ISSN 1870-4069

# Semáforo inteligente utilizando procesamiento digital de imágenes controlado por medio del perceptrón de Rosenblatt

Mario Rosa Otero, Natalia Sánchez Patiño, David Tinoco Varela

Universidad Nacional Autónoma de México,
Facultad de Estudios Superiores Cuautitlán,
México

{malttz15, natyangelessp}@gmail.com,
dativa19@hotmail.com

**Resumen.** En este artículo se presenta el diseño e implementación de un semáforo inteligente. Para su funcionamiento, este semáforo capta imágenes en tiempo real, pre procesa tales imágenes y las convierte en vectores de información binaria que se ingresan a un perceptrón dentro de un microcontrolador que se encargará de definir el cambio de colores en un semáforo para vehículos y para peatones. La propuesta de semáforo inteligente busca que sea un sistema capaz de distinguir entre personas y vehículos para garantizar una interacción adecuada entre estos dos entes y afianzar la seguridad de los peatones, sin embargo, en los experimentos realizados, se han utilizado figuras humanas de plástico, así como autos de juguete a escala, pero detallados lo suficientemente bien para poder representar la forma real de estas entidades. Originalmente se planteó la posibilidad de sólo utilizar la binarización de una imagen para su clasificación dentro del perceptrón, sin embargo, esta situación identificaba las posiciones de los objetos a clasificar en vez de la distinción entre vehículo y peatón, por tal motivo se generó un sistema de identificación diferente, en el cual se extraen características de la imagen correspondientes a su área dentro de un entorno y estas características eran clasificadas por el perceptrón, dando una convergencia de gran robustez. Es posible obtener el código total del proyecto en[1].

**Palabras clave:** Perceptrón, inteligencia artificial, procesamiento de imágenes, redes neuronales.

# Intelligent Traffic Light Using Digital Image Processing Controlled by Rosenblatt Perceptron

**Abstract.** This article presents the design and implementation of an intelligent traffic light. For its operation, this traffic light captures images in real time, pre-processes such images and converts them into vectors of binary information that are entered into a perceptron within a microcontroller that will be in charge of defining the change of colors in a traffic light for vehicles and for pedestrians.

The intelligent traffic light proposal seeks to be a system capable of distinguishing between people and vehicles to guarantee an adequate interaction between these two entities and strengthen the safety of pedestrians, however, in the experiments carried out, plastic human figures have been used, as well as toy cars to scale, but detailed well enough to be able to represent the real form of these entities. Originally, the possibility of only using the binarization of an image for its classification within the perceptron was raised, however, this situation identified the positions of the objects to be classified instead of the distinction between vehicle and pedestrian, for this reason a different identification system was generated, in which characteristics of the image corresponding to its area within an environment are extracted and these characteristics were classified by the perceptron, giving a convergence of great robustness. It is possible to obtain the total code of the project[1].

**Keywords:** Perceptron, artificial intelligence, image processing, neural networks.

## 1. Introducción

Con el crecimiento de áreas como Inteligencia Artificial (IA), Big Data (BD), Internet de las Cosas (IoT), Industria 4.0, entre otras; se ha popularizado el uso de sistemas de ingeniería para control automático e inteligente de espacios comunes. Un sistema dentro de una ciudad, así como dentro de algunos edificios industriales y comerciales, que requiere de forma intrínseca un control (inteligente), es el sistema que componen los semáforos de paso.

Estos elementos están centrados en espacios donde es necesario tomar el control de un flujo, principalmente, vehicular y de esta manera eficientar la movilidad de un entorno, además de esto, mediante estos sistemas de control también se busca lograr una interacción segura entre dos entes que comparten las zonas de acción, como pueden ser, los peatones y los vehículos. Por los motivos mencionados, se vuelve importante e interesante, la generación de semáforos que reaccionen de forma inteligente a las circunstancias del entorno en los que se ven inmersos.

Por lo que, el esquema planteado en este artículo implica construir un semáforo independiente, que sea capaz de ceder el paso a humanos o vehículos motorizados, reaccionando de forma autónoma de acuerdo a las imágenes que el sistema adquiera en tiempo real, estas imágenes serán pre procesadas, evaluadas por el perceptrón y esto finalmente producirá una respuesta que se reflejara en el cambio de luz del semáforo.

