Methodology for an Automatic License Plate Recognition System using Convolutional Neural Networks for a Peruvian Case Study

D.M. Valdeos Acevedo, A.S. Vadillo Velazco, M.G.S. Pérez Paredes, and R.M. Arias Velásquez, *Senior Member, IEEE*

***Abstract*—In Peru, the number of vehicles increased in the last decade, therefore, the automatic control requires a long database with a specific license plate model. It should be used for registration, evaluation and information extraction associated to the cars to control access parking with a particular license plate characteristic, especially for government buildings, therefore, an automatic evaluation of the license plate should be more dynamic and effective than European systems due to quantity of characters, specific segmentation and additional information in the Peruvian plate. License plate recognition is a widely applied system in artificial vision for the automatic obtaining of car license plates. This research article seeks to design a Peruvian license plate recognition system to reduce the time of vehicle registration and it involves accurate recognition of the plate location and extraction. Image processing techniques are used using the Python programming language, together with the OpenCV library. In addition, with YoloV4 a neural network is trained to locate the area where the license plate is located to facilitate the application of an Optical Character Recognizer (OCR). Our findings are a new improvement and evaluation in the traditional license plate software, with a new Peruvian database, so it allowed extracting the registration information instantly with high accuracy evaluated with 200 to 1000 images; therefore, the new contribution is the improvement in the false positive values with an accuracy of 100%, rate of failure of 0% and the sensibility of 100% with a specificity of 100% (neural network trained with 1000 images); besides it is the database for the future works for Peruvian cars.**

***Index Terms*—Artificial vision, license plate recognition, optical Character, vehicle.**

I. Introducción

Actualmente, en el Perú, el mercado de automóviles ha aumentado, así como los centros comerciales y estacionamientos para poder guardar el vehículo y mantenerlo seguro durante una jornada de compras o paseos. El flujo de ingreso de los automóviles a estos estacionamientos se ve ralentizado por la fuerte demanda de parqueos seguros y el registro manual de la matrícula del vehículo, ocasionando congestión y estancamientos en autopistas, especialmente en Perú con su capital Lima, que cuenta con 31.2 millones de habitantes y 8.2 millones de vehículos [19].

"This work was supported in part by Universidad Tecnológica del Perú without Grant."

D.M. Valdeos Acevedo, A.S. Vadillo Valazco, M.G.S. Pérez Paredes and R.M. Arias Velásquez are with Universidad Tecnológica del Perú e-mail: ricardoariasvelasquez@hotmail.com

Debido a esta problemática, surge la necesidad de lograr un ingreso más dinámico a estacionamientos, con la mínima intervención de la mano del hombre para reducir tiempo, costo en recursos humanos y evitar congestiones. Por otro lado, con el avance de la tecnología se han logrado crear métodos para la detección de objetos en imágenes en tiempo real, incluso se puede obtener los caracteres presentes en una imagen tomada del vehículo para identificar la matrícula utilizando visión artificial y redes neuronales. Además, existen herramientas de programación y librerías, como Python y OpenCV, que facilitan la elaboración de un código para la ejecución de dichas tareas. Con lo expuesto anteriormente, se plantea la siguiente pregunta: ¿Se podrá realizar un sistema para el reconocimiento de placas vehiculares en Perú, que pueda ser implementado en el acceso de estacionamientos privados para controlar la entrada y salida de automóviles peruanos con una nueva base de datos?

II. Metodología

A. Metodología PICOC

En otro aspecto, se realizó la búsqueda en la base de datos IEEE, Science Direct y Scopus con la siguiente cadena de búsqueda sistemática: (License plate) AND (Neural Network) NOT (Trajectory) NOT (Speed) NOT (Tracking) NOT (Temperature) Los resultados fueron de once artículos en un intervalo de años entre 2019 y 2021, de las cuales se escogieron 6 artículos. Adicionalmente se realizó la búsqueda con el siguiente Search String: (License plate) AND ((Tensorflow) OR (Tesseract)) Entre los años 2017 y 2021 se encontraron diéz artículos, de los cuales se escogieron tres, dando un total de nueve artículos para la elaboración del estado del arte. En la tabla I, las técnicas de visión artificial, en la tendencia del 2019 fue la aplican módulos para detección y reconocimiento de la placa vehicular, siendo tres los módulos: Red de propuestas, Red de refinamiento y Red de salida. En el módulo de reconocimiento se emplea un tipo de algoritmo de extremo a extremo (End to End) para reconocer el texto de la placa [5]. Por otro lado, en revisando las tendencias del procesamiento de imagenes desde el 2017 al 2019, se aplican un limitante dentro de un rango de colores para que se muestra la placa vehicular con mayor intensidad. La imagen en blanco y negro muestra claramente los caracteres de la placa vehicular una vez realizado el tratamiento de la imagen [9]. Finalmente, como factores, las placas vehiculares tienen

