: Seleccionar los k vecinos en D
6: Buscar la moda (clasificación) o media aritmética (regresión logística) de D
7: Asignar clase a Y_i según lo obtenido en el paso anterior.

Las predicciones se obtienen utilizando una función de distancia para calcular la proximidad de todos los puntos de datos de entrenamiento al nuevo dato que

se está tratando de etiquetar y después elegir la clase que más se repite. El valor de k es el número de vecinos más cercanos para realizar la comparación y llevar a cabo la votación. Para elegir su valor se recomienda utilizar un método conocido como validación cruzada donde el total de muestras se divide en un conjunto de entrenamiento y otro de validación de tal modo que se evalúa el rendimiento por bloque y al final se promedia el error (Figura 3.7).

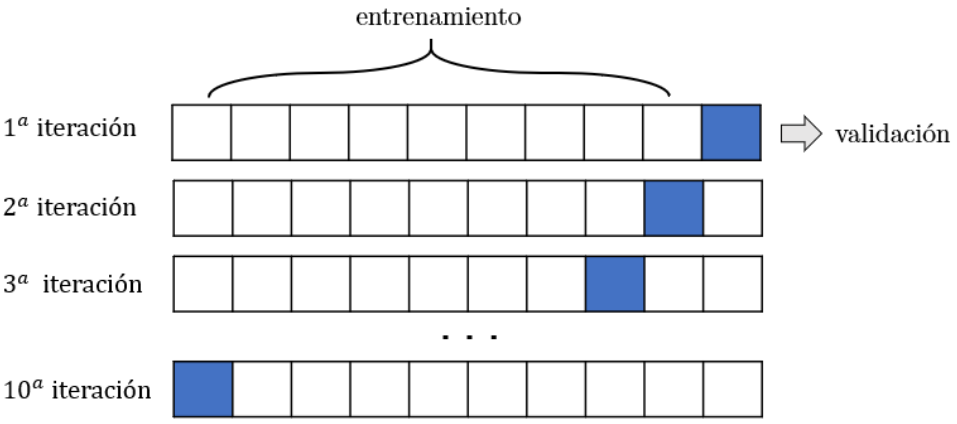


Figura 3.7: Ejemplo de validación cruzada con 10 bloques

Para la resolución de empates (no existe clase predominante en los k vecinos) existe una variante de este algoritmo llamada votación ponderada, *weighted k-nn*, donde se le asigna un peso mayor a un vecino con menor distancia a uno de mayor distancia durante el proceso de votación. Un ejemplo de función para el cálculo de los pesos es la siguiente:

$$w = exp(-\frac{distancia^2}{\sigma^2}) \tag{3.15}$$

Al elegir k, también se debe considerar que un valor bajo causará que el resultado de la predicción se ajuste mucho a los datos de entrenamiento, mientras que un valor muy elevado tendrá pocos cambios de un punto a otro. Estos errores de generalización [59] son conocidos como sobreajuste y subajuste respectivamente (Figura 3.8).

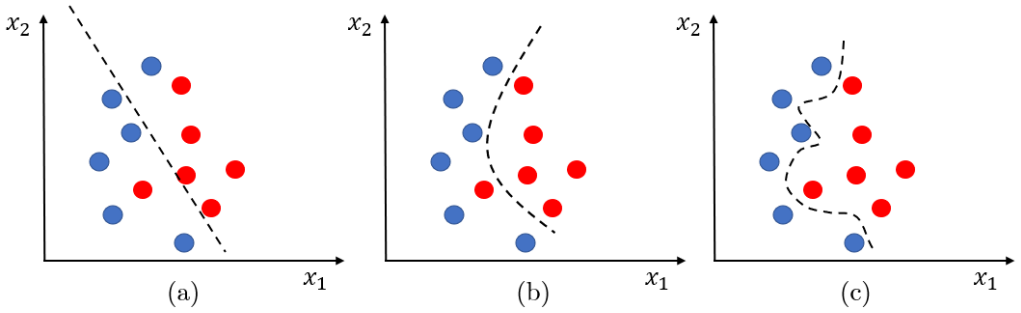


Figura 3.8: Clasificación a) con subajuste, b) apropiada y c) con sobreajuste

3.4. Redes neuronales

Las redes neuronales son en esencia sistemas de procesamiento de información que intentan emular las funciones cognitivas del ser humano, siendo por su constitución un modo alterno de procesar y modelar datos para resolver problemas que no se adecuan bien a las tecnologías convencionales. Estas redes tienen la capacidad de extraer información de masas de datos para realizar predicciones a partir de ciertas variables significativas que colaboran para producir una salida a partir de los datos de entrada. Actualmente existen muchas aplicaciones en torno a las redes neuronales referentes a clasificación de información. No obstante, también podemos incluir otras como detección de anomalías bancarias (fraudes), reconocimiento de voz, comportamientos de mercado, máquinas traductoras y como lo veremos en los siguientes capítulos, procesamiento de imágenes y reconocimiento de caracteres.

3.4.1. Perceptrón

El perceptrón también conocido como red neuronal de una sola capa (*Single-Layer Neural Network*), es un algoritmo de clasificación binaria creado por Frank Rosenblatt [60] a partir del modelo neuronal de Warren McCulloch y Walter Pitts desarrollado en 1943 cuyo propósito fue implementar un modelo computacional para simular el funcionamiento de una neurona biológica. La neurona es un tipo de célula representativa del sistema nervioso, cuya función consiste en transmitir información en forma de impulsos nerviosos a través del axón (Figura 3.9) a una dendrita de otra neurona para enviar respuestas al cerebro.

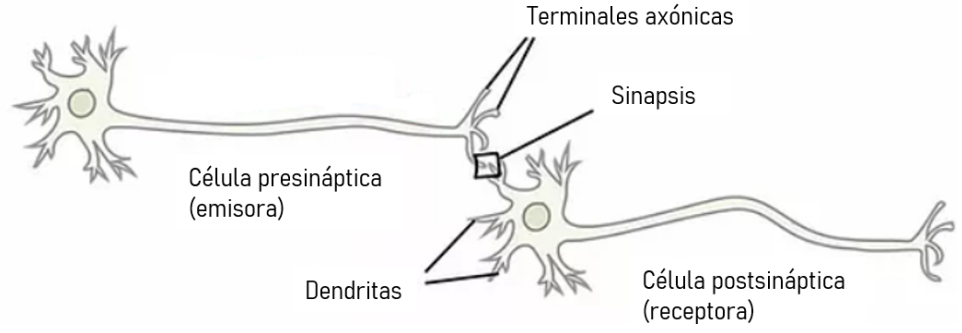


Figura 3.9: *Khan Academy*. Ilustración del intercambio de información entre neuronas, www.khanacademy.org

En la Figura 3.10 se puede observar que la única neurona de salida del perceptrón realiza la suma ponderada de las entradas y pasa el resultado a una función de transferencia (escalón unitario o bipolar) para separar la clase A de la B.

El procesamiento interno puede ser descrito por la siguiente expresión:

$$y = \sum_{i=1}^{n} w_i x_i + w_0 x_0 \;\; ; \quad y = \begin{cases} 1, si \;\; y \geq 0 \\ -1, si \;\; y < 0 \end{cases} \tag{3.16}$$

donde x_i es la entrada a la red, w_i el peso asociado con la i-ésima entrada (influencia que tiene cada variable en el sistema), y la salida de la red y x_0 una constante a menudo llamada "sesgo o *bias*"que controla qué tan predispuesta está la neurona a generar como salida un 1 o -1 independiente de los pesos.

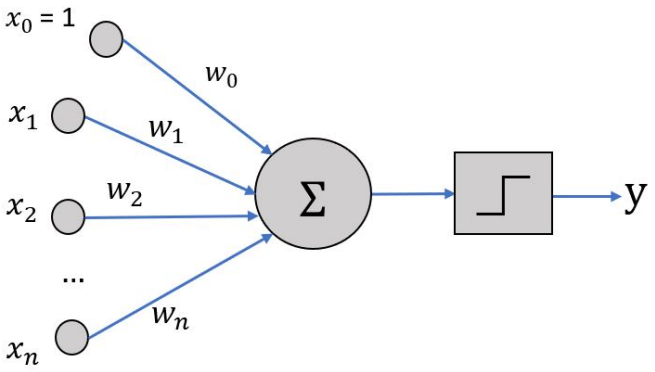


Figura 3.10: Representación de un perceptrón

Algoritmo 3 : Perceptrón

Entrada: conjunto de datos $\{x\}$
Salida: pesos ajustados para etiquetar nuevas muestras

1: Asociar todos los vectores de entrada con su salida deseada $\{x_i, y_{(ideal)}\}$
2: Asignar valores aleatorios pequeños a los pesos $\{w\}$
3: Asignar un valor entre 0.0 y 1.0 a la tasa de aprendizaje $\{\eta\}$
4: Fijar un número de épocas $\{T\}$
5: Error $\{e\} = y_{(ideal)} - y$
6: t = 0
7: Mientras e $\neq$ 0 or t < T, para cada muestra x_i:
    Resolver la sumatoria $\{a = w^T x\}$
    Obtener la salida de la red aplicando la función escalón unitario $\{y = f(a)\}$
    Modificar los pesos $\{ \; w_i = w_i + \eta(e)(x_i) \; \}$
    t = t+1

Durante la etapa de entrenamiento el perceptrón aprende al ajustar los pesos de los valores de entrada en repetidas iteraciones (también denominadas épocas) hasta lograr que la salida de la red $y_{(real)}$ coincida con la etiqueta correcta $y_{(ideal)}$. Por esta razón es fundamental que los atributos del conjunto de datos sean linealmente separables.

3.4.2. Perceptrón multicapa

El perceptrón multicapa (MLP) se basa completamente en la configuración del perceptrón y se caracteriza por tener al menos una capa intermedia. En la mayoría de los problemas del mundo real, las variables de entrada afectan la salida de la red de forma compleja, por ello es necesario agregar más capas para formar niveles de abstracción que permitan comprender mejor los datos a estudiar y así lograr que el modelo final sea más flexible para poder resolver problemas que no sean linealmente separables.