Si se consigue trasladar el esquema planteado en este proyecto, a un sistema utilizable en calles reales, donde será necesario tomar en cuenta más parámetros de variación dados en las condiciones de dicha situación, podríamos optimizar los tiempos de paso de acuerdo a la afluencia entre carros y personas, mejorando así, la eficiencia de movilidad en una ciudad. Adicionalmente, con este semáforo se busca mejorar la seguridad de los peatones.

[1] https://github.com/NM-Labs/Semaforo-Inteligente-Con-Perceptron

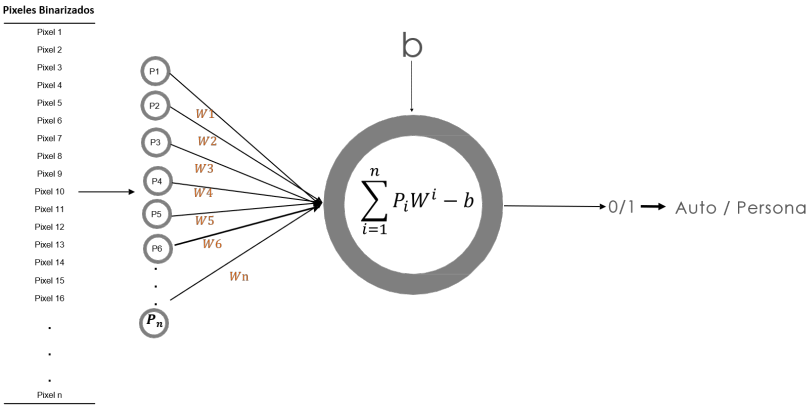


**Fig. 1.** Representación de un perceptrón para clasificación de imágenes.

## 2. Estado del arte

Normalmente los semáforos cambian de estado en sus luces de acuerdo a tiempos de activación y desactivación, al menos en México, sin embargo, estos esquemas no son de ayuda en eficientar el control vehicular y el control de paso peatonal ya que no consideran las condiciones de flujo. En una búsqueda de lograr un semáforo eficiente y que responda a las necesidades del entorno en donde se encuentra, se han ideado los semáforos autónomos, estos tratan de responder dinámicamente a las situaciones vehiculares y ciudadanas que se presentan en su entorno.

Existen diferentes, y muy variadas, propuestas para la generación de semáforos de control autónomos, algunas de ellas trabajando sólo mediante sensores y otras de ellas realizando análisis inteligente de los datos de entrada. En esta sección vamos a considerar las diferentes propuestas existentes para el diseño de los semáforos vehiculares, de forma general, y los semáforos peatonales, en forma particular.

Algunas de las propuestas de semáforos existentes están enfocadas al análisis de la densidad vehicular por medio de sensores IR y así lograr ranuras de tiempo dinámico [5]; otras propuestas consideran una conexión al IoT [9] o una conexión a internet por medio de redes de sensores inalámbricos [12].

Obviamente el uso de la IA ha jugado un papel importante en el desarrollo de estos nuevos esquemas de control, utilizando muy variados algoritmos y técnicas para lograr el objetivo, por ejemplo el uso de algoritmos de optimización adaptativa basado en el aprendizaje por refuerzo [16], el diseño de sistemas expertos para control de semáforo dinámico y automático [15], técnicas de aprendizaje profundo para lograr un ajuste dinámico de los tiempos de trafico real [14], utilización de controladores difusos para el control de semáforos en situaciones de emergencia [11] (hay que mencionar que existen una gran cantidad de propuestas basadas en lógica difusa, entre las que podemos observar [6, 8, 1]), y por supuesto, el uso de redes neuronales, como lo son el control basado en la teoría de la red neuronal de extensión (ENN) para encrucijadas [3] o el uso de redes neuronales para predecir y ajustar los tiempos de las señales en ambos lados de la carretera al mismo tiempo [10].

**Fig. 2.** Representación de coches y personas.

Hasta este punto, se ha hablado del diseño de diferentes tipos de semáforos vehiculares, sin embargo, un aspecto de principal interés dentro de este proyecto es el diseño de los semáforos que permiten el paso de peatones en un entorno vehicular.

Al igual que los semáforos vehiculares, este otro tipo también cuenta con una gran cantidad de propuestas dentro de la literatura científica, por ejemplo: el uso de algoritmos genéticos para mejorar el rendimiento del control de semáforos con paso de peatones [13], el diseño y entrenamiento de una red neuronal de memoria a corto plazo para predecir las intenciones de los peatones al cruzar la luz roja y evitar accidentes [17], la utilización del aprendizaje Q multi agente distribuido [7], así como el uso de redes neuronales convolucionales para el análisis de los datos del entorno [2].