TABLA I
TABLA DE EVALUACIÓN DE BÚSQUEDA SISTEMÁTICA

Evaluación	Aporte
Visión artificial utilizada para reconocimiento de placas vehiculares	En el diseño se realiza el entrenamiento de una red neuronal convolucional con una cantidad considerable de muestras, estas a su vez estan divididas en imágenes de muestra y prueba. Luego son llevadas a resoluciones bajas para un procesamiento mas rápido [1]. Después de la etapa de entrenamiento, se generan archivos con extensión XML que luego son convertidos a TFRecord que son interpretados por Tensor Flow [2].
Procesamiento de imágenes	El color en algunas placas vehiculares describe el tipo de vehículo, es por ello que se puede emplear el filtro Canny de OpenCV para poder clasificar un modelo de vehículo por el color de su matrícula [4].
Criterios o métodos que se utilizan para evaluar los resultados de la simulación del sistema.	La velocidad de detección de un sistema de detección de placas vehículares varía según el método. YOLOv3 es un tipo de red neuronal, el cual realiza el entrenamiento. El resultado del entrenamiento se ejecuta en un entorno de python y la velocidad de detección es de alrededor de 42ms [8].
Factores que garantizan un mayor porcentaje de éxito en el resultado del sistema de reconocimiento de placas vehiculares.	Se emplean grandes cantidades de muestras para un dataset mas robusto. Para una mejor clasificación se emplean 2049 imágenes para detectar placas vehiculares de una localidad (Taiwan) y 3977 de otra (China). Los resultados superan el 95% de precisión en la detección [3]. De igual manera, el método de identificación puede garantizar un mayor porcentaje de éxito en el reconocimiento de la placa. La segmentación de los caracteres e identificación de los patrones pixel a pixel, resulta en un porcentaje de éxito bastante alto en tres grupos de dataset (superior al 97%) [6]. .

similitud de una localidad a otra, pero eso no garantiza un porcentaje de éxito elevado. Si se emplea un dataset con placas de distintos tamaños y caractereisticas, se puede notar una eficiencia por debajo del 90% [7]

De acuerdo con la tabla de la búsqueda sistemática, para realizar un correcto reconocimiento de la placa vehicular, se deben realizar los pasos siguientes:

- Se debe tener una cantidad considerable de muestras para que la detección sea lo más precisa posible [3].
- Realizar el entrenamiento de diversos conjuntos de datos para placas vehiculares, que permitan establecer la cantidad de muestras necesarias para el reconocimiento de la placa vehicular.
- Priorizar la velocidad de detección de la placa vehicular para garantizar una mayor eficiencia del sistema [8].
- Emplear el motor de reconocimiento óptico de caracteres (OCR), una vez detectada la placa del vehículo [9].

III. ESTADO DEL ARTE

Un estudio realiza una comparación de algoritmos relacionados con la detección de placas vehiculares [1]. Se emplea un dataset de 300 imágenes a una resolución de 480x360, todas ellas tomadas por cámaras en una ciudad de Turquía. Uno de estos algoritmos se denomina Red Neuronal convolucional de región más rápida (Faster – R-CNN) y otro como Red neuronal totalmente convolucional basado en regiones (Based – Region FCN). En una primera prueba ambas obtienen una precisión cercana al 100% para la detección de placas vehiculares. En una segunda prueba se obtienen resultados similares destacando Faster R-CNN, obteniendo un porcentaje de precisión del 95%. El autor afirma que las CNN son más eficaces para la detección de objetos en imágenes de gran tamaño. Todas estas pruebas fueron realizadas con Tensorflow y Google colab, el cual te da la posibilidad de utilizar una tarjeta gráfica en la nube.

En la tabla II se muestra las limitaciones de las últimas técnicas de procesamiento de imágenes aplicadas a placas de rodaje. Asimismo, investigaciones desde el 2018, dan énfasis en el uso de tensores como metodología que permita la detección de placas vehiculares [2]. Un total de 480 muestras pasan por un proceso de etiquetado (selección del área de la placa vehícular). El dataset generado es un archivo XML que luego es convertido en uno de formato TFRecord que es interpretado por Tensorflow. Luego durante el proceso de entrenamiento y pruebas, realiza la utilización de los datos de entrenamiento, convolución y la capa de agrupación. Además el algoritmo propuesto realiza una serie de fases de entrenamiento y pruebas hasta alcanzar una alta tasa de precisión. En los resultados de la Tabla II, se muestra la evolución y limitaciones para cada capa de la red neuronal, asociada a la segmentación y dificultades en la doble fila de información; en la capa de convolución, para reconocer los bloques de datos de placas de la Unión Europea y finalmente las capas de activación con la de agrupación, ayudan para mejorar el resultado asociado a la calidad de imágenes de baja resolución de 63% a 78%. Esta ultima capa, de agrupación, contiene tantas neuronas como clases para predecir la ubicación de la placa. En la etapa final, se dibuja un cuadro en la imagen con la ubicación de la placa vehicular con el porcentaje de precisión de la misma.

Otro estudio realizado emplea técnicas de reconocimiento de placas vehiculares basadas en redes neuronales, además de la identificación de los caracteres [3]. El método consiste en dos detectores de objetos totalmente convolucionales de un solo estado. Esta técnica predice la ubicación de las "bouding boxes", las cuales contienen información de los objetos a detectar en fragmentos de la imagen, localiza su parte central, la orientación y por ultimo clasifica las placas y los caracteres. El trabajo realizado ,denominado ALPRNet, esta implementado en PyTorch y ResNet-50. Se emplean 2049 de Taiwan para la prueba denominada AOLP y 3977 imágenes de china denominadas PKU. En los resultados de AOLP todas sobrepasan el 95% de precisión tanto como para la detección y el reconocimiento de placas vehiculares. En los resultados de PKU donde solo se detectan las placas vehiculares y además está subdividido en 5 grupos (datasets), se obtiene una exactitud por encima del 99%.