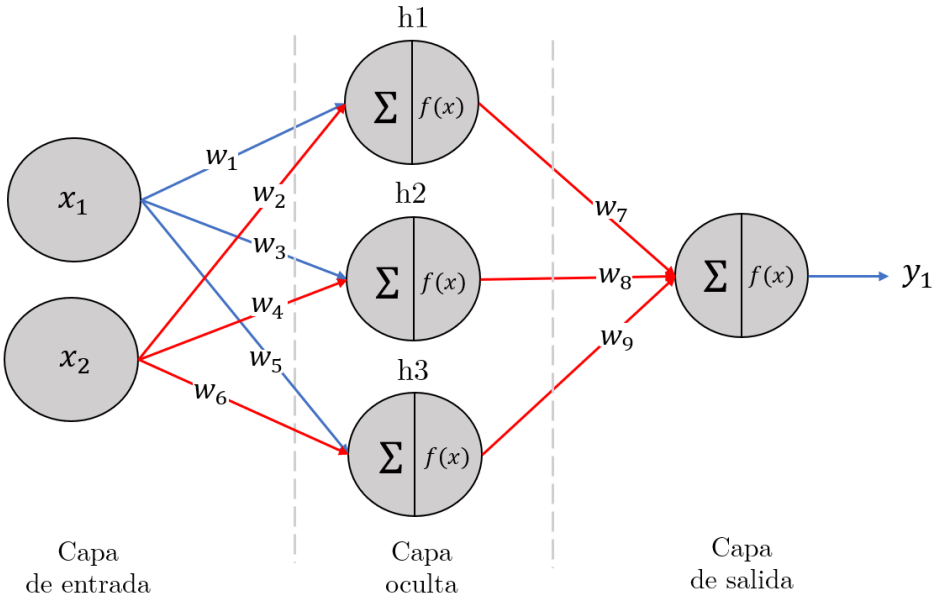


Figura 3.11: Topología de un perceptrón multicapa

En la Figura 3.11, se muestra la configuración de un perceptrón multicapa con dos neuronas de entrada x_1 y x_2, una capa oculta con tres neuronas ($h1$, $h2$ y $h3$) y una neurona en la capa de salida. Para obtener y_1, cada neurona recibe como entrada los valores de todas las neuronas de la capa anterior siguiendo una sola dirección (proceso conocido como *feedforward* o "propagación hacia adelante"), pasando por una función de activación h que suele tener un rango de valores continuo. Las más utilizadas son las siguientes:

- Función sigmoidal:

$$h = \frac{1}{1 + e^{-x}} \tag{3.17}$$

- Función tangente hiperbólica:

$$h = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{3.18}$$

A diferencia del perceptrón, el entrenamiento se puede llevar a cabo mediante la retropropagación del error (*backpropagation*) entre capas a través del descenso del gradiente [61] para que los pesos se actualicen en función del aporte en que han contribuido a ese error y como función de coste global se utiliza el error cuadrático medio, que representa la suma de los errores resultantes de la diferencia entre la salida ideal y la salida de la red (ecuación 3.19).

$$ECM = \sum \frac{1}{2} * (y_(ideal) - y_(real))^2 \tag{3.19}$$

El gradiente es el conjunto de todas las derivadas parciales de una función y contiene la información de cuánto crece en un punto en específico por cada una de sus dimensiones, siendo la pendiente del punto en que nos encontremos de la función de coste. Habrá uno respecto a los pesos de la capa de salida y otro respecto a los pesos de la capa oculta. El ajuste de los pesos tiene que ser en dirección opuesta del gradiente para minimizar el error implementando la siguiente ecuación:

$$w_{n+1} = w_n - \eta \partial \frac{e}{w} \tag{3.20}$$

El valor de la tasa de aprendizaje (η) influye en la velocidad con la que converge el algoritmo. Se recomienda asignarle un valor pequeño [0.05, 0.25].

Algoritmo 4 : Perceptrón multicapa

Entrada: conjunto de entrenamiento y validación $\{x, v\}$
Salida: etiquetas del conjunto de validación

1: Asociar todos los ejemplos de entrenamiento con su salida deseada $\{x_i, y_i\}$
2: Asignar valores aleatorios pequeños a los pesos $\{w\}$
3: Definir una tasa de aprendizaje $\{\eta\}$
4: Para cada x_i obtener la salida de la red (*feedforward*)
    Resolver la sumatoria $\{a = w^T x\}$
    Aplicar la función de activación $\{y = h(a)\}$
5: Evaluar el error (3.19)
6: Calcular el incremento parcial de los pesos
7: Calcular el incremento total de los pesos
8: Entrenar la red (*backpropagation*)
    Actualizar los pesos de la capa de salida
    Actualizar los pesos de la capa oculta
9: Obtener el mínimo coste global
10: Asignar etiqueta a v con los pesos ya entrenados

Un último aspecto a tener en cuenta es la posibilidad de convergencia hacia alguno de los mínimos locales de la función. Cuando la red encuentra un mínimo, sea

local o global cesa el aprendizaje y si el error tiene un valor aceptable el entrenamiento ha sido exitoso, pero en caso contrario se tendrá que recurrir a:

- Cambiar la arquitectura de la red (más capas ocultas)
- Modificar los parámetros de aprendizaje.
- Asignar un conjunto de pesos iniciales diferentes.
- Modificar el conjunto de entrenamiento o presentar los patrones en distinta secuencia.

3.4.3. RNA de función de base radial

Las redes neuronales de base radial (RBFN) se diferencian del perceptrón multicapa por tener una única capa oculta y no tener pesos asociados entre las primeras dos capas como se muestra en la Figura 3.12. En la capa oculta se utilizan las llamadas funciones de base radial para encontrar los centroides y en la capa de salida simplemente se realiza una combinación lineal de las activaciones de la capa anterior. Han sido aplicadas en una gran variedad de problemas como análisis de series temporales, procesamiento de imágenes, diagnósticos médicos y reconocimiento automático del habla [62].

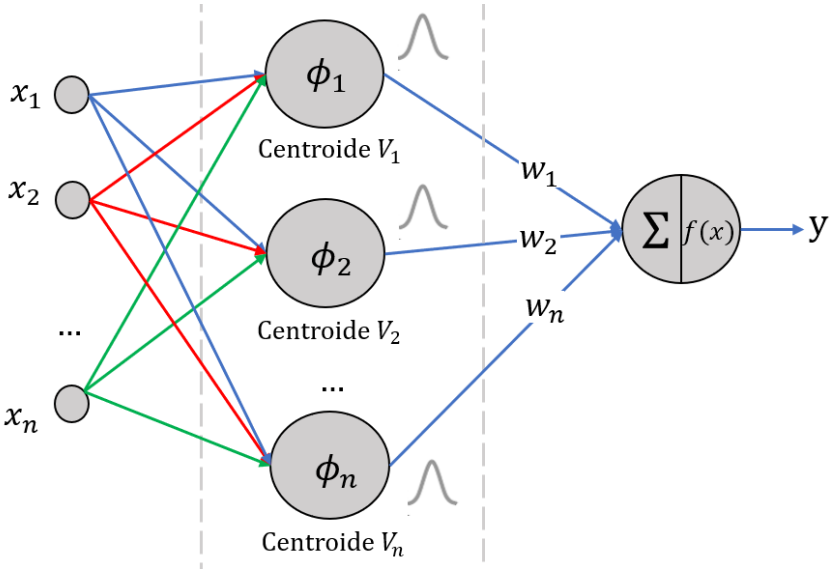


Figura 3.12: Representación de una red de base radial

Esta red utiliza dos algoritmos, uno para encontrar los centroides ϕ de cada conjunto de datos y otro para ajustar los pesos de la salida de la red, es decir, una fase no supervisada y otra supervisada.

- K-medias (*Clustering*). Este algoritmo trabaja iterativamente para agrupar los datos de entrada en base a la similitud de sus características de tal forma que se obtienen los centros de cada grupo para poder etiquetar nuevas muestras. Las muestras que se parezcan tendrán una menor distancia entre ellas.

- Fase supervisada. En este paso se calculan los pesos de la capa de salida. El entrenamiento se realiza para minimizar la función del error de la red, definida como:

$$E = \frac{1}{N}\sum_{n=1}^{N} e(n) \tag{3.21}$$

donde N representa el número de muestras y e(n) el error cometido por la red para cada x_n. Este problema de optimización suele resolverse por mínimos cuadrados o con la matriz pseudoinversa.

Algoritmo 5 : RBFN

Entrada: conjunto de entrenamiento $\{x\}$
Salida: RBF y pesos ajustados para etiquetar nuevas muestras

Primera parte: ***k-medias***
Determinar el número de conjuntos $\{K\}$
Inicializar aleatoriamente los centroides $\{ct\}$ de K
Para cada x_i:
    Calcular la distancia Euclidiana a cada centroide
        $d = \sqrt{\sum_{i=1}^{n}(x_i - ct_i)^2}$
    Buscar el centroide más cercano a x_i
    Asociar x_i al subconjunto Ω_i
Ajustar los centroides de cada subconjunto Ω_i con el promedio de sus x_i
Para cada ct:
    Calcular la desviación de cada función de activación gaussiana:
        $\sigma^2 = \frac{1}{m}\sum_{x_i \epsilon \Omega_i}\sum_{i=1}^{n}(x_i - ct_i)^2$, m = No. de muestras (x_i)
Iterar hasta que no haya cambio en los centroides
Segunda parte: ***mínimos cuadrados***
Obtener las etiquetas $\{y\}$ de cada x_i
Para cada x_i:
    Obtener las funciones gaussianas
        $\phi_n(x) = exp(-\frac{1}{2\sigma^2}||x - ct||^2)$
    Generar los subconjuntos = $\{\phi_1, \phi_2 ... \phi_n\}$
Obtener w = $(\phi^T\phi_n)^{-1}\phi_n^T y$

3.4.4. PCNN

Las redes neuronales pulso acopladas (*Pulse Coupled Neural Networks*) surgen a partir de las investigaciones de Eckhorn [63] en 1989 para emular la corteza visual de los gatos. Se han utilizado para diversas aplicaciones como mejoramiento de luminosidad de imágenes cerebrales de gran resolución [64], filtrado de ruido gaussiano [65], segmentación de imágenes [66] y localización de rostros [67].