Como se ha descrito en esta sección, existen diferentes técnicas para el control de semáforos, tanto vehiculares como peatonales, algunas de ellas con gran potencia de procesamiento y conexión a internet, sin embargo, la propuesta realizada en este artículo se enfoca al desarrollo de un semáforo que pueda ser implementado en entornos donde no se cuente con grandes cantidades de potencia de procesamiento y de conexión a internet, es decir, que pueda reaccionar de forma adecuada sin importar si existe conexión o no a internet.

## 3. Marco teórico: Perceptrón de Rosenblatt

Un perceptrón es la unidad básica de una red neuronal y consiste en una neurona con pesos sinápticos y un bias que son ajustables y puede ser usado para la clasificación en problemas linealmente separables. El algoritmo usado para ajustar los parámetros libres de esta red neuronal apareció por primera vez en un procedimiento de aprendizaje desarrollado por Rosenblatt (1958) para su modelo de perceptrón del cerebro.

Dicho perceptrón está limitado a realizar clasificación de patrones con sólo dos clases. Expandiendo la capa de salida del perceptrón para incluir más que una neurona, podemos realizar dicha clasificación con más de dos clases. Sin embargo, las clases tendrían que ser linealmente separables para que el perceptrón trabaje correctamente.

En la figura 1, podemos observar la representación básica del perceptrón. En la figura 1 los pesos sinápticos se denotan por $W_1, W_2, \ldots, W_n$ y la información que entra al perceptrón son los datos que representan al problema a clasificar, para fines de este proyecto, estos datos serán los píxeles de una imagen, representados como $P_1, P_2, \ldots, P_n$, y donde $b$ es el sesgo o umbral que se añade al proceso.

Después de realizar la operación de suma de la multiplicación entre pesos sinápticos y entradas, para posteriormente sumar el sesgo y obtener una salida, dicha salida será

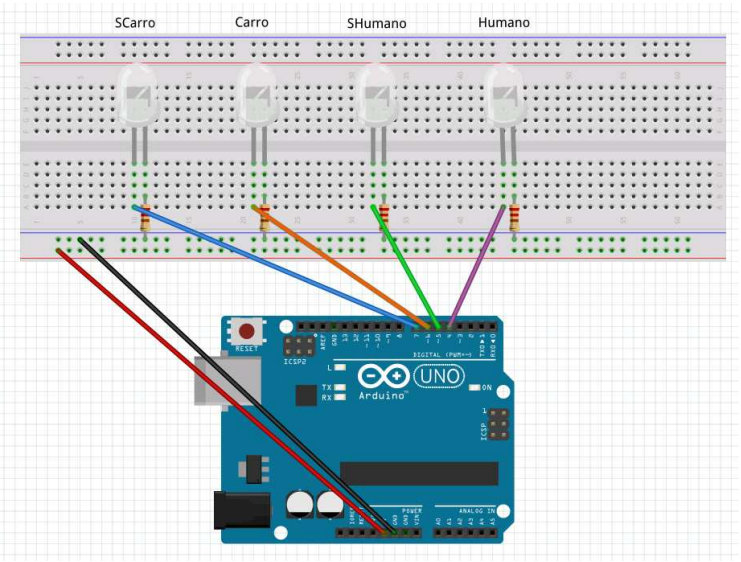


**Fig. 3.** Esquema de conexión del arduino.

evaluada por una función de activación, en este caso una función escalón de Heavisise, el cual establece un límite para determinar si la imagen contiene una determinada clase.

El aprendizaje de este algoritmo se basa en el cálculo del error obtenido de la diferencia entre la salida esperada $y_e$ con respecto a la salida obtenida $y_i$, entonces el error es calculado como: $e = y_e - y_i$.

