Detección y lectura robusta de placas de automóviles utilizando modelos de máquinas de soporte vectorial, agrupamiento jerárquico de puntos de interés, estructuras de recuperación de información y media adaptativa del umbral

Alan Samuel Aguirre Salazar
Facultad de Ingeniería.
Universidad Panamericana
Jalisco, México
Emails: ASASauqui@protonmail.com / 0227221@up.edu.mx

Josué Olmos Hernández
Facultad de Ingeniería.
Universidad Panamericana
Jalisco, México
Email: 0226153@up.edu.mx

***Abstract*—En este paper, se presentará el uso de máquinas de soporte vectorial, agrupamiento jerárquico, detección de contornos utilizando recuperación externa y árboles de recuperación, y el algoritmo de la media adaptativa del umbral, para la detección y lectura de placas de automóviles. Nuestro método combina diversas técnicas de procesamiento de imágenes para una detección robusta de placas y hace uso de dos modelos de máquinas de soporte vectorial para la predicción de letras y números.**

Keywords—Detección, placas de automóviles, procesamiento de imágenes, agrupamiento jerárquico, contornos.

I. Introducción

El problema a resolver es sobre la detección de placas de automóviles y su posterior lectura e interpretación utilizando sólo técnicas de Machine Learning (ML) y procesamiento de imágenes, algo complicado debido a la falibilidad de algunas técnicas de Machine Learning comparadas con otras ramas más complejas, como lo son Deep Learning e Inteligencia Artificial. Diversas propuestas han aparecido haciendo uso de estas últimas dos ramas para la resolución de estos tipos de problemas, pero se optó por hacer una metodología basada en ML puro, la cual obtuvo resultados más que aceptables en cuanto predicción de caracteres de las placas de automóviles, siendo capaz de detectar y leer placas con poco contraste o que están algo saturadas de información visual.

Nosotros propusimos el uso de dos modelos de máquinas de soporte vectorial para la predicción de letras y números basados en probabilidad; la segmentación de imágenes utilizando árboles de recuperación y procesamiento de imágenes; la detección de placas mediante el uso de estructuras de recuperación y la media adaptativa del umbral; búsqueda de rectángulos a partir de contornos; y la utilización de agrupamiento jerárquico para la detección de contornos similares a través de puntos de interés extraídos de los rectángulos que encierran a dichos contornos.

II. Preparación de datasets y modelos para la predicción de letras y números

A. Datasets

Primeramente, se consiguieron dos datasets: uno de 26,400 imágenes de letras del abecedario inglés, y otro de 10,100 imágenes de números. Pero, para la aplicación de nuestra propuesta, se decidió aplicar un procesamiento de imagen a estos datasets, ya que cada imagen contenía demasiado espacio sobrante que afectaría la calidad de las predicciones de los modelos. El procesamiento de imagen para los datasets consistió en: convertir la imagen a escala de grises; posteriormente aplicar una umbralización de Otsu para poder binarizar la imagen y distinguir los objetos de mejor manera; buscar los contornos dentro de la imagen utilizando la recuperación externa para la detección del mayor bloque de píxeles (que en este caso es el carácter en específico de la imagen); buscar el mayor rectángulo de la imagen (en caso de que exista algún tipo de segmentación no deseada); hasta este punto se hizo todo esto para solamente identificar el carácter que nos importa dentro de la imagen y recortarlo para eliminar el espacio sobrante de la misma, una vez obtenida la nueva imagen recortada, se debe redimensionar a 28x28 píxeles; aplicar nuevamente escala de grises (ya que la imagen recortada se obtuvo directamente de la imagen sin procesamiento); aplicar el desenfoque gaussiano para la corrección de imperfecciones dentro de la imagen; y, finalmente, volver a aplicar una umbralización de Otsu para obtener solamente valores de negro o blanco en la matriz de píxeles de la imagen (0 ó 255).

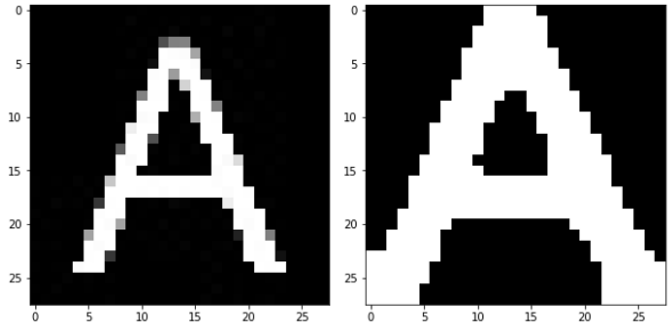

Fig. 1. Imagen de la letra ‘A’ después de pasar por el procesamiento de imagen.