$$F_{ij}[n] = \exp^{\alpha F \delta n} F_{ij}[n-1] + S_{ij} + V_F \sum klM_{ijkl}Y_{kl}[n-1] \tag{3.22}$$

$$L_{ij}[n] = \exp^{\alpha L \delta n} L_{ij}[n-1] + V_L \sum klW_{ijkl}Y_{kl}[n-1] \tag{3.23}$$

$$U_{ij}[n] = F_{ij}[n](1 + \beta L_{ij}[n]) \tag{3.24}$$

$$Y_{ij}[n] = \begin{cases} 1 & \text{si } U_{ij}[\text{n}] > \Theta_{ij}[\text{n-1}] \\ 0 & \text{en otro caso} \end{cases} \tag{3.25}$$

$$\Theta_{ij}[n] = \exp^{\alpha \Theta \delta n} \Theta_{ij}[n-1] + V_\Theta Y_{ij}[n] \tag{3.26}$$

Las ecuaciones 3.11- 3.15 definen el modelo PCNN original. F y L son las señales *feeding* y *linking*. En el proceso ambos canales se combinan para regular el potencial interno U de la red y posteriormente compararlo contra un umbral para generar la salida Y (Figura 3.13). Cada neurona está representada con los índices (i, j) y sus neuronas vecinas con los índices (k, l). M y W son los pesos sinápticos de los canales *feeding* y *linking*. V_L, V_F, $\exp^{\alpha F \delta n}$, $\exp^{\alpha L \delta n}$, $\exp^{\alpha \Theta \delta n}$, V_Θ y β son valores constantes, mientras que S es la imagen a procesar.

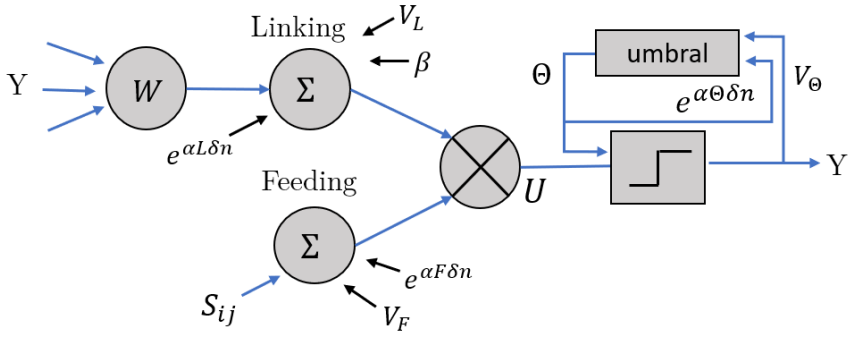


Figura 3.13: Representación esquemática de una red PCNN

Cuando la neurona se activa ($Y > \Theta$) el umbral aumenta significativamente su valor y decrece gradualmente hasta que se vuelve a activar la neurona. Esta característica hace que las neuronas correspondientes a píxeles vecinos con valores de intensidad similares se activen al mismo tiempo en ciertas regiones, a lo que se

denomina activación de pulsos síncrona. Para su implementación las matrices de M y W pueden tomar la siguiente forma:

$$(3x3) = \begin{bmatrix} 0.1 & 0.2 & 0.1 \\ 0.2 & 0 & 0.2 \\ 0.1 & 0.2 & 0.1 \end{bmatrix} \quad (5x5) = \begin{bmatrix} 0.12 & 0.25 & 0.50 & 0.25 & 0.12 \\ 0.25 & 0.50 & 1.00 & 0.50 & 0.25 \\ 0.50 & 1.00 & 0 & 1.00 & 0.50 \\ 0.25 & 0.50 & 1.00 & 0.50 & 0.25 \\ 0.12 & 0.25 & 0.50 & 0.25 & 0.12 \end{bmatrix} \tag{3.27}$$

El algoritmo consiste en calcular iterativamente las ecuaciones ya descritas hasta que el usuario lo considere conveniente. Desde la perspectiva del procesamiento de imágenes, las primeras iteraciones no son importantes, ya que todas las neuronas se activan en los primeros pulsos a pesar de asignar un umbral alto.

Algoritmo 6 : PCNN

Entrada: imagen en escala de grises $\{S\}$
Salida: múltiples imágenes binarias $\{Z\}$

1: Definir los parámetros de la red
   Algunos valores de muestra [68] son los siguientes:
   $\exp^{\alpha F \delta n} = 0.1$, $\exp^{\alpha L \delta n} = 1.0$, $\exp^{\alpha \Theta \delta n} = 1.0$
   $V_L = 0.2$, $V_F = 0.5$, $V_\Theta = 20.0$, $\beta = 0.1$
2: Inicializar las matrices M, W, F, L, Y, U, T con las dimensiones de S
3: Fijar un número máximo de iteraciones $\{N\}$
4: Normalizar entre 0 y 1 los valores de S
5: Z = lista vacía
6: Para n=1:N
   Obtener la salida de la red $\{y\}$ resolviendo las ecuaciones 3.11 - 3.15
   Añadir y a Z

A partir de la red PCNN surgieron otros modelos para reducir la complejidad del algoritmo original u optimizar la velocidad de cálculo. Entre ellos se encuentran *Unit-linking*, *Spiking*, multicanal e ICM [69].

Firma G[n]

La firma se utiliza para reducir las dimensiones de una imagen (se reduce la cantidad de operaciones y almacenamiento de datos). Esta técnica solo puede utilizarse en redes pulso acopladas y resulta conveniente en procesos de extracción de características por ser invariante a las distorsiones geométricas (escalado, rotación y traslación). Una firma se define como la suma de las neuronas que se activan en cada iteración, siendo un vector que se puede describir de la siguiente forma:

$$G[n] = \sum_{ij} Y_{ij}[n] \tag{3.28}$$

donde $Y_(ij)[n]$ es la secuencia de pulsos de la red ICM y G[n] una serie de tiempo que varía con el número de iteraciones.

3.4.5. ICM

El modelo de intersección cortical (*Intersection Cortical Model*) es una alternativa al presentado por Eckhorn cuyo propósito es facilitar el procesamiento de imágenes mas allá de replicar un sistema biológico. El esquema de la Figura 3.14 muestra las principales ventajas de la red ICM tras eliminar el canal de *linking* y las diversas constantes involucradas en la configuración original de una red pulsante, reduciéndose a solo tres. El valor de S permanece como un estímulo externo pero ahora la señal de *feeding* se compone por la última salida del vecindario de neuronas ponderada por una matriz W_{ijkl}.

Este nuevo modelo se rige bajo las ecuaciones:

$$F_{ij}[n+1] = fF_{ij}[n] + S_{ij} + W\{Y[n]\}_{ij}, \tag{3.29}$$

$$Y_{ij}[n+1] = \begin{cases} 1 & \text{si } F_{ij}[\text{n+1}] > \Theta_{ij}[\text{n}] \\ 0 & \text{en otro caso} \end{cases} \tag{3.30}$$

$$\Theta_{ij}[n+1] = g\Theta_{ij}[n] + hY_{ij}[n+1] \tag{3.31}$$

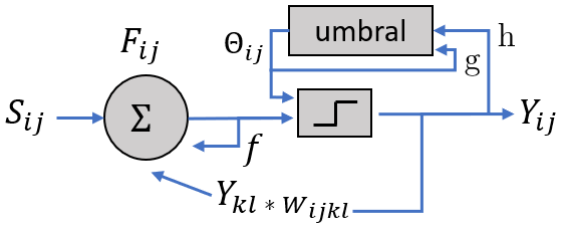


Figura 3.14: Diagrama de una red ICM

Pese a esta modificación, en la práctica los parámetros involucrados en este modelo computacional continúan limitando su rendimiento ocasionando que surjan distintas estrategias de optimización [70–72], es ahí donde surge el término "conexiones nulas" [73] (ICMNN, *Intersecting Cortical Model with Null Connections*) que más allá de realizar una elección automática de los parámetros de ajuste lo que se busca es reducir al mínimo el número de variables eliminando la influencia de los vecinos de cada neurona, simplificando aún más el modelo original de una red pulso acoplada a través las ecuaciones (3.32)-(3.34).

$$F(n) = fF(n-1) + S \tag{3.32}$$

$$E(n) = gE(n-1) + hY(n-1) \tag{3.33}$$

$$Y(n) = \begin{cases} 1, & F(n) > E(n) \\ 0, & en\ \ caso\ \ contrario \end{cases} \tag{3.34}$$

Entre los aportes más destacados se encuentran los de Jin, X. et al. [74], quienes demostraron que la información que este tipo de red emite puede ser analizada por la condición de activación continua de cada neurona, lo que permite restringir los parámetros al satisfacer la ecuación 3.35:

$$S > h\frac{1-f}{1-g} \tag{3.35}$$

donde S representa la imagen de entrada y *f, g* y *h* los valores de ajuste.

3.4.6. Descripción del hardware

Raspberry Pi es una computadora de una sola placa de bajo costo desarrollada en el Reino Unido por la Fundación Raspberry Pi [75] con el objetivo de ser accesible para los estudiantes. Entre los elementos principales que la conforman se encuentra un SoC, *System-on-chip*, que incluye la unidad central de procesamiento (CPU) y una unidad de procesamiento gráfico (GPU), una memoria de acceso aleatorio (RAM), puertos USB, HDMI, Ethernet, salidas de audio y video, conector para cámara, y lector de tarjetas micro SD que presentan menores o mayores prestaciones dependiendo del modelo (Figura 3.15).