A partir de este se van a actualizar los pesos sinápticos y el sesgo, intentando adaptarse a las entradas ingresadas e intentando buscar una configuración de los mismo con los que se puede obtener una salida esperada para todos los elementos, dentro de las operaciones de actualización interviene también un valor conocido como factor de aprendizaje, o tasa de aprendizaje denotado como $\lambda$, este factor regula la cantidad de corrección de error para no divergir en la respuesta al intentar corregir. [4] A continuación se muestran las ecuaciones utilizadas para actualización de sesgos y pesos:

$$W_n[t+1] = W_n[t] + \lambda(y_e - y_i)P_n, \tag{1}$$

$$b[t+1] = b[t] + \lambda(y_e - y_i)x. \tag{2}$$

## 4. Descripción del sistema

De forma general se da la descripción del sistema, para posteriormente ir detallando cada paso: Una cámara web enviará información de las imágenes tomadas hacía Matlab, en donde se realiza el procesamiento de las mismas, realizando esta vinculación por medio de la URL dada por IP webcam, la cual contendrá la imagen en tiempo real. La imagen pre procesada y vectorizada se guarda en archivos .cvs, los cuales se enviaran al perceptrón, y a su vez, el perceptrón regresará el resultado de la clasificación hacia Matlab, quién realizará el control de la tarjeta Arduino.

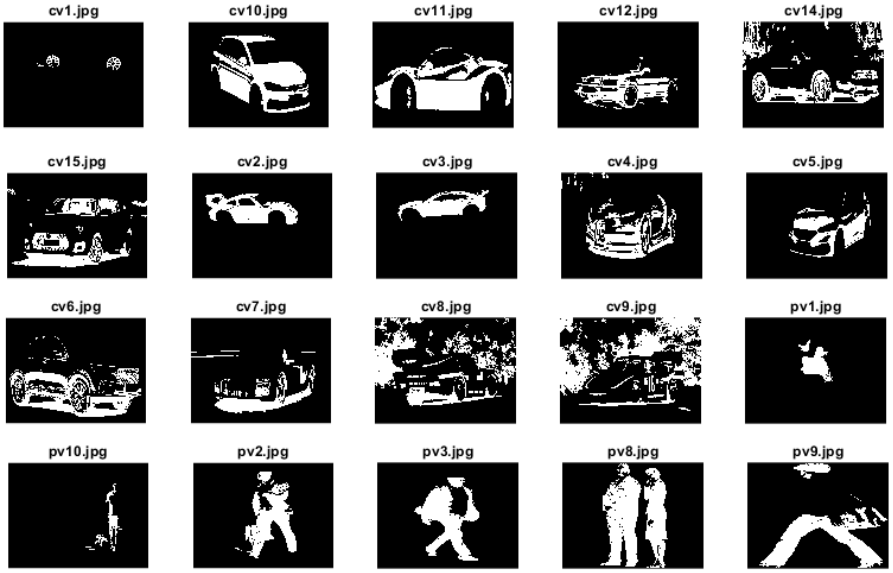


**Fig. 4.** Ejemplos de imágenes binarizadas que presentan ruido en estructura.

### 4.1. Adquisición y almacenamiento de datos

Para la obtención de un conjunto de datos representativo del problema, se procedió a colocar un fondo blanco y los elementos a utilizar en él, en esta forma, se tomaron fotografías a los objetos a clasificar, cuidando que no existan tantas variaciones de ángulo, distancia e iluminación entre las fotografías tomadas, para mantener controladas las condiciones del problema.

Siendo además importante tomar la misma cantidad de imágenes para el total de categorías a clasificar, en este caso misma cantidad de imágenes de autos y personas, estas imágenes se pueden ver en la figura 2. Del total de imágenes captadas de los elementos, se tomó un porcentaje entre el $15\,\% - 30\,\%$ que serán las muestras de validación, y el resto serán imágenes para entrenamiento, verificando que haya variedad de las mismas respecto a la variación de posición entre los elementos utilizados para ambas clases.

### 4.2. Pre-procesamiento de datos

Con el fin de simplificar la información contenida en una imagen y que pueda ser clasificable por un perceptrón se realizan métodos de pre-procesamiento digital de imágenes, en este procedimiento lo primero que se realiza es cambiar las imágenes a otro espacio de color, en este caso la imágenes cuando se ingresan están dadas en el espacio de color RGB, pero estas como primer paso sufren una transformación hacia el espacio CMYK, que tiene cuatro capas. Después de realizar la transformación se toma solo la capa *Y*, que resalta un rango de tonalidades en específico.