Por otro lado, un Método basado en CNN-RNN para el reconocimiento de matrículas [4] adiciona una clasificación extra, matrículas privadas o públicas. Las placas vehiculares se clasifican por su color de fondo, utilizando detector de borde Canny, que separa la información del primer plano y el fondo. Luego de este proceso, las imágenes clasificadas

TABLA II
EVOLUCIÓN Y LIMITACIONES DE LA TEORÍA ACTUAL

Investigación	Limitaciones
Redes convolucionales con YoloV3 aplicando método de Jordanian (2020) [22]	Limitación en el proceso de segmentación con YoloV3 tiene 87% de precisión para placas con doble fila de información [22], debido al proceso de segmentación que origina errores al extraer la información de doble fila con tamaño distinto y con menos caracteres.
Segmentación binaria (2021) [20]	Si el carro es de unión europea, se segmenta en 7 bloques con 14 segmentos, no segmenta menos caracteres [20], en Perú se tiene 6 caracteres y 4 en la parte superior, a doble fila.
Reconocimiento de varias capas con Filtro Sobel y filtro de densidad para mejorar característica de placas (2021) [21]	Se puede utilizar solo para sistemas binarios, no es apropiado para doble línea de números como las placas peruanas.
Sighthound (2022) [23]	Permite una precisión del 63% [22], [23].
Redes antagónicas de doble generación para mejora de imagen y super resolución.	Incrementa la precisión para imágenes de baja resolución a 74.5% y 78% para imágenes de alta resolución.
Neighborhood pixel variance [24], el cual permite identificar la ubicación de los candidatos de texto en las imágenes, divididos en componentes R, G y B	Identifica los bloques de texto del video con recuperación 79.34%, precisión 71.63% y medida f 75.28%.

pasarán por CNN y RNN (Red Neuronal Recurrente), incluido un BLSTM (Bi-directional Long Short Term Memory), para el reconocimiento de los caracteres de la matrícula.

Otro método a mencionar es la Red Neuronal Multitarea para la detección y reconocimiento de matrículas [5]. Este trabajo tiene varias ventajas, puesto que es una red en cascada con estructura simple, de alta precisión y tiene bajo costo computacional. Utiliza una compilación de placas de China. Ellos utilizan una mejora de una MTCNN (Multi-Task Convolucional Neural Network), que es usada principalmente para la detección de puntos clave del rostro y reconocimiento del mismo.

Asimismo, en el 2021, se han realizado investigaciones que permiten mejorar el desempeño del procesamiento de imágenes usando YOLOv4-DenseNet, comparado con la versión de YOLOv3, comparando ambos resultados con la calificación F1 [25].

La arquitectura del MTCNN se divide en dos grandes módulos: módulo de detección y módulo de reconocimiento. El módulo de detección alberga tres redes. La primera, P-Net (Red de propuestas), obtiene el cuadro delimitador de vector de regresión y utiliza Supresión no máxima (NMS) para obtener la mejor detección del candidato de licencia; por esa razón tiene un umbral bajo para maximizar la toma de muestras positivas y recaudar mayor información de matrículas. La segunda, R-Net (Red de refinamiento), filtra los candidatos negativos que tienen malos efectos, optimizando la predicción de los candidatos seleccionados. La última, O-Net (Red de salida), ubica cuatro posiciones de esquina de la placa. En el módulo de reconocimiento de la licencia de placa, se utiliza CNN y CTC (Connectionist Temporal Classification) con un algoritmo End to end (de extremo a extremo) para emitir el texto de la placa sin segmentar. También se subdivide este segmento en tres capas: convolucionales, recurrentes y de transcripción. Aquí se obtienen resultados del 98% de precisión en reconocimiento.

Entre las técnicas de LPR se destaca particularmente la segmentación semántica. Para emplearla primero se debe tener la imagen que contenga la placa e identificar el color de fondo, y luego se aplica la técnica. Como resultado se obtiene el mapa semántico de la imagen de la matrícula, cuyos valores son la clase de caracter que contiene cada pixel, que finalmente son operados para obtener la secuencia de caracteres de la matricula. [6] Sin embargo, en los mapas de segmentación semántica resultantes no se pueden diferenciar los caracteres sucesivos del mismo tipo.Aplicando un módulo de recuento adicional a la técnica, la segmentación semántica obtuvo resultados del 99% em conjuntos de datos públicos de matrículas (AOLP y Media Lab).[6]

Cuando se utilizan imagenes que contienen autos con placas en escenarios distintos es necesario aplicar mas etapas e integrar una serie de métodos para lograr el eficiente reconocimiento de placas en vehículos en situaciones no controladas. Para esto se emplea redes neuronales convolucionales (CNN) y un algoritmo de Reconocimiento Optico de caracteres (OCR). El Método se divide en 3 partes: 1) Detección del vehículo empleando algoritmos libres de detección de objetos, 2) License Plate detection (LPR) donde se crea una CNN que identifica una placa en sus diferentes posiciones/ángulos dentro de una imagen para rectificarla de manera que se reubique plana y frontal al finalizar el proceso. y finalmente el 3) OCR que emplea YOLO. Los resultados de este método compuesto no logran superar la exactitud promedio de 89.33% en las pruebas a las que se le sometió. (Datasets con diferentes placas de diferentes países).[7]

Por otro lado, se utiliza la detección de secuencia de caracteres generalizada, para mejorar la exactitud de reconocimiento de placas vehiculares, cuando el banco de imágenes es procedente de varios países [8]. En esta última parte es muy interesante, puesto que el algoritmo detecta la secuencia de los caracteres sin importar cómo estén distribuidos, de esta forma usan un algoritmo universal que se puede aplicar a cualquier país.

Tambien se han realizados LPR con base en procesamiento de imágenes con OpenCV y Tesserat [9]. Sin embargo, R. R. Palekar et al. en su proyecto “Real Time License Plate Detection Using OpenCV and Tesseract” [9], la cual es muy práctica puesto que no se entrena una red, en cambio se utiliza filtros para procesar la imagen y con un detector de bordes obtener el area de la placa, que luego es procesada por

el OCR. La desventaja de esta técnica es que si los bordes de la placa son irregulares no se podra detectar la placa correctamente.