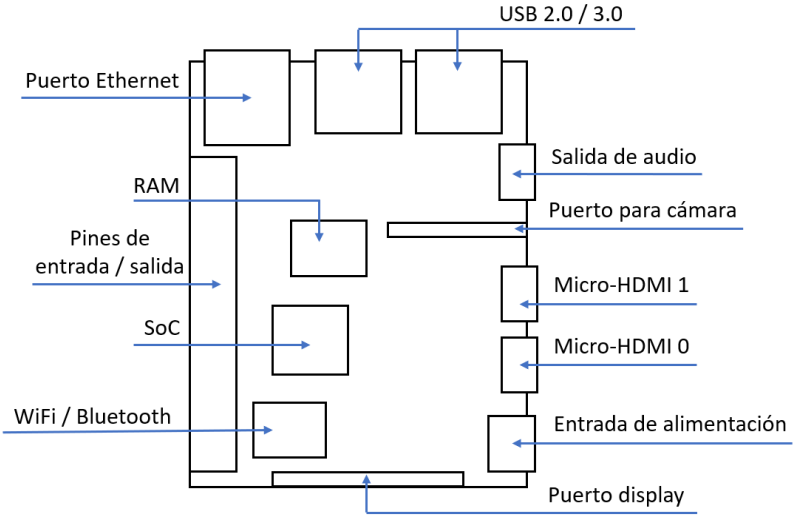


Figura 3.15: Elementos principales de Raspberry Pi 4 modelo B

Todas las técnicas que han sido descritas fueron implementadas durante el desarrollo del sistema de reconocimiento propuesto, sin embargo, solo algunas tuvieron un buen desempeño en el conjunto de datos de prueba. En el siguiente capítulo se presenta la metodología propuesta y el análisis de las alternativas que fueron descartadas.

Capítulo 4

Implementación

En este capítulo se describirán las técnicas aplicadas al sistema de reconocimiento y el diseño de la interfaz gráfica de usuario. La propuesta del método consiste de los siguientes pasos: 1) ubicación de la región de interés, 2) separación de los objetos y 3) reconocimiento.

4.1. Ubicación de la región de interés

En esta etapa, se separa la zona donde se encuentran los caracteres utilizando la red ICMNN para segmentar las imágenes del conjunto de datos de prueba, el cual consiste de 660 imágenes provenientes de diversos sitios web compatibles con el formato asignado al estado de Puebla. Todas comparten tres características, fueron tomadas durante el día, el rango captura es de 1 a 3 metros y tienen una vista frontal o trasera del automóvil. El formato de la matrícula incluye una serie de tres letras seguidas de cuatro números, un diseño utilizado en muchos estados de la República Mexicana. Además de Puebla, en el conjunto de datos también se incluyen placas de Morelos, Tlaxcala, Tabasco, Hidalgo, Quintana Roo, Guanajuato y la ciudad de México. En la Figura 4.1 se muestran algunas fotografías que conforman el banco de imágenes.

Figura 4.1: Algunas muestras del banco de imágenes

Tras analizar repetidas veces la información de salida de la red ICMNN, cuando los valores de f=0.85, g=0.15, h=15 y n=10 (parámetros de ajuste, véase sección

3.4.5) se satisface la condición de activación de todas las neuronas (ecuación 3.35) y también se logra que los caracteres sean legibles en alguna de las iteraciones. La cantidad de iteraciones para la segmentación se estableció en ese valor a razón de que se observó que a partir de la décima iteración la red comenzaba a repetir el patrón de salida. Posteriormente, se realizó la integración de todos los pulsos aplicando el operador unión dado que la información estadística de cada imagen binaria era altamente variable entre el conjunto de datos utilizado. En la Figura 4.2 se muestran ejemplos de segmentación aplicados al conjunto de prueba.

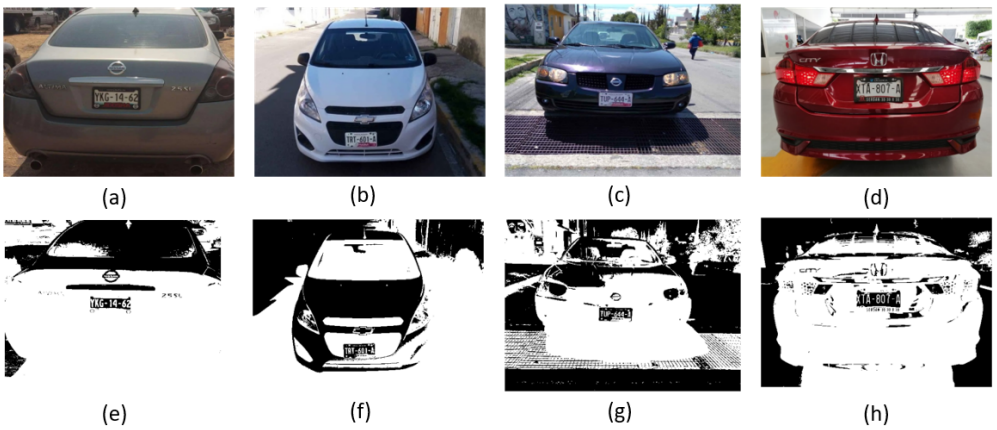


Figura 4.2: Resultados del proceso de segmentación

Para ubicar los caracteres dentro de la imagen segmentada se realizó una serie de filtrado de objetos como se observa en la Figura 4.3, utilizando como referencia características geométricas. Primero eliminando el área más grande correspondiente al vehículo empleando la técnica de componentes conectados y después limitando las dimensiones y la relación de aspecto de los posibles candidatos utilizando cuadros delimitadores generados por coordenadas. Posteriormentente a los objetos resultantes se les aplicó el operador dilatación para extraer los caracteres que pudieran haber sido descartados en la etapa anterior utilizando como estructura de referencia una matriz de 1x25.

Figura 4.3: (a) Resultado del proceso de segmentación por ICMNN. Proceso de filtrado de objetos (b) por área, (c) dimensiones y relación de aspecto, (d) forma geométrica y longitud

4.2. Separación de los objetos

Tras obtener la región de la matrícula, se aislaron los objetos de la zona tomando como referencia la imagen binaria original para eliminar los guiones entre caracteres y así aislar cada dígito y letra utilizando una vez más el operador dilatación con una nueva estructura de referencia de 50x1, para conservar las principales características de forma de los objetos. En la Figura 4.4 se aprecia que este técnica resultó efectiva en casos donde los caracteres estaban compuestos por más de cuatro objetos. Por último se empleó nuevamente la red ICMNN sin cambiar los parámetros de ajuste ya definidos en los caracteres por separado para facilitar el proceso de reconocimiento eliminando el ruido generado tomando como salida el último pulso.

La implementación del método propuesto para la ubicación de las regiones de interés y la separación de objetos se presenta en el algoritmo 7.

Algoritmo 7 : Algoritmo propuesto para la ubicación y separación de los caracteres

Entrada: imagen en escala de grises I_g
Salida: imágenes individuales de cada carácter

1: Estandarizar las dimensiones de I_g
2: Dividir el valor de cada píxel entre el máximo nivel de intensidad en I_g
3: Definir los parámetros de la red ICMNN condicionados por la ecuación 3.35 y sumar la constante obtenida a I_g para la segmentación
4: Aplicar el operador unión a los pulsos obtenidos para generar una sola imagen de salida I_s
5: Etiquetar la imagen $I_s \rightarrow I_e$
6: Para cada objeto en I_e:
    Obtener las coordenadas $[x, y]$
    Definir un rango para el área (base x altura = b*a) $\rightarrow 25 > b * a < 400$
    Definir un rango para la relación de aspecto $\rightarrow 0.1 > b/a < 2$
    Aplicar el operador dilatación, estructura de referencia 1x25
        Si $b > 3a$ & $b/a > 4$:
            Acceder al objeto en I_s y volver a etiquetar $\rightarrow I_E$
7: Para cada objeto en I_E:
    Obtener las coordenadas $[x, y]$
        Si $b! = 1$ & a> 6:
            Aplicar operador dilatación, estructura de referencia 50x1
            Aplicar segmentación con ICMNN, seleccionando el último pulso

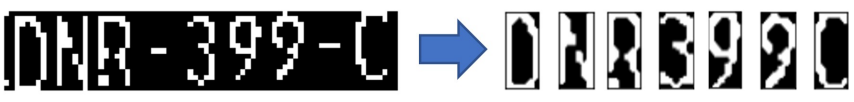


Figura 4.4: Eliminación de ruido y separación de los objetos

4.3. Reconocimiento

En este proceso se les asigna una etiqueta única a cada carácter para obtener la cadena de valores alfanuméricos correspondientes a la matrícula. Para reconocer el patrón de cada letra y número se ingresa cada objeto a un MLP previamente entrenado que tiene la arquitectura de la Figura 4.5 utilizando un conjunto datos de 6600 imágenes que incluye dígitos del 0 al 9 y las letras mayúsculas del alfabeto latino de la A a la Z excluyendo la I, Ñ, O y la Q dado que no aparecen en la nomenclatura de las matrículas de Puebla.

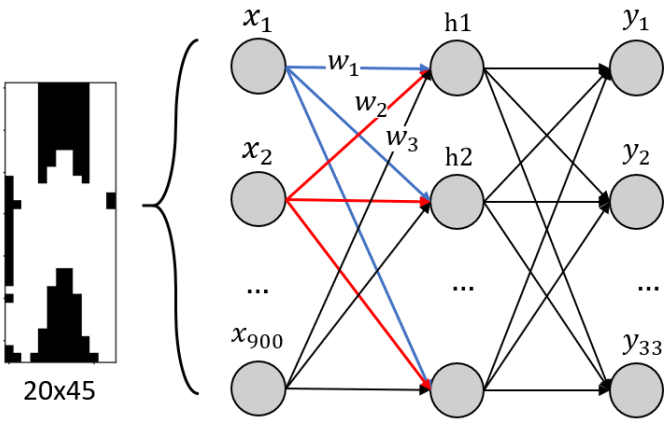


Figura 4.5: Configuración del perceptrón multicapa para 33 clases

Para elegir el número de neuronas en la capa oculta, se llevaron a cabo pruebas de validación cruzada con diferentes arquitecturas dividiendo aleatoriamente el conjunto de datos en cinco grupos y eligiendo el de mayor precisión. Durante el entrenamiento también se emplearon las imágenes de Scikit learn y EMNIST [76] para comparar el desempeño del MLP, no obstante, al final se descartaron por su incompatibilidad con el conjunto de datos de prueba siendo utilizados únicamente para definir los parámetros iniciales de convergencia. Ambos conjuntos de datos incluyen dígitos del 0 al 9 pero EMNIST incluye además letras mayúsculas y minúsculas. El término *balanced* hace referencia al subconjunto utilizado de la base de datos de EMNIST.