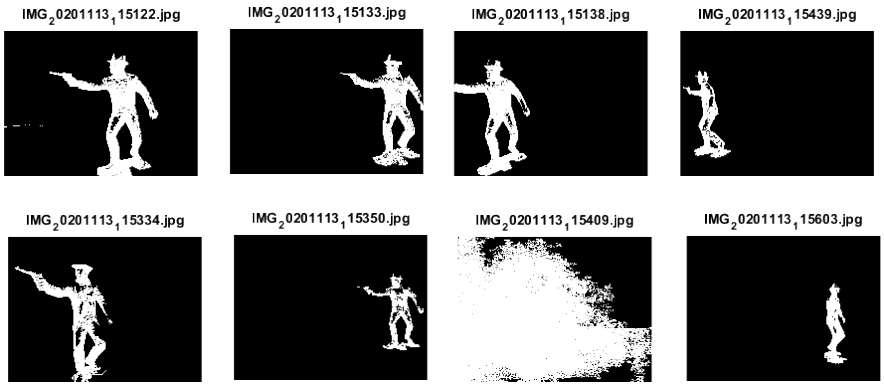


**Fig. 5.** Ejemplos de humanos binarizados.

Posteriormente dicha imagen en la capa *Y*, se pasa por una función de contraste que resalta aún más ciertos rangos de valor en la imagen, en esta caso se utiliza una función cúbica para realizar la mejora de contraste. Los píxeles $P_1, P_2, ..., P_n$ son utilizados para la operación:

$$\frac{P_n^3}{256^2}. \tag{3}$$

Esto dado que el rango de valores de píxeles están en una escala de 0 a 255. Posterior a esto, se pasa los pixeles de la imagen, ya con mejoras de contraste, por el algoritmo de Otsu, el cual busca en los valores de píxeles el valor de umbral que minimice la varianza $\sigma^2$ entre los mismos, el cual va a dar un umbral para realizar la binarización de la imagen.

El proceso del algoritmo requiere de obtener el histograma de la imagen, y obtener las probabilidades de cada nivel de intensidad de los píxeles, se establecen pesos iniciales $W_n(0)$ o probabilidades, y se obtiene así mismo las medias entre clases $\mu_i(0)$, con estos datos para cada nivel de intensidad se itera para obtener las varianzas entre clase, y después de tener todas las varianzas, toma la mayor varianza entre dos clases para establecer el umbral de binarización.

Posterior a este procedimiento, se utilizan algunos Momentos de Hu, que describen la manera en que se distribuyen los píxeles de un objeto sobre el plano de una imagen, estos momentos son invariantes a las transformaciones geométricas como traslación, rotación y escalamiento. Estos momentos se calculan para eliminar ruido en las imágenes binarizadas, dado que se discriminan objetos que hayan superado el umbral establecido por el algoritmo de Otsu, discriminándolos por algunas propiedades como el área.

Es necesario mencionar que se binarizan las imágenes para posteriormente convertirlas en vectores de entrada, vectores que contendrán únicamente dos valores, simplificando así, el proceso de clasificación del perceptrón. Durante el proceso de clasificación, se generó una respuesta errónea que será explicada más adelante, y debido a esta situación, al proceso ya mencionado, posteriormente se le agrego la extracción de características a las imágenes binarizadas, por medio de cajas de identificación, a estas cajas se le extrajeron datos como las distancias en los ejes X y Y, y el área de tales cajas.