A. Diseño de la Investigación

Para el presente trabajo de investigación se aplicó un diseño cuasi-experimental. El diseño de este sistema de detección de placas vehiculares con redes neuronales utilizando el lenguaje de programación Python abarca el procesamiento de imágenes, el entrenamiento con redes neuronales y la simulación de todo el sistema. Para la simulación, se utiliza una imagen de la parte frontal o trasera de un automóvil. Esta imagen es procesada por el modelo entrenado, que devuelve o no la placa señalada en un rectángulo. Este modelo, que previamente fue generado por el entrenamiento con múltiples muestras de placas vehiculares, nos da como resultado el área de interés, es decir la zona donde se encuentra la placa vehicular. Luego se aplica una herramienta de reconocimiento óptico de caracteres (OCR), para obtener la matrícula.

B. Método y Enfoque de la Investigación

Se inicia de una teoría general y va a lo específico, considerando la contribución de varios autores con sus respectivas investigaciones relacionadas al tema, en donde se mejoró la detección de objetos y del reconocimiento de caractéres (número de placa) en imágenes. El desarrollo del sistema se realizó mediante el modelamiento de un algoritmo de código abierto en Python YolovV4 por las ventajas para el uso de librerías de visión artificial y modelamiento de redes neuronales que presenta dicho software.

El enfoque de investigación cuantitativa, permite medir la eficiencia en la clasificación mediante la exactitud, tasa de error, sensibilidad y especificidad. Por un lado, se deseó detectar la placa vehicular de un automóvil mediante la creación de un sistema con redes neuronales convolucionales que tienen como entradas para el entrenamiento imágenes en formato .jpg de la parte frontal o posterior del auto donde se encuentra la placa, luego esta imagen es procesada para detectar la zona donde se encuentra el objeto, para luego ser recortada e identificar el número de placa utilizando un reconocedor óptico de caracteres. Por otro lado, se desea medir los resultados para obtener el mejor rendimiento.

C. Detalle de las Placas en Perú

La presente investigación, se enfoca en optimizar el reconocimiento de placas especificas para un país, por las diferencias en otros países, las características principales de la nueva base de datos son las siguientes:

- La actualización de las placas en Perú, tiene un nuevo diseño obligatorio para todas las unidades.
  Algunas características suelen variar dependiendo de la modalidad o tipo de servicio que estos presten [18].
- Los conductores de vehículos presentan resistencia al cambio de placas; conservando modelos de placas que no están actualizados a la normativa actual, o el envejecimiento de las matrículas, las cuales pueden estar borrosas, con adornos lumínicos o fuera de su posición horizontal habitual; siendo una baja calidad en la placa, aunque la fotografía sea de buena calidad.
- Solucionar la dificultad del procesamiento para la identificación de los caractéres especiales de la placa Peruana, como en la figura 1, teniendo en cuenta la bandera, palabra Perú, código de seguridad, letras de la licencia.

D. Variable Dependiente e Independiente

En la Fig. 2, se identificaron cuatro variables: independientes, asociado a la claridad de la imágen a través de una escala de tres valores, asimismo, el número de muestras utilizados para el entrenamiento del modelo de detección, basado en pruebas de 200, 400, 600, 800 y 1000 muestras, siendo el caso de estudio evaluado con la finalidad de verificar el aprendizaje y precisión del modelo de clasificación y detección segun número de imágenes.
Seguido se considera el ángulo de visión basado en las coordenadas 3D de ubicación de la cámara, y los tipos de plataforma considerados para la aplicación de los algoritmos que son YoloV4 y Tesseract OCR. Las cuatro variables independientes, actúan directamente sobre el modelo de detección y reconocimiento de placas vehiculares peruanas.

E. Diagrama de Bloques para la Detección de las Placas

Por otro lado, se ha medido los resultados para obtener el mejor rendimiento, de acuerdo al diagrama de bloques Fig. 3.

Fig. 1. Detalles físicos de la placa de rodaje peruana. Ministerio de transportes y comunicaciones.

IV. Caso de Estudios

Para el entrenamiento se recopilaron una cantidad de 1000 imágenes de autos que contienen placas vehiculares, tanto de la vista frontal del vehículo como trasera, como la Fig. 4. Se realizaron cinco entrenamientos indepedientes, donde se utilizaron 1000 imágenes, 800 imágenes, 600 imágenes, 400 imágenes y 200 imágenes en cada uno de los entrenamientos para comparar la precisión de los resultados que se obtienen. Todos los entrenamientos tuvieron parametros iguales. Siendo cada imagen etiquetada, seleccionando el área de la placa vehicular de manera manual. Este proceso es único, ya que para cada etiquetado se genera un archivo de extension txt. Fig.

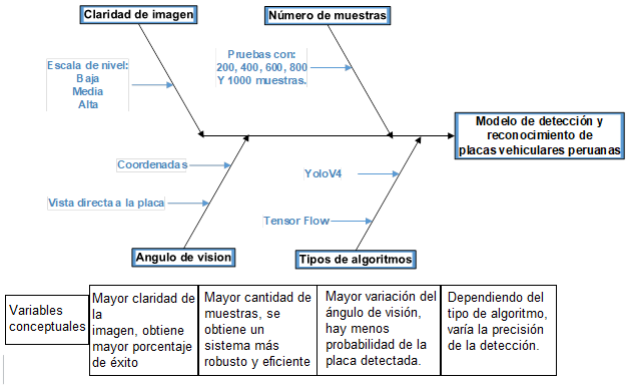

Fig. 2. Información de las variables dependiente e independientes.