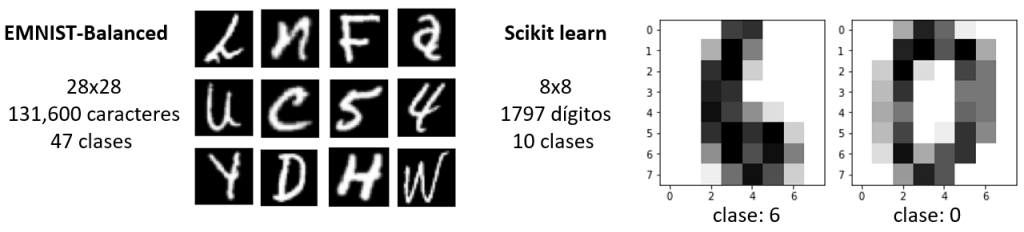


Figura 4.6: Ejemplos de EMNIST y Scikit learn

Tabla 4.1: Características de los conjuntos de datos

Nombre / Uso	No de muestras	Tamaño (píxeles)	Tipo de información
Conjunto de entrenamiento	6,600	20x45	dígitos y letras mayúsculas
Conjunto de prueba	660	900x700	fotos de automóviles con vista frontal o trasera
EMNIST	131,600	28x28	dígitos, letras mayúsculas y minúsculas escritas a mano
Scikit learn	1,797	8x8	dígitos

En la Tabla 4.1 se resumen las características de los conjuntos de datos utilizados durante la implementación del método.

Aumento de datos

Debido al balanceo de las clases, del conjunto de datos creado para el proceso de entrenamiento del MLP solo se pudieron seleccionar 100 ejemplos por etiqueta, lo cual afectó el entrenamiento del perceptrón multicapa disminuyendo el porcentaje de aciertos. Por esta razón se empleó la técnica de dilatación utilizando la estructura de referencia de la ecuación 4.1 sobre los 33,300 ejemplos para duplicar la cantidad de muestras.

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{4.1}$$

Para ingresar las muestras a la red, primero se categorizó cada imagen en carpetas según el número o letra que representaba (0, 1, 2, 3, etc.), después se convirtieron a escala de grises normalizando los valores entre 0 y 1 y, por último, se manipularon como vector para añadirlos a una matriz que definiría el conjunto de entrenamiento y validación que se utilizaría durante el entrenamiento del MLP. Este proceso se llevó a cabo por los siguientes pasos:

1. Revolver aleatoriamente el conjunto de muestras y destinar 1/5 parte al set de validación
2. Asociar todos los vectores de entrada con su salida deseada $\{x_i, y_i\}$ utilizando la codificación *one hot*
3. Asignar valores aleatorios pequeños a los pesos $\{w\}$
4. Definir una tasa de aprendizaje (η) y un momento (α)
5. Para cada muestra de entrenamiento obtener la salida de la red

$$ai = \sum_{i=1}^{n} w_n x_n \ ; \ \ y_{real} = h(ai) \tag{4.2}$$

6. Evaluar el error cuadrático medio

$$ECM = \sum \frac{1}{2} * (y_{ideal}) - y_{real})^2 \tag{4.3}$$

7. Calcular el incremento parcial de los pesos utilizando el descenso del gradiente y la regla de la cadena.

$$\nabla E = \frac{\partial E}{\partial wo} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial ao} \cdot \frac{\partial ao}{\partial wo} \tag{4.4}$$

$$\frac{\partial E}{\partial wh} = \frac{\partial E}{\partial h} \cdot \frac{\partial h}{\partial ai} \cdot \frac{\partial ai}{\partial wh} \tag{4.5}$$

8. Actualizar los pesos de la capa de salida

$$wo = wh + \alpha(wo - wo) - \eta \cdot \frac{\partial E}{\partial wo} \tag{4.6}$$

9. Actualizar los pesos de la capa oculta

$$wh = wh + \alpha(wh - wh) - \eta \cdot \frac{\partial E}{\partial wh} \tag{4.7}$$

10. Obtener el mínimo coste global y definir un número máximo de épocas

La codificación *one hot* es un método de conversión de datos que se utilizó para encontrar una probabilidad más cercana a cada clase.

4.4. Análisis de alternativas realizadas

Durante las pruebas realizadas, se tuvo oportunidad de explorar diferentes técnicas que no habían sido aplicadas al reconocimiento de matrículas con características mexicanas. En la Tabla 4.2 se muestra la comparación entre tres métodos de segmentación más utilizados en el procesamiento de imágenes, el umbral global con promedio, Otsu, clustering con k-means y uno no tal convencional como lo es la red ICM.

Tabla 4.2: Comparación entre métodos de segmentación

Método	Descripción	Aciertos	Observaciones
Umbral global con promedio	Si la intensidad del píxel es mayor al promedio, el píxel de destino se establece al máximo valor definido (255 o 1), en caso contrario, cero	324	Los objetos deben tener un alto contraste y las imágenes deben tener buena resolución.

Otsu	El umbral óptimo se logra cuando la varianza entre clases genera un valor mínimo	412	Se necesitan más características para descartar regiones de interés.
Clustering k-means	Divide los píxeles de la imagen en grupos con características similares, para ello se consideró la distancia entre muestras	579	El tiempo de procesamiento es mayor, siendo alrededor de un minuto por imagen.
ICM	Genera una señal cuando el píxel logra superar un umbral para poder analizar su vecindad	425	En imágenes con altas variaciones en los niveles de gris resulta complicado elegir el pulso de salida.

Los métodos de segmentación se evaluaron en 730 imágenes imágenes debido a que se tuvo que crear el conjunto de datos recopilando fotografías de los automóviles y las pruebas se realizaban con el número de imágenes con las que se disponía en ese momento. Otro tipo de técnica que también se implementó fue el uso de proyecciones con el objetivo de eliminar la zona del porta placas para obtener solo la región de los caracteres, pero se descartó porque algunos componentes conectados centrales se unían. Entre las métricas que se utilizaron para elegir el pulso de salida de la red ICMNN, se encuentra la entropía de Shannon, el promedio y la desviación estándar (Figura 4.7), sin embargo, no se logró encontrar un patrón que coincidiera en todo el conjunto de prueba por lo que se optó por el operador unión.

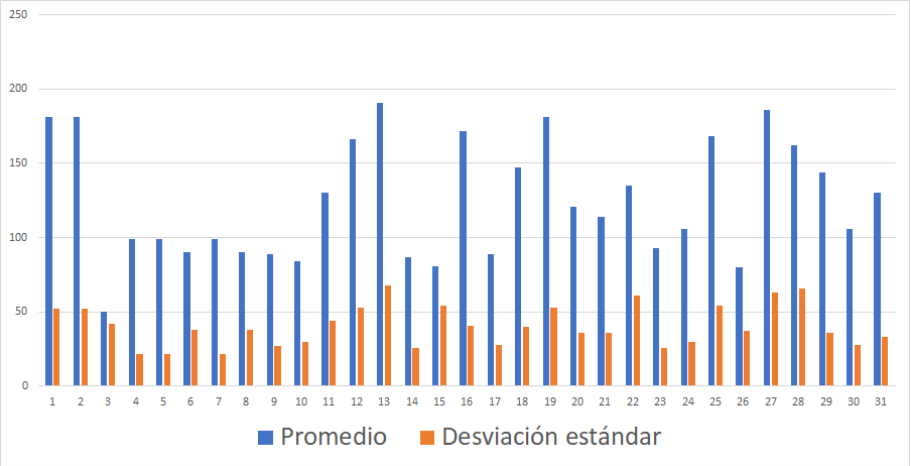
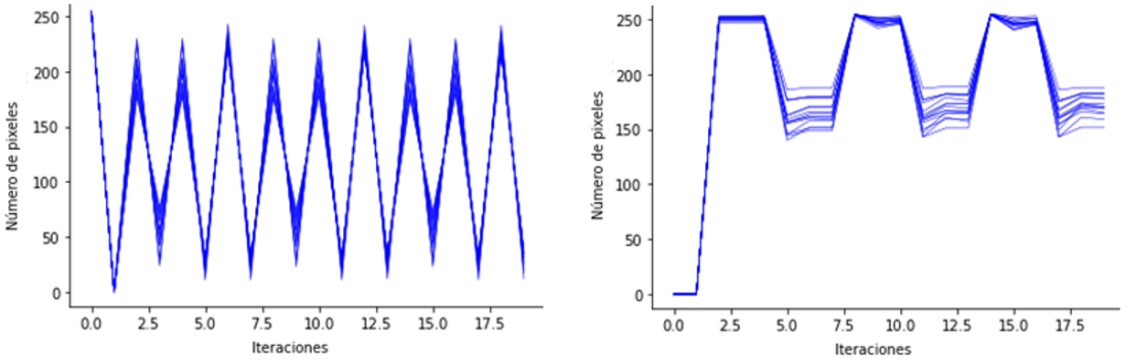


Figura 4.7: Promedio y desviación estándar de 31 placas

Para verificar que las características que se obtuvieron fueran las correctas en la etapa previa a la separación de los objetos, se utilizó la firma y la secuencia de

entropía para realizar la clasificación entre placas y no placas.



(a) (b)

Figura 4.8: (a) Firma y (b) secuencia de entropía con ICM de 15 placas

Como se observa en la Figura 4.8, la firma y la secuencia de entropía presentan patrones muy similares entre las regiones resultantes, lo que logró reflejarse en la razón de reconocimiento obtenida al evaluarlo en 411 imágenes donde se alcanzó un 88.8 % con la firma y 92.4 % con la secuencia de entropía. No obstante, resulta ineficaz para la clasificación de múltiples objetos. Se eligieron únicamente 15 placas a razón de que en todas las imágenes la firma y la secuencia de entropía eran la misma.

Durante en análisis de las regiones de interés, también se consideró utilizar el esqueleto pero no fue viable dado que en algunas regiones el número de bordes es mayor que la que contiene la matrícula. Este problema se atribuyó a las pegatinas y adornos en el vehículo. En el siguiente capítulo se mostrarán los resultados del entrenamiento y la interfaz de usuario del sistema de reconocimiento propuesto.

Capítulo 5

Resultados

En este capítulo se muestran los resultados que se obtuvieron de las diferentes pruebas realizadas en la fase de implementación.