Una vez que se le haya aplicado el procesamiento de imagen adecuado, se procedió a convertir la imagen a una sola dimensión, estos valores serán nuestra ‘x’ en el entrenamiento de nuestro modelo, y la ‘y’ será el código ASCII de dicho carácter, por ende, este código ASCII debe ser agregado al final del nuevo vector de una sola dimensión creado con anterioridad. Cada imagen se agrega a una lista, ya sea letra o número, aquí existe una división, debido a que habrá un modelo exclusivamente para letras y otro para números. Cada una de estas dos listas se aleatorizaron en posición para obtener muestras revueltas; cada lista se convirtió en un Dataframe y se exportó en un archivo de tipo “.csv”.

De esta forma, se han creado dos datasets, uno exclusivo para letras y otro para números.

Estos datasets serán de utilidad para el entrenamiento de nuestros modelos para la predicción de caracteres.

B. Modelos de máquinas de soporte vectorial

Se tenía en mente utilizar modelos de regresión logística multinomiales o modelos de máquinas de soporte vectorial para la predicción de caracteres, ya que éstos se dedican a clasificar elementos de diversas clases en un espacio vectorial, pero al final se optó por el uso de modelos de máquinas de soporte vectorial por obtener mejores resultados de accuracy y de R2. En el caso del modelo para las letras, se obtuvo un accuracy de 0.908 y un R2 de 0.844; y en el caso del modelo de números un accuracy de 0.949 y un R2 de 0.872, mientras que los resultados de los modelos de regresión logística mantuvieron valores inferiores a los anteriormente presentados. Los modelos de máquinas de soporte vectorial, a partir de estas métricas estadísticas, revelaron que sus predicciones son buenas y conservan una relación de “valores reales-predicciones” cercana a la forma de la función y=x.

Como ya se mencionó en la sección anterior, se crearon dos modelos, uno exclusivo para letras y otro para números, esto para obtener mejores discernimientos entre caracteres de su misma especie (letras o números), y poder hacer comparaciones entre caracteres similares y reducir el margen de error de predicción entre éstos. Para la creación de cada modelo, se segmentó la información de dichos datasets en ‘x’ y en ‘y’, donde ‘x’ son los valores binarizados de cada imagen, y ‘y’ es el valor ASCII que corresponde a dicha imagen. Estos valores se metieron al modelo de máquinas de soporte vectorial para entrenarlo y posteriormente se exportó el modelo para utilizarlo en la metodología para la lectura de placas de automóviles.

III. Detección de placas

Esta es la primera parte de la metodología para la captura y lectura de placas, la detección de placas; para esto se realizó la siguiente metodología.

Se necesita un frame o una imagen donde aparezca un automóvil que contenga algún tipo de placa visible. Parar extraer la placa del mismo, a la imagen se le aplicarán diversos procesamientos de imagen para obtener información importante que ayude a su identificación.

Primeramente, la imagen es convertida a escala de grises; seguido de la aplicación de un filtro bilateral, para la eliminación de ruido de la imagen y suavización de la misma; posterior a esto, se agregó un desenfoque gaussiano para la corrección de imperfecciones dentro de la imagen; y una vez aplicadas las correcciones pertinentes, se incluyó el algoritmo de la media adaptativa del umbral, la cual detecta información importante de la imagen en forma de bordes.

Ya con los bordes ubicados, se aplicó una búsqueda de contornos utilizando la técnica de recuperación en lista (muestra todos los contornos posibles); dichos contornos son ordenados de mayor a menor según su área y solamente se eligen los 25 contornos más grandes para eliminar los innecesarios.

Para cada contorno, se procede a sacar su área y se le encierra en un rectángulo para su detección. Al contorno original se le calcula su perímetro para aproximar la forma de su figura, aquí el objetivo es encontrar figuras que posean 4 vértices (sean rectángulos o cuadrados), por ende, si la figura aproximada contiene solamente 4 vértices, y si el área de dicha figura es menor o igual al 50% del área de la figura original, y si su relación de aspecto es mayor o igual a 1.7 y menor o igual a 5 (relación de aspecto que suelen tener las placas de automóviles), entonces se considerará que dicho rectángulo contiene una placa de automóvil, y se extraerá de la imagen original el sector de dicho rectángulo para obtener la placa y enmarcarla.