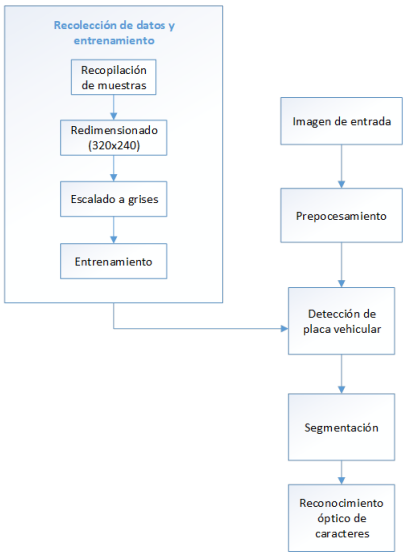

Fig. 3. Diagrama de bloques para el entrenamiento y recolección de datos.

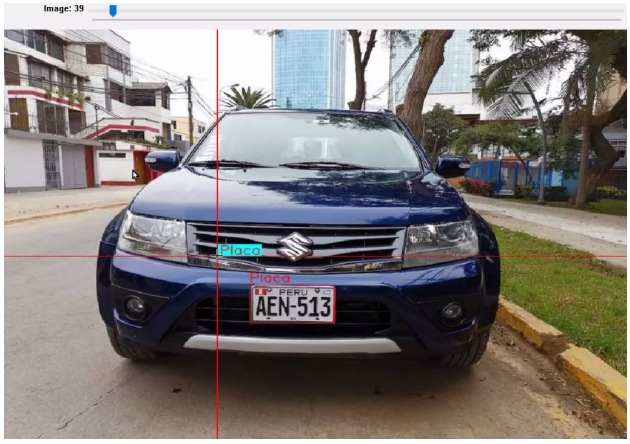

Fig. 4. Proceso de etiquetado de una imágen .

5 muestra como es almacenado la etiqueta “placa”, donde se encuentran cinco valores. El primero de ellos es la identificaión de índice de clase, en nuestro caso solo tenemos una clase, por lo que a la clase ‘placa’ se le asigna la ID de número 0. Los otros cuatro valores, son las coordenadas de la ubicación de la placa en la imagen, representando la altura y el ancho del limitador. Todas las imágenes y etiquetas se almacenan en una sola carpeta comprimida

```
0 0.4724358974358974 0.6068376068376068 0.12435897435897436 0.09914529914529914
```

Fig. 5. Etiqueta generada.

A. *Entrenamiento*

Para el proceso de entrenamiento se utiliza el entorno de programación de Google Collaboratory (Collab), donde se crea un notebook, utilizando el lenguaje de programación Python. Se debe tener una carpeta en Google Drive con nombre “yolov4”.

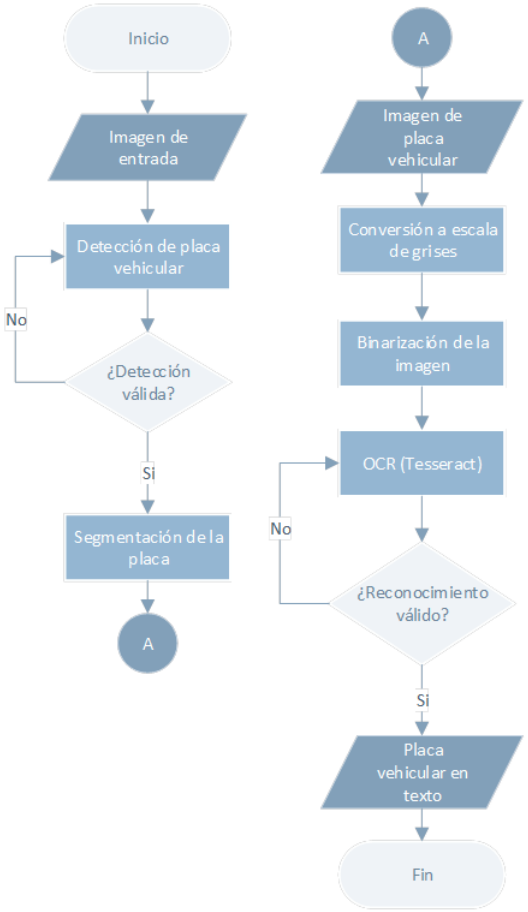

Fig. 6. Diagrama de flujo del proceso de entrenamiento.

La carpeta que contiene las imágenes y sus respectivos etiquetados son comprimidos en un archivo llamado ‘obj.zip’, para ser subida a la carpeta ‘yolov4’ de Drive. El siguiente paso es descargar el archivo yolov4-custom.cfg del directorio darknet/cfg, dicho archivo de configuraciones es perzonalizado con las siguientes caracteristicas:

- En el apartado training, batch=32 y subdivions=16.
- El ancho y alto (width y height) tendrán el valor de 416.
- El número de iteraciones serán un máximo de 6000 (max batches=6000).
- Los pasos (steps) serán el 80% y 90% del número máximo de iteraciones (steps=4800,5400).

- En la parte final del archivo, veremos tres capas [yolo] donde aparecerá el número de clases, en este caso classes=1.
- También se modifica el número de filtros de la capa convolucional previa a cada capa [yolo], filters=18, Fig. 7.

```
Archivo  Edición  Formato  Ver  Ayuda
[net]
# Testing
#batch=1
#subdivisions=1
# Training
batch=32
subdivisions=16
width=416
height=416
channels=3
momentum=0.949
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 6000
policy=steps
steps=4800,5400
scales=.1,.1
```

```
[convolutional]
size=1
stride=1
pad=1
filters=18
activation=linear


[yolo]
mask = 6,7,8
anchors = 12, 16, 19, 36, 40, 28,
classes=1
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
scale_x_y = 1.05
iou_thresh=0.213
cls_normalizer=1.0
iou_normalizer=0.07
iou_loss=ciou
nms_kind=greedynms
beta_nms=0.6
max_delta=5
```

Fig. 7. Perzonalización del archivo 'yolov4-custom.cfg'.