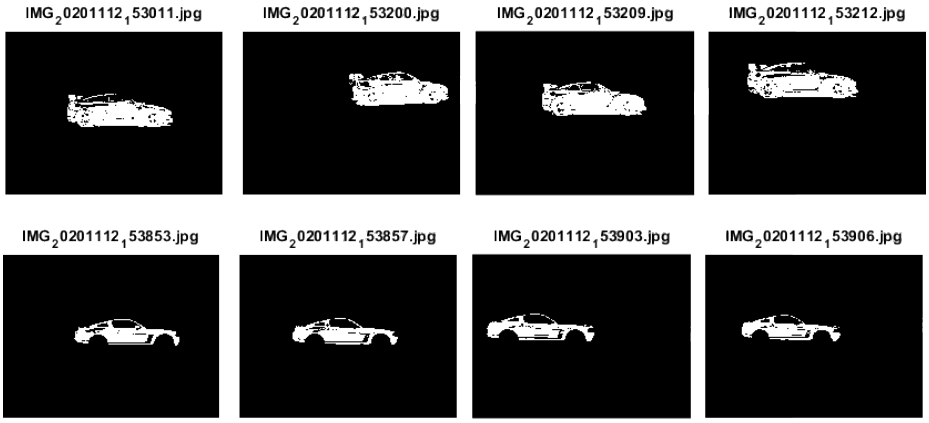


**Fig. 6.** Ejemplos de coches binarizados.

### 4.3. Entrenamiento de la red

Para poder ingresar las imágenes al perceptrón, estas tienen que ser re-dimensionadas, ya que se tienen matrices, que son convertidas a vectores. Realizado el pre-procesamiento a todas las imágenes de entrenamiento, se forma una matriz de características, es decir, se forma una matriz con todas las imágenes de entrenamiento ya re-dimensionadas como vectores, cada vector es una imagen binarizada, y se construye el vector de salidas esperadas conforme se acumularon las imágenes en la matriz de características.

Con estos cambios en los detalles de las imágenes, se puede realizar el entrenamiento. La construcción del perceptrón y su algoritmo de entrenamiento se divide en dos partes principales: La primera parte es con la que se realiza el entrenamiento, donde se lee la matriz de características de un archivo externo, y se realiza el mismo procedimiento para la lectura del vector de valores esperados, así como se realiza la inicialización de pesos y sesgo.

Cada una de las imágenes es pasada por el perceptrón, logrando así el entrenamiento de la neurona, cuando el entrenamiento es alcanzado, se guarda el vector de pesos en un archivo así como el sesgo, para que sean posteriormente utilizados al momento de realizar simulación o simplemente implantar el sistema; La segunda parte del código es una función para realizar la simulación del sistema en tiempo real.

### 4.4. Simulación del sistema

Una vez que se tuvieron todos los elementos integrados y ejecutándose de forma correcta, se inicia con la simulación del sistema, armando un circuito como el que se muestra en la figura 3, donde el LED con el nombre ”SHumano” representa el alto para peatones mientras que ”Humano” representa el siga. De manera similar pasa con los nombres de ”SCarro” y ”Carro”.

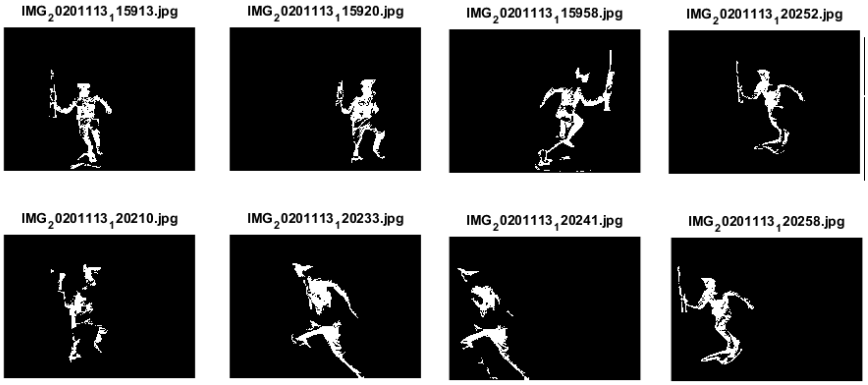


**Fig. 7.** Ejemplos de humanos binarizados con imagen limpia.

## 5. Resultados obtenidos: Binarización de imágenes

Al inicio se realizaron pruebas con imágenes y fotografías tomadas de internet donde se aplicó el binarizado a dichas imágenes, con esto se verificó que tan bien se realizaba la segmentación de los objetos de interés.

Como podemos ver en la figura 4 las imágenes al tener una gran cantidad de elementos pueden resultar muy ruidosas. Las mejores binarizaciones de imágenes fueron las personas que tenían un fondo blanco o transparente, así como autos en un entorno claro y sin demasiadas cosas alrededor. Debido al ruido que presentan las imágenes de la figura 4 , se decidió utilizar imágenes de un entorno controlado, donde las condiciones fueran óptimas para la binarización y detección de dichas imágenes.

### 5.1. Experimentos de clasificación

De manera formal, y para la clasificación de imágenes de autos y personas, se realizaron dos modelos experimentales. Las diferencias principales entre ambos, están dadas por el número de imágenes utilizadas para entrenamiento y validación, así como diferencias en la forma de clasificación.

**Modelo experimental 1.** Para la realización de este proyecto, se realizaron diferentes modelos experimentales hasta lograr una adecuada interpretación de la información obtenida, es decir, que la convergencia del perceptrón fuera adecuada al problema a resolver. Para el primer experimento de este modelo se utilizaron 48 imágenes en total de entrenamiento, contemplando 24 por clase, y 12 para validación. Las imágenes fueron tomadas mediante las cámaras de dos celulares distintos, ambos en un escenario blanco, como se puede ver en la figura 2.