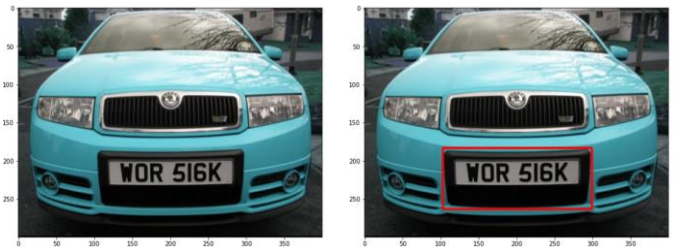

Fig. 2. Algoritmo de detección de placas enmarcando de rojo la placa del automóvil.

IV. Lectura de placas

Una vez se haya identificado la placa y se haya obtenido su imagen, se puede proceder a la lectura de la de misma, para averiguar cuál es su código de serie.

A. Importación de modelos

Para la lectura de placas es importante importar los modelos previamente creados para la predicción de letras y números.

B. Procesamiento de imagen

Nuevamente, el paso esencial en todo proceso de lectura de información de imágenes es el procesamiento de imagen. La imagen de la placa será convertida a escala de grises, se aplicará un desenfoque gaussiano para eliminar imperfecciones de dicha imagen y finalmente una umbralización de Otsu para poder binarizar la imagen y distinguir los objetos dentro de la misma de mejor manera.

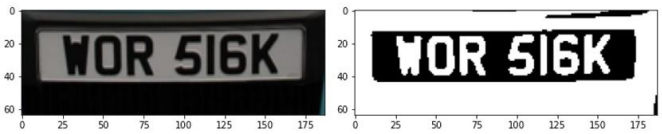

Fig. 3. Placa de automóvil después del procesamiento de imagen.

C. Obtención de componentes importantes

Teniendo la imagen procesada de manera idónea, se procederá a la búsqueda de contornos utilizando la técnica de árboles de recuperación para obtener todos los contornos de la imagen; y a dichos contornos se les agregará un rectángulo que les cubra de la mejor manera posible.

El problema ahora es que hay demasiados rectángulos por haber obtenido todos los rectángulos de la imagen, así que la siguiente tarea es reducir la cantidad de rectángulos de manera considerable, eliminando aquellos que no sea mayores o iguales en anchura al 2% de anchura de la imagen original y que no sean menores o iguales al 30% de anchura de la imagen original; y al mismo tiempo, deben cumplir ser mayores o iguales al 30% de la altura de la imagen original y en ser menores o iguales al 80% de la altura de la imagen original. De esta forma, se suelen eliminar hasta un 95% de los rectángulos obtenidos inicialmente (ya que suelen existir decenas de rectángulos del tamaño de un solo píxel) y sólo quedan aquellos que tienen una forma delgada en anchura y que tienen una altura alargada, básicamente, la figura que suelen tener las letras y números en una placa de automóviles.

Aquellos rectángulos que lograron cumplir con las especificaciones deseadas, se les extrae la imagen que contienen dentro de ellos desde la imagen original y se les redimensiona a 28x28 píxeles (el tamaño de las imágenes que aceptan nuestros modelos). Estas imágenes y rectángulos son guardados en variables para su posterior utilización en algunos procedimientos restantes.

Fig. 4. Rectángulos que cumplieron con las especificaciones.

D. Obtención de puntos de interés de los rectángulos

Este paso es de suma importancia para la identificación de los rectángulos correctos (por correctos nos referimos a aquellos rectángulos que contengan alguna letra o número y que pertenezcan al código de serie de la placa).

Se decidió que cada rectángulo poseyera 3 puntos que nos resultan de interés para la identificación de aquellos que sean idóneos, los cuales son: coordenada ‘y’ del punto de origen del rectángulo (el vértice del lado superior izquierdo); coordenada ‘y’ del vértice debajo del punto de origen (el vértice del lado inferior izquierdo del rectángulo); y, finalmente, la altura del rectángulo. Estos puntos resultan de mucho interés, ya que normalmente los números y letras que forman parte del código de serie de la placa suelen tener la misma altura y suelen estar alineados horizontalmente en línea recta, así que, aquellos que cumplan con estas semejanzas, tienen altas probabilidades de formar parte de los caracteres que conforman el código de serie.