Se creó los archivos obj.data y obj.names, donde se almacenó el número de clases, los archivos train.txt y test.txt que se generan después de la creación de los archivos mencionados; finalmnte, se generó el archivo con el nombre de la clase 'Placa', respectivamente.

Paso siguiente, se ha creado el archivo process.py en la Fig. 8, el cual es un código en python con las instrucciones para crear los archivos train.txt y test.txt, donde estan las rutas del 90% de imágenes para el entrenamiento y 10% de las imágenes para las pruebas, respectivamente. En el notebook

```python
import glob, os

current_dir = os.path.dirname(os.path.abspath(__file__))

print(current_dir)

current_dir = 'data/obj'

percentage_test = 10;

file_train = open('data/train.txt', 'w')
file_test = open('data/test.txt', 'w')

counter = 1
index_test = round(100 / percentage_test)
for pathAndFilename in glob.iglob(os.path.join(current_dir, "*.jpg")):
    title, ext = os.path.splitext(os.path.basename(pathAndFilename))

    if counter == index_test:
        counter = 1
        file_test.write("data/obj" + "/" + title + '.jpg' + "\n")
    else:
        file_train.write("data/obj" + "/" + title + '.jpg' + "\n")
        counter = counter + 1
```

Fig. 8. Archivo 'process.py'.

de Google Collab, se habilita el uso de OpenCV; GPU, CUDNN, CUDNN HALF y LIBSO. Finalmente, se ejecuta el archivo process.py para la creación de los archivos train.txt y test.txt, estos archivos contienen 90% de imágenes para el entrenamiento y 10% de las imágenes para las pruebas, respectivamente. A continuacion, se descargan los pesos pre-entrenados YOLOv4, haciendo uso del aprendizaje por transferencia.

TABLA III
EVALUACIÓN DE RESULTADOS

	N=1000	N=800	N=600	N=400	N=200
VP	60	56	60	60	60
VN	40	39	40	40	40
FP	0	5	0	0	0
FN	0	0	0	0	0
Total	100	100	100	100	100
Exactitud	100%	95%	100%	100%	100%
Tasa de error	0%	5%	0%	0%	0%
Sensibilidad	100%	93%	100%	100%	100%
Especificidad	100%	98%	100%	100%	100%
Precisión	100%	92%	100%	100%	100%

V. DISCUSIÓN

Se realizaron las pruebas de la detección, dando como resultados la detección válida de las placas vehiculares, como se muestra en la Fig. 9; en la cual se puede comprobar la eficacia de esta red neuronal puesto que detecta correctamente una placa con diferentes ángulos de perspectiva.

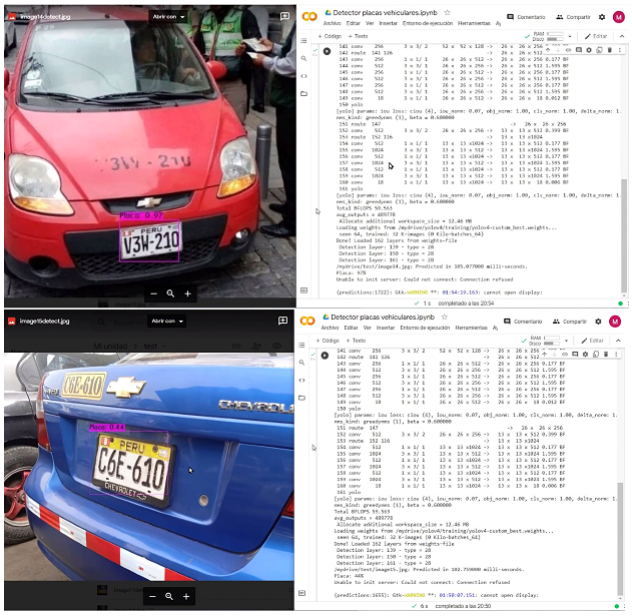

Fig. 9. Detección válida de una placa vehicular con calidad media.

Para evaluar la eficiencia del modelo entrenado en YoloV4, se utiliza 100 imágenes que no se encuentran en la base de datos de entrenamiento ni de prueba, donde 60 imágenes contienen una placa (imágenes positivas) y 40 imágenes no contienen placas (imágenes negativas). Además, se realiza la evaluación de los resultados verdaderos positivos (VP), verdaderos negativos (VN), falsos positivos (FP) y falsos negativos (FN). En la tabla II se muestra los resultados obtenidos en cada red, siendo N el número de imágenes. Se logra apreciar que el mayor error se presenta en la red con 800 imágenes, estabilizandose en las 1000 imáganes para el entrenamiento.

Para verificar el cálculo de exactitud en la ecuación 1.

$$Exactitud = \frac{VP + VN}{TOTAL} \tag{1}$$

TABLA IV
COMPARACIÓN CON OTRAS METODOLOGÍAS

Descripción	Exactitud	Tasa de Error	Sensibilidad	Especificidad
CNN y YoloV4 con N=1000	100%	0%	100%	100%
CNN y YoloV4 con N=800	95%	5%	93%	98%
YoloV3 y método Jordanian	87%	10%	88%	87%
Sighthoun con imágenes de alta resolución	78%	22%	80%	86%
Sighthoun con imágenes de baja resolución	63%	37%	81%	79%

El valor de tasa de error, se obtiene de la ecuación 2.

$$Tasadeerror = \frac{FP + FN}{TOTAL} \tag{2}$$

El valor de sensibilidad, se obtiene de la ecuación 3.

$$Sensibilidad = \frac{VP}{TotalPositivos} \tag{3}$$

La especificidad se obtiene de la ecuación 4.