Para establecer un valor inicial a los parámetros del MLP se utilizaron dos bases de datos, EMNIST y Scikit-learn. Se realizaron diferentes pruebas considerando respetando una arquitectura piramidal comenzando de 10 a 250 neuronas en la capa oculta. El porcentaje de precisión se obtuvo evaluando las muestras de entrenamiento

Tabla 5.1: Diferentes tasas de reconocimiento(%) con η =0.5

Conjunto de datos	No. de neuronas en la capa oculta	No. de Épocas	ECM	Porcentaje de precisión
EMNIST	30	500	0.246	74.240
	30	20000	0.073	92.200
	50	500	0.210	79.300
	50	20000	0.072	92.530
Scikit learn	10	500	0.107	90.278
	10	5000	0.019	95.556
	10	20000	0.003	96.944
	30	5000	0.004	98.611
	30	10000	0.001	98.889
	30	20000	0.000	99.167
	50	5000	0.002	98.333
	50	10000	0.000	98.611
	50	20000	0.000	98.889

y validación (divididas aleatoriamente) comparándolas con la etiqueta correcta y luego multiplicando el número de aciertos por 100. En la Tabla 5.1 se presenta un resumen de las pruebas intercambiando algunos parámetros. Como se observa, en EMNIST se logra un porcentaje de precisión aceptable con 50 neuronas en la capa oculta y 20,000 épocas, mientras que con los mismos valores en el conjunto de Scikit learn se obtiene un porcentaje de precisión más cercano al 100 %. Esto se le atribuye al tamaño que tienen las imágenes de ambos conjuntos (8x8 y 28x28, véase Tabla 4.1) y el número de etiquetas disponibles, 47 de EMNIST contra 10 de Scikit learn.

Partiendo de los datos anteriores, se estableció un rango de valores para la convergencia del MLP utilizando el conjunto de datos de entrenamiento.

Tabla 5.2: Resumen de las pruebas de entrenamiento al cambiar los parámetros.

Tasa de aprendizaje	Neuronas ocultas	Momento	Épocas	Predicción correcta (%)		ECM	Tiempo
				Entrenamiento	Validación		
22 imágenes x 33 clases, total: 726, validación: 145							
0,3	50	0,5	10,000	100,0	77,931	0,0011	3 min
0,3	50	0,5	30000	100,0	83,448	0,0000	8 min
0,3	50	0,5	50,000	100,0	79,310	0,0000	16 min
100 imágenes x 33 clases, total: 3300, validación: 660							
0,1	180	0,2	30,000	99,8106	95,4545	0.0092	57 min
200 imágenes x 33 clases, total: 6600, validación: 1320							
0,5	250	0,5	30,000	99,981	98,0303	0.0013	2h

En la Tabla 5.2 se muestran diferentes combinaciones de parámetros para el conjunto de datos de prueba con variaciones en el número de muestras. Aunque en el caso a) se obtuvo un porcentaje de precisión del %100 para el conjunto de entrenamiento, en el conjunto de validación no era cercano ni siquiera al 90 % por lo que se recurrió al caso b) que a pesar de arrojar resultados favorables, no logró ser suficiente y se tuvo que recurrir a la técnica de aumento de datos para duplicar las muestras con las que ya se contaba, de lo cual se deduce que el número de neuronas en la capa oculta y el rendimiento del MLP está ligado a la cantidad de ejemplos que estén disponibles. El tiempo reportado corresponde a la ejecución del script implementado en el lenguaje de Python utilizando el entorno de desarrollo de Spyder en Windows 10 con un procesador Intel Core i7 y 8GB de memoria RAM.

La tasa de aprendizaje se fijó considerando el comportamiento de la Fig.5.1 evaluando el decremento del error cuadrático medio y el aumento del porcentaje de aciertos para descartar un subajuste o sobreentrenamiento de la red, lr representa

el valor de la tasa de aprendizaje.

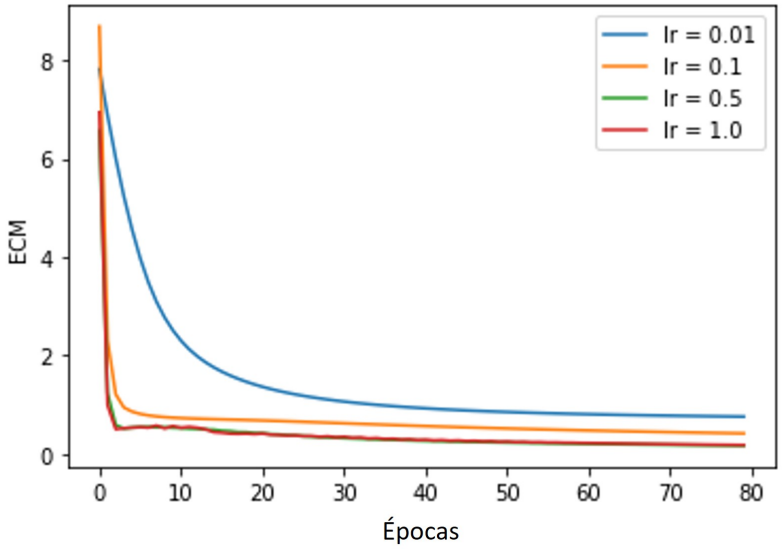


Figura 5.1: Diferentes curvas de aprendizaje.

Además del MLP, durante el proceso de reconocimiento también se utilizó una red de base radial y el método WKNN. Como se observa, la red de base radial (Tabla 5.3) alcanza un porcentaje similar al MLP con menos neuronas en la capa oculta por lo que puede utilizarse para definir este parámetro inicial.

Tabla 5.3: Tasa de reconocimiento con red de base radial

RBFN, validación: 145, entrenamiento: 580, t = 25min			validación cruzada kFold = 10, t = 3h20min	
	Predicción correcta (%)		Predicción correcta (%)	
Neuronas ocultas	Entrenamiento	Validación	Entrenamiento	Validación
70	96,901	82,068		
100	98,106	91,034	98,425	91,805
120	98,967	96,551	99,027	93,055
125	99,483	93,103		
130	98,795	92,413		
150	99,139	88,275		

En la Tabla 5.4 se muestran los resultados de la clasificación de las muestras del conjunto de pruebas con WKNN. En ella se observa una mayor precisión utilizando un número de k vecinos menores a 10 y que el tiempo de procesamiento es menor en comparación a los algoritmos previamente expuestos aunque no se logra superar el 95 % de precisión en el conjunto de validación por la semejanza entre algunos caracteres como el número 8 y la letra B. En la Figura 5.2 se muestran algunos ejemplos de imágenes obtenidas durante el proceso de reconocimiento. Como se

Tabla 5.4: Tasa de reconocimiento con WKNN

WKNN, validación: 145, entrenamiento: 580, t = 1min			t = 7min
	Predicción correcta		
No. de vecinos	Validación (%)	No. de caracteres	kFold = 10
2	95,862	139	94,444
3	93,103	135	93,611
4	91,724	133	93,333
5	93,793	136	92,083
6	91,034	132	93,055
7	92,413	134	91,666
8	91,034	132	91,666
9	88,275	128	91,253
10	87,586	127	91,388
11	88,965	129	90,833
12	88,275	128	90,138
13	88,275	128	90,972

observa, durante el proceso de segmentación se logra eliminar detalles de la placa que no pertenecen al número de matrícula.

Figura 5.2: Resultados del método de reconocimiento de matrículas

5.1. Descripción de la Interfaz

La interfaz gráfica de usuario consta de una sola ventana principal (Figura 5.3). En la parte superior izquierda se encuentra un botón (1) para subir el archivo de excel que contiene las placas autorizadas para realizar las comparaciones y enviar las alertas, las cuales se identifican visualmente en la tabla de información de accesos (2) con un "1". En la parte inferior (3) se muestra la secuencia de caracteres obtenidos

por el algoritmo y la fecha de captura.

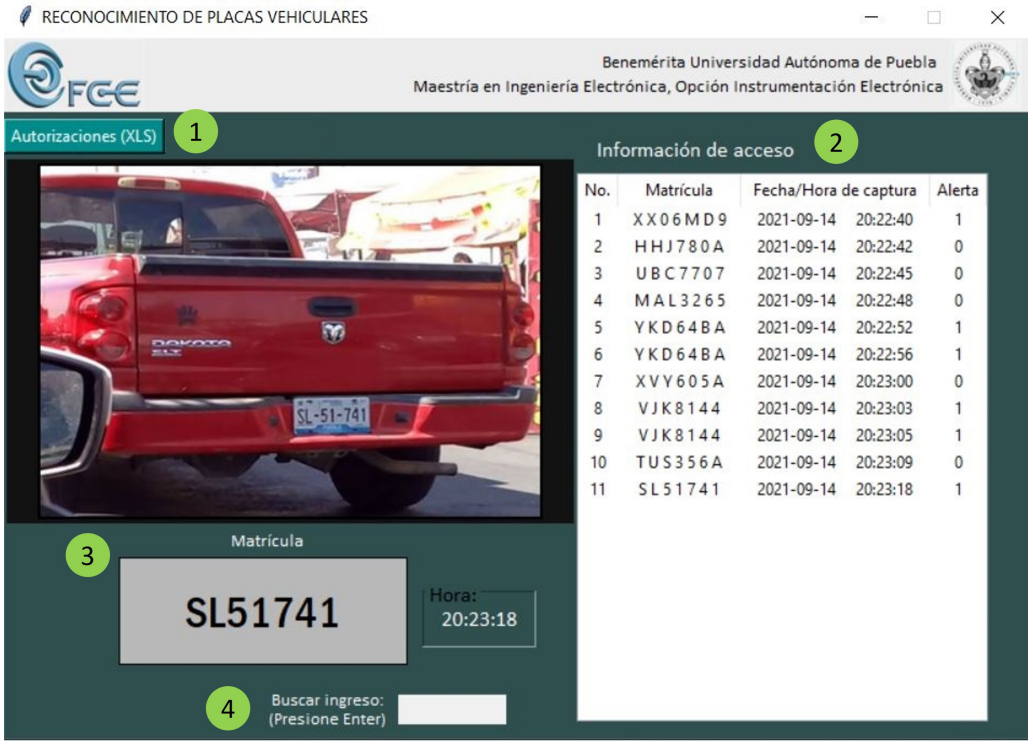


Figura 5.3: Ventana principal

También se incorpora una etiqueta de entrada (4) para buscar el ingreso de un vehículo de en específico. Las posibles respuestas se muestran en la Figura 5.4.