Se guardaron los puntos de interés de cada rectángulo en una lista y se procedió a normalizar los valores, esto debido a que, dependiendo de la imagen, pueden haber más o menos píxeles, no siempre será igual; con esto se asegura que, independientemente de la cantidad de píxeles que existan, se tengan valores “relativos”.

E. Obtención de componentes idóneos

En esta parte se extraerán aquellas imágenes que formen parte del número de serie de la placa con ayuda de los puntos de interés. Los puntos de interés serán sometidos a un agrupamiento jerárquico utilizando el algoritmo de minimización de varianza “Ward”, y gracias a que los valores de los puntos de interés están normalizados, podemos determinar que dichos puntos siempre estarán variando en una distancia entre 0 y 1, donde 0.15 es la distancia adecuada donde habitualmente todos los miembros pertenecientes al código de serie de la placa forman un solo cluster en conjunto.

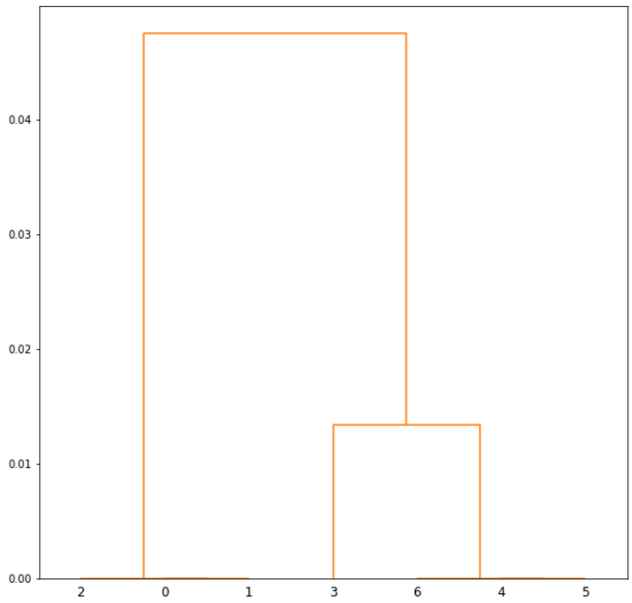

Fig. 5. Dendograma del agrupamiento jerárquico que demuestra que todos los puntos de interés de los rectángulos de cada imagen, en este ejemplo, se encuentran dentro de la distancia de 0.15.

Lo siguiente es buscar el cluster que contenga la mayor cantidad de elementos, ya que éste es el que cumple que todos sus puntos de interés sean relativamente similares y porque normalmente una placa de automóviles suele tener entre 4 y 9 caracteres, y con todas las restricciones puestas con anterioridad, las imágenes y rectángulos restantes han disminuido demasiado, por esto mismo, el grupo que resulte contener la mayor cantidad de elementos a esta distancia de separación, tiene la mayor probabilidad de ser el correcto, ya que es demasiada coincidencia que a una distancia menor o igual a 0.15, existan elementos muy similares que formen un grupo grande y que no se traten de los caracteres que conforman el código de serie de la placa.

Una vez obtenido el cluster mayor, elegimos las imágenes pertenecientes a dicho cluster y descartamos aquellas que no forman parte de él; de esta forma, podemos decir que ya tenemos, en cierta medida, los componentes adecuados.

Fig. 6. Imágenes dentro del mayor cluster que pertenecen al código de serie de la placa del vehículo.

F. Ordenamiento de imágenes.

El orden de las imágenes que tenemos no corresponde con el orden real en el cual se encuentran en la placa, suelen estar aleatorizados, por ello, para encontrar el orden adecuado, ordenamos de menor a mayor las imágenes según su coordenada en ‘x’ del punto de origen (vértice del lado superior izquierdo del rectángulo de la imagen), de esta sencilla manera, logramos poner las imágenes en el orden correcto.

Fig. 7. Imágenes ordenadas de forma correcta.

G. Procesamiento de imagen de los componentes

Nuevamente, se le debe aplicar un procesamiento de imagen a cada imagen de los componentes obtenidos (los cuales contienen un carácter de la placa). Se aplican los filtros de imagen ya usados con anterioridad: conversión a escala de grises; desenfoque gaussiano; y aquí lo diferente es que creamos dos imágenes basadas en la misma: a una le aplicamos la umbralización binaria de Otsu, y a otra lo mismo, pero siendo la binarización inversa; esto se propuso para hacer frente a cualquier tipo de color que utilice la placa. Ya en procesos posteriores se decidirá cuál binarización es la correcta.