$$Especificidad = \frac{VN}{TotalNegativos} \tag{4}$$

Finalmente, la precisión basado en el cálculo de los valores positivos, en la ecuación 5. Siendo el valor "TotalNegativos" la cantidad de imágenes clasificadas de forma incorrecta y "TotalPositivos" las imágenes clasificadas de forma correcta.

$$Press = \frac{VP}{Totalclasificadospositivos} \tag{5}$$

Las limitaciones principales se pueden dividir en 3 aspectos:

- Tiempo, el CMD de Windows llega a los 1500ms, comparado con el Google Colab, son 200ms; para un entrenamiento de 1000, la memoria RAM es estable con un mínimo de 16Gb. El uso del GPU mejora el tiempo de respuesta para el procesamiento de información.
- Cantidad mínima de imágenes para entrenamiento, la cual requiere como mínimo 1000 imágenes, para obtener la mejor precisión.
- Calidad de información, y dirección de las imágenes.

La presente investigación permite clasificar placas en Perú, de seis dígitos con segmentos de doble línea y con alta precisión. Por lo cual, este método mejora el uso de los algoritmos y entrenamientos actuales; superando las limitaciones actuales, como los sistemas de la India, Europa o Australia; asimismo, permite obtener buenos resultados con placas antiguas y con nuevas placas conservando los mismos resultados de evaluación; basado en la base de datos construida y su entrenamiento.

Como comparación importante, el motor de reconocimiento de caracteres "Tesseract" presenta inconvenientes al momento de identificar la placa vehicular. A la placa recortada después de la detección, no se le puede aplicar directamente el OCR porque genera como resultado una detección nula o errónea. A diferencia de un reconocimiento por redes neuronales, tesseract necesita de un procesamiento de imágenes previo para que la detección de los caracteres sea exitosa.

VI. CONCLUSIONES

Como se indica en la Tabla III, los resultados de exactitud y precisión fueron al 100%. Sin embargo, en el caso de la red cuyo entrenamiento fue con 800 imágenes la precision obtenida es de 92% y exactitud del 95%, puesto que en 5 imágenes presentaron falsos positivos (se considera falso positivo a la detección errónea de un área que no es una placa, a la detección parcial de la placa y a la doble detección de una misma placa); una mayor cantidad de imágenes tiene mejoras marginales llegando al máximo valor de 100%. En las imágenes resultantes que contienen placas y fueron detectadas correctamente por la red, se les aplico un preprocesamiento y OCR Tesseract, dando como resultado el reconocimiento efectivo de las matriculas de los autos. Asimismo, comparando versiones anteriores de YoloV3, métodos aplicados durante el 2021 como el Jodanian y sistemas como el Sightboun, la presente investigación conserva mejores resultados usando los mismos indicadores de calidad, con placas envejecidas y con placas de rodaje nuevas. De igual manera, debemos tener en cuenta las variables independientes para la generación de nuevas bases de datos, Fig. 2, se verifica que las cuatro variables influyen directamente en la probabilidad de detección de la placa, siendo los factores con mayor infuencia el ángulo de visión y claridad de la imagen, a partir de las 1000 imágenes en el entrenamiento. Por otro lado el número de muestras con bajo valor, incrementa el error por underfitting basado en valores de 800 imágenes o menos; emplear YoloV4 ofrece estándares de entrenamiento de 100% de precisión y sensibilidad al ser entrenados con imagenes mayores a 1000. Finalmente, la librería creada para el entrenamiento de 1000 muestras es suficiente para mejorar la clasificación de las placas y detectarlas con una tasa de precisión de valores falsos positivos de 100 %, tasa de falla de la detección de 0 % y la sensibilidad de 100%, con un valor de especificidad de 100%; siendo un aporte importante para el aprendizaje automático y futuros proyectos de extracción de información vehicular para mejorar el control de los sistemas de vigilancia, para los automóviles con placa peruana. Por tanto, los criterios utilizados para la generación de la base de datos, considerando las variables independientes, permitirá crear nuevos modelos y mejores bases de datos en un futuro. Los tiempos de detección son considerablemente menores utilizando el Google Colab, obteniendose un tiempo promedio de 200ms y en el CMD de Windows llega a los 1500ms. Este tiempo es inversamente proporcional a la potencia computacional del ordenador. En Google Colab, el tiempo se mantuvo estable debido al uso de la GPU en la nube. Finalmente, se recomienda utilizar CNN en lugar del "Tesseract" como motor de reconocimiento, dado

que incrementa un paso adicional y genera un punto de falla y de entropía mayor.

AGRADECIMENTOS

Los autores agradecen a la Universidad Tecnológica del Perú.