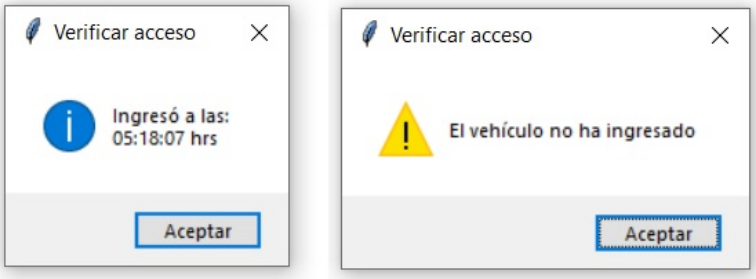


Figura 5.4: Verificación de ingreso

En la Figura 5.5 se muestra la alerta enviada vía correo electrónico. La alerta se envía cuando no existe coincidencia en todos los caracteres de las matrículas en la base de datos de accesos añadida por el usuario. Como se observa, además de enviar los datos de la matrícula se añaden las posibles coincidencias encontradas en el registro proporcionado por el usuario para que él mismo pueda descartar errores del sistema.

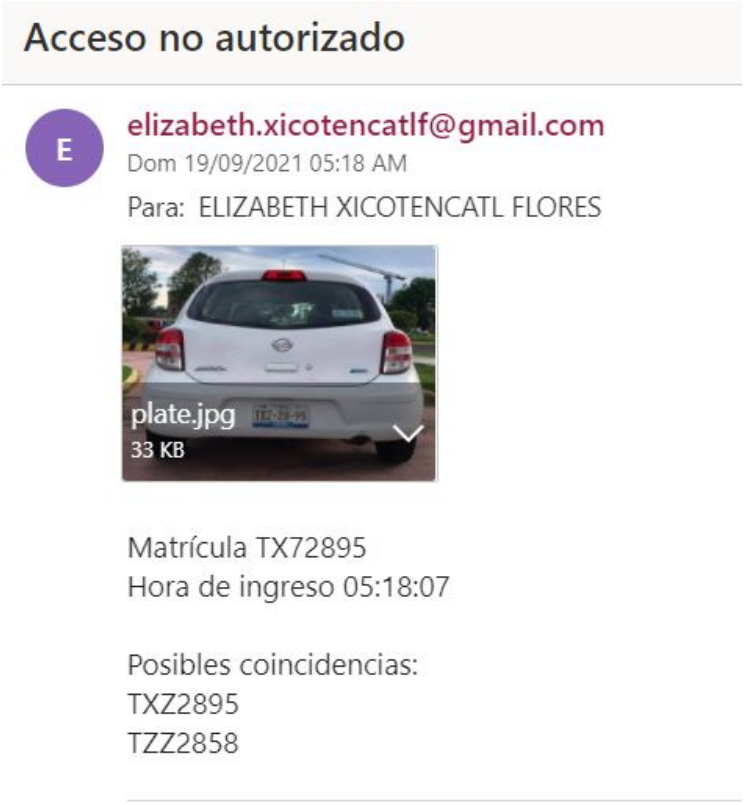


Figura 5.5: Notificación por Email

En promedio, el tiempo de ejecución por captura en la PC es de 1.6 segundos y en la raspberry pi de 1.8 segundos a los que adicionalmente se agregan 13 seg. para el enviar el correo.

Capítulo 6

Conclusiones

Después de haber hecho la implementación y las pruebas comparativas las conclusiones a las que se llega en este trabajo son las siguientes.

- Las redes pulso acopladas pueden aprovecharse al máximo en procesos de segmentación al ser un algoritmo que busca automáticamente los umbrales óptimos que dividen los valores de intensidad de los píxeles aprovechando los resultados de cada iteración. A diferencia de otras técnicas de segmentación, se logra observar un mejor desempeño en imágenes con altas variaciones.
- Los resultados obtenidos presentan una precisión del 96.2 %, lo cual demuestra que el método propuesto es ligeramente robusto en comparación a los presentados por otros autores, teniendo una baja sensibilidad a los cambios de iluminación y sombras.
- La implementación del sistema en la raspberry pi es una opción viable por tener un tiempo de respuesta aceptable además de ser de bajo costo.

Para lograr que el sistema de reconocimiento aumente el porcentaje de precisión en ambientes complejos en trabajos futuros se propone ajustar la matrícula a un solo perfil para evitar errores por efecto de la inclinación utilizando el centroide de los objetos y aumentar el conjunto de datos de entrenamiento y de prueba.

Bibliografía

[1] A. Essa and V. K. Asari, "Face recognition based on modular histogram of oriented directional features," in *Proc. IEEE National Aerospace and Electronics Conf. (NAECON) and Ohio Innovation Summit (OIS)*, 2016, pp. 49–53.

[2] P. K. Sethy, L. Panda, and S. K. Behera, "Ann based image restoration in approach of multilayer perceptron," in *Proc. Int. Conf. Inventive Computation Technologies (ICICT)*, vol. 2, 2016, pp. 1–4.

[3] C.-C. Huang, Y.-S. Tai, and S.-J. Wang, "Vacant parking space detection based on plane-based bayesian hierarchical framework," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 9, pp. 1598–1610, 2013.

[4] S. Mandal, V. Daivajna, and V. Rajagopalan, "Machine learning based system for automatic detection of leukemia cancer cell," in *2019 IEEE 16th India Council International Conference (INDICON)*. IEEE, 2019, pp. 1–4.

[5] M. Rajalakshmi, P. Saranya, and P. Shanmugavadivu, "Pattern recognition-recognition of handwritten document using convolutional neural networks," in *2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS)*. IEEE, 2019, pp. 1–7.

[6] H. Hongping and B. Yanping, "A kind of license plate location based on mathematical morphology and edge detection," in *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology*, vol. 5. IEEE, 2011, pp. 2291–2294.

[7] S. Israni and S. Jain, "Edge detection of license plate using sobel operator," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*. IEEE, 2016, pp. 3561–3563.

[8] H. Lin, J. Zhao, S. Li, and G. Qiu, "License plate location method based on edge detection and mathematical morphology," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1. IEEE, 2020, pp. 853–857.

[9] S. Das and R. Kumari, "Application of horizontal projection lines in detecting vehicle license plate," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*. IEEE, 2020, pp. 1–6.

[10] C.-C. Liu and Z.-C. Luo, "Extraction of vehicle license plate number using license plate calibration," 2010.

[11] A. Zweng and M. Kampel, "High performance implementation of license plate recognition in image sequences," in *International Symposium on Visual Computing*. Springer, 2009, pp. 598–607.

[12] Y. Qian and S. J. Gan, "Research of license plate recognition based on hsv space," 2016.

[13] L. Li and F. Guangli, "The license plate recognition system based on fuzzy theory and bp neural network," in *2011 Fourth International Conference on Intelligent Computation Technology and Automation*, vol. 1. IEEE, 2011, pp. 267–271.

[14] A. H. Ashtari, M. J. Nordin, and M. Fathy, "An iranian license plate recognition system based on color features," *IEEE transactions on intelligent transportation systems*, vol. 15, no. 4, pp. 1690–1705, 2014.

[15] Y. Wu, S. Liu, and X. Wang, "License plate location method based on texture and color," in *2013 IEEE 4th International Conference on Software Engineering and Service Science*. IEEE, 2013, pp. 361–364.

[16] Y. Shima, "Extraction of number plate images based on image category classification using deep learning," in *2016 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*. IEEE, 2016, pp. 19–26.

[17] Y. Dong, M. Pei, and X. Qin, "Vehicle color recognition based on license plate color," in *2014 tenth international conference on computational intelligence and security*. IEEE, 2014, pp. 264–267.

[18] J. H. Santiago, J. S. R. Castilla, C. H. M. Montiel, and B. H. Santiago, "Segmentación de placas vehiculares usando haar-adaboost y clustering." *Res. Comput. Sci.*, vol. 147, no. 5, pp. 269–279, 2018.

[19] S. Nigussie and Y. Assabie, "Automatic recognition of ethiopian license plates," in *AFRICON 2015*. IEEE, 2015, pp. 1–5.

[20] H. Huang, M. Gu, and H. Chao, "An efficient method of license plate location in natural-scene image," in *2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 4. IEEE, 2008, pp. 15–19.

[21] M.-K. Wu, J.-S. Wei, H.-C. Shih, and C. C. Ho, "2-level-wavelet-based license plate edge detection," in *2009 Fifth International Conference on Information Assurance and Security*, vol. 2. IEEE, 2009, pp. 385–388.

[22] E. Medvedeva, I. Trubin, and P. Kasper, "Vehicle license plate recognition based on edge detection," in *2020 26th Conference of Open Innovations Association (FRUCT)*. IEEE, 2020, pp. 291–296.

[23] Z. Qin, S. Shi, J. Xu, and H. Fu, "Method of license plate location based on corner feature," in *2006 6th World Congress on Intelligent Control and Automation*, vol. 2. IEEE, 2006, pp. 8645–8649.

[24] F. Jun and D. Shuguang, "A vehicle license plate location and correction method based the characteristics of license plate," in *Proceedings of the 10th world congress on Intelligent Control and Automation*. IEEE, 2012, pp. 42–46.

[25] B. Chen, W. Cao, and H. Zhang, "An efficient algorithm on vehicle license plate location," in *2008 IEEE International Conference on Automation and Logistics*. IEEE, 2008, pp. 1386–1389.

[26] L. Angeline, W. L. Khong, F. Wong, I. Saad, and K. T. K. Teo, "Multiple vehicles license plate tracking and recognition via isotropic dilation," in *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*. IEEE, 2011, pp. 54–59.

[27] Z. Liu and Y. Zhu, "Vehicle license plate recognition in complex scenes," in *2020 IEEE 5th International Conference on Intelligent Transportation Engineering (ICITE)*. IEEE, 2020, pp. 235–239.

[28] Y. Elhadi, O. Abdalshakour, and S. Babiker, "Arabic-numbers recognition system for car plates," in *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*. IEEE, 2019, pp. 1–6.