Estas dos nuevas imágenes creadas en base a cada imagen que se contenga, se deberán guardar, ya que son de utilidad para el siguiente paso.

H. Predicciones de caracteres en base a la probabilidad

Ahora, se procederá a predecir qué carácter es más propenso a ser el que contenga la imagen. Este proceso de debe realizar para cada una de las dos imágenes obtenidas para cada imagen de los componentes, es decir, se debe aplicar por separado para las imágenes de binarización y para las imágenes de binarización inversa.

Primero, se debe redimensionar la matriz de la imagen en una sola dimensión (adelgazamiento), y utilizando los dos modelos que tenemos para la predicción de letras y números, les ingresamos dicho vector lineal a ambos modelos para obtener sus predicciones, tanto en letras como en números. De los resultados de probabilidad en ambos casos, se obtiene el valor de la mayor probabilidad que arrojó cada modelo, y según cuál probabilidad sea mayor, se discernirá si es una letra o un número. Pero en el caso de que la probabilidad de que sea un número sea mayor a la probabilidad de que sea una letra, se deberá tomar en cuenta que la probabilidad de que sea una letra no sea mayor o igual al 97%, ya que si sí lo es, quiere decir que tiene más posibilidades de que sea una letra, ya que el modelo que fue entrenado exclusivamente con letras contiene 26 clases (letras del abecedario inglés) contra las 10 clases que se utilizaron en el modelo de los números (números del 0 al 9), así que si la probabilidad de que sea una letra supera o es igual al 97%, ya es demasiada coincidencia que haya arrojado una probabilidad tan alta teniendo tantas clases en su conjunto vectorial. Y si, en caso contrario, no se cumple que la probabilidad de que sea una letra supere o sea igual al 97%, entonces, definitivamente se tomará que es un número.

De esta manera, se obtuvo qué carácter es el adecuado tanto para la imagen binarizada como para la imagen inversamente binarizada.

I. Decidir la correcta binarización

En este apartado, se decidirá cuál de los dos tipos de binarización es la idónea.

Hasta este punto, se deben tener 2N cantidad de imágenes, donde N es la cantidad de imágenes originales que conforman el código de serie de la placa; N están binarizadas, y la otra N de 2N están inversamente binarizadas. Se debe realizar una sumatoria de las probabilidades de cada imagen y checar cuál sumatoria es mayor; la sumatoria que resulte ser mayor deberá ser el método de binarización correcta, ya que fue la más cercana a tener predicciones acertadas, contrario a la otra que, al tener los valores invertidos, tendió a predecir caracteres inexistentes y de ahí sus bajas probabilidades.

Una vez haya sido determinada la binarización adecuada, desechamos las otras imágenes y datos, y solamente nos quedamos con las que deben ser.

J. Descarte de imágenes por poca probabilidad

Una vez se tengan los caracteres correctos de las imágenes pertenecientes al código de serie de la placa del automóvil, se debe realizar un chequeo de que la probabilidad de que sea dicho carácter sea mayor o igual al 40%, esto para cada carácter. Se optó por el 40% para dar un margen de error considerable, ya que se quiere que el modelo sea robusto, y, en estos modelos, el tener menos de 50% de que sea ese carácter no significa que sea una predicción mala, más bien indica que puede que sea ese carácter porque hay coincidencias suficientes, pero no termina de ser del todo claro.

Aquellos caracteres que no cumplan con el 40% de ser el carácter que dicen ser, serán descartados, ya que, probablemente, se trate de una imagen intrusa que cumplió con los demás requisitos de las restricciones que se impusieron.

Hasta aquí, se deben tener los que, posiblemente, sean los caracteres correctos que estén en el código de serie de la placa.

K. Formar la cadena

Finalmente, sólo basta con concatenar los caracteres resultantes y entregar una sola variable de tipo “string” al detector de placas para que lo coloque como identificador de la placa.

Fig. 8. Predicción de la placa completada.

V. Resultados

Como ya se ha mencionado previamente, los resultados de los modelos utilizados para la detección de letras y números fueron muy buenos, teniendo, en el caso del modelo para las letras, un accuracy de 0.908 y un R2 de 0.844; y, en el caso del modelo de números un accuracy de 0.949 y un R2 de 0.872.