REFERENCES

[1] M. Peker, "Comparison of Tensorflow Object Detection Networks for Licence Plate Localization," 2019 1st Global Power, Energy and Communication Conference (GPECOM), 2019, pp. 101-105.
[2] I. Taufiq, et al., "Real-time Vehicle License Plate Detection by Using Convolutional Neural Network Algorithm with Tensorflow," 2018 2nd Borneo International Conference on Applied Mathematics and Engineering (BICAME), 2018, pp. 275-279.
[3] Q. Huang, Z. Cai and T. Lan, "A Single Neural Network for Mixed Style License Plate Detection and Recognition," in IEEE Access, vol. 9, pp. 21777-21785, 2021.
[4] P. Shivakumara, et al., "CNN-RNN based method for license plate recognition," in CAAI Transactions on Intelligence Technology, vol. 3, no. 3, pp. 169-175, 9 2018.
[5] W. Wang, et al., "A Light CNN for End-to-End Car License Plates Detection and Recognition," in IEEE Access, vol. 7, pp. 75-83, 2019.
[6] Jiafan Zhuang, et al., "Towards Human-Level License Plate Recognition" Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 306-32.
[7] Sergio Montazzolli Silva, Claudio Rosito Jung; "License Plate Detection and Recognition in Unconstrained Scenarios". Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 580-596.
[8] C. Henry, S. Y. Ahn and S. Lee, "Multinational License Plate Recognition Using Generalized Character Sequence Detection," in IEEE Access, vol. 8, pp. 35185-35199, 2020, doi: 10.1109/ACCESS.2020.2974973.
[9] R. R. Palekar, S. U. Parab, D. P. Parikh and V. N. Kamble, "Real time license plate detection using openCV and tesseract," 2017 International Conference on Communication and Signal Processing (ICCSP), 2017, pp. 2111-2115.
[10] B. Martín del Brío and C. Serrano Cinca, "Fundamentos de las redes neuronales artificiales hardware y software". Scire: Representación y organización del conocimiento, ISSN 1135-3716, Vol. 1, Nº 1, 1995, págs. 103-125, 1995
[11] J.V. Mejía Lara, R.M. Arias Velásquez, "Low-cost image analysis with convolutional neural network for herpes zoster", Biomedical Signal Processing and Control, 71, Part B, 2022, 103250, 1-8.
[12] Redes neuronales convolucionales. (s. f.). MathWorks - Creadores de MATLAB y Simulink - MATLAB y Simulink - MATLAB & Simulink. https://la.mathworks.com/discovery/convolutional-neural-network-matlab.html
[13] Arias Velásquez, R.M., Mejía Lara, J.V., Melgar, A., Converting data into knowledge for preventing failures in power transformers, Eng. Fail. Anal., 2019, 101, 215–229
[14] L. Rodriguez, et al., "Integración de funciones para Procesamiento Digital de Imágenes en Python", Universidad Tecnológica de Torreón
[15] OpenCV: Smoothing Images. (2021). OpenCV documentation index. $https$: $//docs.opencv.org/4.5.2/d4/d13/tutorial_py_filtering.html$
[16] OpenCV: Canny Edge Detection. (2021). OpenCV documentation index. $https$: $//docs.opencv.org/3.4/da/d22/tutorial_py_canny.html$
[17] "Github", 2019. [En línea]. Disponible en: tesseract-ocr.github.io/ . [Accedido: 08-sep-2021]
[18] "Asociación Automotriz del Perú", 2021. [En línea]. Disponible en: https://aap.org.pe/placas/tipos/ordinarias/tipo-de-placas/ . [Accedido: 01-sep-2021]
[19] Y. Romero, et al., "Temporal and spatial analysis of traffic – Related pollutant under the influence of the seasonality and meteorological variables over an urban city in Peru", Heliyon, 6, 6, 2020, e04029, 1-10.
[20] Karrar A. Kadhim, et al., "Automated high-security license plate recognition system", Materials Today: Proceedings, 2021, 1-4.
[21] Cong Zhang, Qi Wang, Xuelong Li, "V-LPDR: Towards a unified framework for license plate detection, tracking, and recognition in real-world traffic videos", Neurocomputing, Volume 449, 2021, 189-206.
[22] Salah Alghyaline, "Real-time Jordanian license plate recognition using deep learning", Journal of King Saud University - Computer and Information Sciences, 2020, 1-9.
[23] Sighthound [WWW Document], 2022. URL https://www.sighthound.com/products/cloud (accessed 1.17.22).
[24] Basavaraju, H. T., Manjunath Aradhya, V. N., & Guru, D. S. "Neighborhood Structure-Based Model for Multilingual Arbitrarily-Oriented Text Localization in Images/Videos". International Journal Of Interactive Multimedia And Artificial Intelligence, 7, 2021, 134-140
[25] Chen, S., et al. "Modified YOLOv4-DenseNet Algorithm for Detection of Ventricular Septal Defects in Ultrasound Images". International Journal Of Interactive Multimedia And Artificial Intelligence, 6(Special Issue on Current Trends in Intelligent Multimedia Processing Systems), 2021, 101-108.

Valdeos Acevedo, Dina Miluska Student of the Technological University of Perú. She is a student of Electronic Engineering. She has an extended and professional experience instructing fundamentals of electronics as well as carrying out projects to elementary and middle school students. She is part of the United Technologies for Kids-Invent (UTK-Invent), an organization in charge of spreading and teaching STEAM education in Peru and other countries. Dina had the opportunity in UTK-Invent to teach programing classes along with the collaboration of the University of Berkeley.

Vadillo Velazco, Alfredo Simón Student of the Technological University of Perú. Degree on Electronics from the IDAT Institute. He is a programming specialist. Currently, he is working on laboratory and environment equipment. He usually works on projects design based in micro controllers.

Marina Gabriela Pérez Paredes received the B.S. degree in electronic engineering from the Technological University of Peru, Lima, Peru, and M.S. and Ph.D. degrees in electrical engineering from the University of Campinas, Campinas, Brazil, in 2013 and 2018, respectively. From 2015 to 2016 was with the Hitachi Research Laboratory, Japan, working in its green mobility field. She is currently an Associate Professor at the University of Engineering and Technology (UTEC), and Technological University of Peru (UTP), Lima, Peru.

Ricardo Arias Velásquez IEEE Senior Member, he is operational efficiency specialist in ENEL, furthermore, he holds a PhD in Engineering from PUCP, MSc. degree in Engineering and three Bachelor degrees in Electrical engineering, Project engineering & Computer Science. Besides, PhD. Arias is IEEE PES PERU executive chairperson 2021-2022, and he is professor in UTP. Editor associate in IEEE Latin America Transactions. He authored 60 research articles published and indexed in SCOPUS.