[29] D. Menotti, G. Chiachia, A. X. Falcao, and V. O. Neto, "Vehicle license plate recognition with random convolutional networks," in *2014 27th SIBGRAPI conference on graphics, patterns and images*. IEEE, 2014, pp. 298–303.

[30] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafas, "A license plate-recognition algorithm for intelligent transportation system applications," *IEEE Transactions on Intelligent transportation systems*, vol. 7, no. 3, pp. 377–392, 2006.

[31] J.-K. Chang, S. Ryoo, and H. Lim, "Real-time vehicle tracking mechanism with license plate recognition from road images," *The Journal of Supercomputing*, vol. 65, no. 1, pp. 353–364, 2013.

[32] V. Laxmi and H. Rohil, "License plate recognition system using back propagation neural network," *International Journal of Computer Applications*, vol. 99, no. 8, pp. 29–37, 2014.

[33] E. I. Abbas and T. A. Hashim, "Iraqi cars license plate detection and recognition system using edge detection and template matching correlation," *Eng. & Tech. Journal*, vol. 34, pp. 257–271, 2016.

[34] A. K. Bachchan, A. Gorai, and P. Gupta, "Automatic license plate recognition using local binary pattern and histogram matching," in *International Conference on Intelligent Computing*. Springer, 2017, pp. 22–34.

[35] S. Montazzolli and C. Jung, "Real-time brazilian license plate detection and recognition using deep convolutional neural networks," in *2017 30th SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*. IEEE, 2017, pp. 55–62.

[36] P. Dhar, S. Guha, T. Biswas, and M. Z. Abedin, "A system design for license plate recognition by using edge detection and convolution neural network," in *2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2)*. IEEE, 2018, pp. 1–4.

[37] Z. Selmi, M. B. Halima, U. Pal, and M. A. Alimi, "Delp-dar system for license plate detection and recognition," *Pattern Recognition Letters*, vol. 129, pp. 213–223, 2020.

[38] W. Wang, J. Yang, M. Chen, and P. Wang, "A light cnn for end-to-end car license plates detection and recognition," *IEEE Access*, vol. 7, pp. 173 875–173 883, 2019.

[39] C. Henry, S. Y. Ahn, and S.-W. Lee, "Multinational license plate recognition using generalized character sequence detection," *IEEE Access*, vol. 8, pp. 35 185–35 199, 2020.

[40] I. V. Pustokhina, D. A. Pustokhin, J. J. Rodrigues, D. Gupta, A. Khanna, K. Shankar, C. Seo, and G. P. Joshi, "Automatic vehicle license plate recognition using optimal k-means with convolutional neural network for intelligent transportation systems," *IEEE Access*, vol. 8, pp. 92 907–92 917, 2020.

[41] M.-X. He and P. Hao, "Robust automatic recognition of chinese license plates in natural scenes," *IEEE Access*, vol. 8, pp. 173 804–173 814, 2020.

[42] Q. Huang, Z. Cai, and T. Lan, "A single neural network for mixed style license plate detection and recognition," *IEEE Access*, vol. 9, pp. 21 777–21 785, 2021.

[43] H. Karwal and A. Girdhar, "Vehicle number plate detection system for indian vehicles," in *2015 IEEE International Conference on Computational Intelligence & Communication Technology*. IEEE, 2015, pp. 8–12.

[44] K. Indira, K. Mohan, and T. Nikhilashwary, "Automatic license plate recognition," in *Recent Trends in Signal and Image Processing*. Springer, 2019, pp. 67–77.

[45] M. Ismail, "License plate recognition for moving vehicles case: At night and under rain condition," in *2017 Second International Conference on Informatics and Computing (ICIC)*. IEEE, 2017, pp. 1–4.

[46] M. J. Ahmed, M. Sarfraz, A. Zidouri, and W. G. Al-Khatib, "License plate recognition system," in *10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003*, vol. 2. IEEE, 2003, pp. 898–901.

[47] N. A. Jalil, A. Basari, S. Salam, N. K. Ibrahim, and M. A. Norasikin, "The utilization of template matching method for license plate recognition: A case study in malaysia," in *Advanced Computer and Communication Engineering Technology*. Springer, 2015, pp. 1081–1090.

[48] K.-H. Jo *et al.*, "A clustering strategy for touching characters in korean and english printed text segmentation," in *2012 9th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*. IEEE, 2012, pp. 23–25.

[49] F. Ariff, A. S. A. Nasir, H. Jaafar, and A. Zulkifli, "Character segmentation for automatic vehicle license plate recognition based on fast k-means clustering," in *2020 IEEE 10th International Conference on System Engineering and Technology (ICSET)*. IEEE, 2020, pp. 228–233.

[50] L. Zheng and X. He, "Character segmentation for license plate recognition by k-means algorithm," in *International Conference on Image Analysis and Processing*. Springer, 2011, pp. 444–453.

[51] A. R. F. Quiros, R. A. Bedruz, A. C. Uy, A. Abad, A. Bandala, E. P. Dadios, and A. Fernando, "A knn-based approach for the machine vision of character recognition of license plate numbers," in *TENCON 2017-2017 IEEE Region 10 Conference*. IEEE, 2017, pp. 1081–1086.

[52] S. Escalante and V. Murray, "Automatic recognition of peruvian car license plates," in *2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*. IEEE, 2020, pp. 1–4.

[53] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafas, "A license plate-recognition algorithm for intelligent transportation system applications," *IEEE Transactions on Intelligent transportation systems*, vol. 7, no. 3, pp. 377–392, 2006.

[54] E. Cuevas, D. Zaldívar, and M. Pérez, *Procesamiento digital de imágenes con Matlab y Simulink*. Alfaomega, 2010.

[55] W. Burger and M. J. Burge, *Digital image processing: an algorithmic introduction using Java*. Springer, 2016.

[56] G. P. Martinsanz, *Imágenes digitales, Procesamiento práctico con Java*. Alfaomega, 2003.

[57] R. Akerkar, *Artificial Intelligence for Business*. Springer, 2019.

[58] S. Marsland, *Machine learning: an algorithmic perspective*. CRC press, 2015.

[59] S. Raschka, *Python machine learning*. Packt publishing ltd, 2015.

[60] I. N. da Silva and D. H. Spatti, *Artificial Neural Networks, A Practical Course*. Springer, 2017.

[61] B. MARTIN and A. SANZ MOLINA, *Redes neuronales y sistemas difusos*, 2002.

[62] P. Isasi and I. Galván, "Redes de neuronas artificiales. un enfoque práctico," *Madrid. España, Pearson Educación, SA*, 2004.

[63] T. Lindblad, J. M. Kinser, T. Lindblad, and J. Kinser, *Image processing using pulse-coupled neural networks*. Springer, 2005.

[64] M. Mejia-Lavalle, K. Aguilar, H. Sossa, D. Mujica, and A. Magadan, "Modelos neuronales pulsantes adaptados para el mejoramiento de luminosidad de imágenes cerebrales de gran resolución," *Research in Computing Science*, vol. 148, pp. 253–266, 2019.

[65] O. Estela, M. Manuel, and s. Humberto, "Filtrado de ruido gaussiano mediante redes neuronales pulso-acopladas," in *Proc. Fourth Int. Conf. Advanced Computing (ICoAC)*, 2012, pp. 1–8.

[66] H. Li and Z. Bai, "A new pcnn-based method for segmentation of sar images," in *2008 10th International Conference on Control, Automation, Robotics and Vision*. IEEE, 2008, pp. 1635–1639.

[67] H. Fan, D. Zhou, R. Nie, and D. Zhao, "Target face detection using pulse coupled neural network and skin color model," in *2012 International Conference on Computer Science and Service System*. IEEE, 2012, pp. 2185–2188.

[68] Z. Wang, Y. Ma, F. Cheng, and L. Yang, "Review of pulse-coupled neural networks," *Image and Vision Computing*, vol. 28, no. 1, pp. 5–13, 2010.

[69] Y. Ma, K. Zhan, and Z. Wang, *Applications of pulse-coupled neural networks*. Springer, 2010.

[70] D. Dharwal, R. Shanker, and M. Bhattacharya, "Automatic parameter setting of pulse coupled neural network for image segmentation," in *2016 International Conference on Communication and Signal Processing (ICCSP)*. IEEE, 2016, pp. 2157–2161.

[71] J. Hernández and W. Gómez, "Automatic tuning of the pulse-coupled neural network using differential evolution for image segmentation," in *Mexican Conference on Pattern Recognition*. Springer, 2016, pp. 157–166.

[72] X. Deng, Y. Ma *et al.*, "Pcnn model analysis and its automatic parameters determination in image segmentation and edge detection," *Chinese Journal of Electronics*, vol. 23, no. 1, pp. 97–103, 2014.

[73] C. Prieto, A. Rodriguez *et al.*, "A soft image edge detection approach based on the time matrix of a pcnn," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 463–469.

[74] X. Jin, D. Zhou, Q. Jiang, X. Chu, S. Yao, K. Li, and W. Zhou, "How to analyze the neurodynamic characteristics of pulse-coupled neural networks? a theoretical analysis and case study of intersecting cortical model," *IEEE Transactions on Cybernetics*, 2021.

[75] G. Halfacree, *The Official Raspberry Pi Beginner's Guide: How to Use Your New Computer*. Raspberry Pi Trading Limited, 2019.

[76] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 2921–2926.

Apéndice A

Publicaciones

El artículo titulado ***Aplicación del modelo de Intersección Cortical para segmentar caracteres en una placa vehicular*** fue publicado por el Congreso Internacional de Innovación Tecnológica y Computación (CIITEC 2020) en la revista electrónica Pistas educativas con ISSN 2448-847X, en el volumen 43, número 139, el cual se puede consultar en la siguiente dirección:
http://www.itcelaya.edu.mx/ojs/index.php/pistas/article/view/2548.

El segundo artículo: ***License plate detection based on Intersecting Cortical Model and entropy signatures*** fue publicado por el VIII Congreso Internacional de Robótica y Computación (CIRC 2021) como capítulo del libro Últimos Avances en Robótica y Computación con ISBN 978-607-98174-9-7.

El tercer artículo se titula ***Localización y reconocimiento de matrículas de automóviles utilizando redes neuronales*** y fue presentado en el III Congreso Internacional de Ciencias de la Computación (CONACIC 2021).