Los resultados entregados por la metodología propuesta, en general, fueron muy buenos, siendo capaz de leer placas que tienen un contraste débil o que poseen diversos colores, incluso es capaz de hacer su trabajo en placas con un poco de saturación innecesaria.

En general, de un dataset exclusivo para comprobar la eficacia del algoritmo, compuesto de 148 placas de diferentes países, tipos, colores, fuentes de letra y ubicaciones de texto, logró acertar de forma perfecta 80 placas (sin ningún tipo de error), otras 43 placas tuvieron solamente un error y las demás restantes tuvieron más de un error. Para observar la cantidad de errores se utilizó el algoritmo de la distancia de Levenshtein para comparar dos “strings”. La desviación estándar de 1.312 nos indica que suele haber un error entre predicción, y el promedio de distancias nos arroja un 0.831, indicando lo mismo que la desviación estándar, que es posible que pueda haber un error por cada predicción entre los 4-9 caracteres que suele poseer una placa de automóvil.

Obviamente, los mejores resultados se dieron en placas del tipo europeo, japonés, chino, argentino, ruso, etcétera, debido a que estas placas poseen poca saturación de información y, normalmente, el contraste de los componentes es muy alto. En el caso de las placas de Estados Unidos, los resultados fueron diversos, ya que las placas en este país se pueden personalizar y suelen tener una excesiva saturación de componentes y colores, dificultando el reconocimiento del código de serie de la misma. Pero, aunque sean diversos, siempre que las letras se puedan discernir, se puede leer la información de manera correcta.

Por ende, estas cifras de éxito se pudieron elevar si se hubieran puesto solamente placas de China, Rusia, Japón, etcétera, y, por el contrario, disminuir si sólo se hubieran puesto placas complejas. Por esto mismo, se le pusieron placas variadas, para evitar cualquier tipo de “favoritismo” hacia un tipo de placa.

Sin embargo, la metodología presenta algunas confusiones entre caracteres similares, de ahí el problema que surgió en gran parte de las 43 placas que presentaron un solo error. Los caracteres que se suelen confundir son los siguientes: entre ‘G’ y ‘6’; entre ‘O’ y ‘0’; entre ‘B’ y ‘8’; entre ‘D’ y ‘O’; entre ‘I’ y ‘1’; entre ‘Z’ y ‘2’; etcétera. Esto puede disminuir la precisión del modelo, pero se comprende la confusión, ya que al momento del procesamiento de imágenes puede que algunas muestras hayan quedado similares y de ahí los errores entre estos caracteres con similitudes. Además de que, en sí, tienden a ser muy parecidos esto pares de caracteres.

VI. Conclusión

Esta metodología no es perfecta, pero presenta resultados decentes para la lectura de placas. Si la placa tiene poca

saturación de elementos y el contraste entre caracteres y el fondo es alto, la probabilidad de que haga su trabajo de manera correcta es muy alta, y si la placa tiene excesiva decoración y el contraste es pobre, obviamente le resultará más complicado.

Para sólo haber utilizado técnicas de Machine Learning, resultó ser un algoritmo robusto, que, sin importar el tipo de colores que tenga la placa, el orden o el país de la misma, es capaz de detectar el código de serie de la placa de manera decente.

Algunos puntos de mejora podrían ser la utilización de más modelos en lugar de sólo dos, es decir, tal vez hacer 10 modelos, donde en algunos estén algunas letras; en otros los números; en otros los caracteres similares donde suele haber más confusión, y encontrar la forma de validarlos para que arroje lo deseado.

También, otro punto de mejora, es intentar encontrar la forma de respetar los espacios entre los caracteres y detectar los guiones (‘-’), para obtener un “string” cien porciento fiel al código de serie de la placa.

Implementar un Haar Cascade para la detección de placas podría ser otro punto fuerte de mejora, ya que nuestra propuesta funciona, pero suele confundirse y en ocasiones no detecta las placas si algo les tapa o les deforma.

Y finalmente, el punto de mejora más importante es la optimización, ya que el algoritmo está poco optimizado y se suele ralentizar al utilizarlo en videos por la gran cantidad de procesos que necesita ejecutar por segundo. Una opción es intentar unificar los procesamientos de imagen en uno solo o en unos pocos; disminuir la cantidad de memoria que se utiliza, adecuarlo a Cython en lugar de utilizar Python, etcétera.