Posteriormente se almacenaron, organizaron y binarizaron dichas imágenes, el resultado de binarización de algunas de ellas se puede ver en las figuras 5 y 6. En un segundo experimento, se utilizó un conjunto nuevo de imágenes más grande que el utilizado en el experimento 1 (100 imágenes de entrenamiento, 50 por clase, y 15 para validación), donde se cuidaron más las condiciones del entorno al momento de tomar las imágenes buscando un escenario más uniforme en texturas e iluminación.

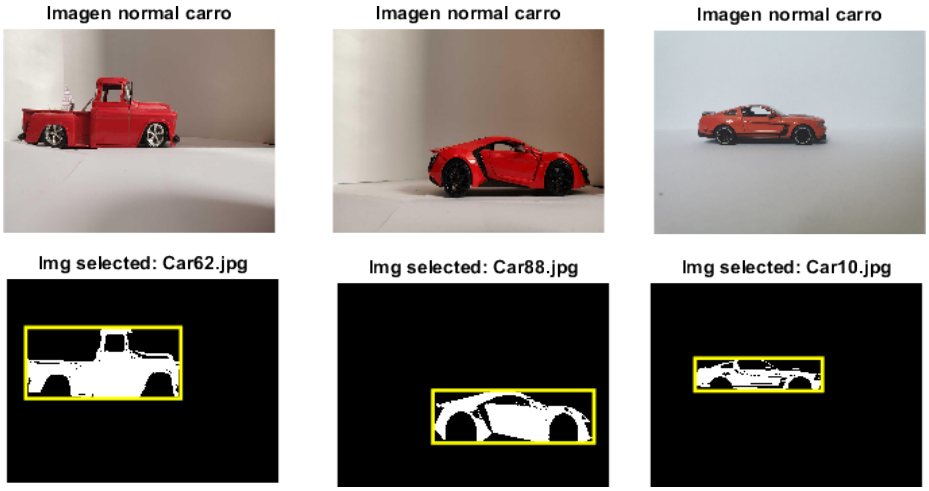


**Fig. 8.** Ejemplos de carros encerrados en una caja a la cual se le extraen carcterísticas tales como dimensiones y área.

Esto ayudó a que sobre todo las imágenes de humanos salieran más limpias al momento de realizar la binarización. Algunos ejemplos se pueden observar en la figura 7. Para este primer par de experimentos, observamos que en ciertas imágenes no se alcanza a segmentar el objeto de interés de forma correcta, esto debido a falta de control en condiciones del entorno como la iluminación y sombras creadas por terceros objetos, como se ve en la figura 5.

A pesar de esto, el perceptrón logró hacer una adaptación exitosa a los datos de entrenamiento, pero una clasificación menor en los datos de validación, presentando errores al momento de clasificar en tiempo real. Se realizaron 20 pruebas con la iluminación parecida a la que se tenía en los datos de entrenamiento, 10 imágenes tomadas en tiempo real por cada una de las cámaras utilizadas, cada una en su respectivo escenario.

Conforme se fueron analizando los resultados de las pruebas en tiempo real, tanto para el modelo obtenido con en el experimento uno, como en el experimento dos, con el sistema completamente armado se pudo notar que, al cambiar las condiciones de luz, los resultados se veían afectados. Con una buena iluminación, es decir midiendo de día y sin crear muchas sombras en el escenario, 100 % de las predicciones eran correctas, siempre y cuando se mantuvieran en una posición determinada para cada una de las figuras, izquierda para carros y derecha para personas.

Pero al hacer experimentos con variación de iluminación el rendimiento decaía a un: 60 %, cometiendo errores más frecuentemente cuando se trataba de coches. A pesar de los resultado de la clasificación, se observó un detalle sumamente importante en estos experimentos, y es que, lo que estaba clasificando el perceptrón eran las zonas donde había activaciones en lugar de los objetos que se buscaban detectar, es decir, no identificaba al objeto en sí mismo, identificaba la posición del objeto y en base a ella, determinaba la clase. Así, se observó que, si había objetos blancos en la parte derecha de la imagen binarizada, el resultado de la clasificación era ‘persona’.

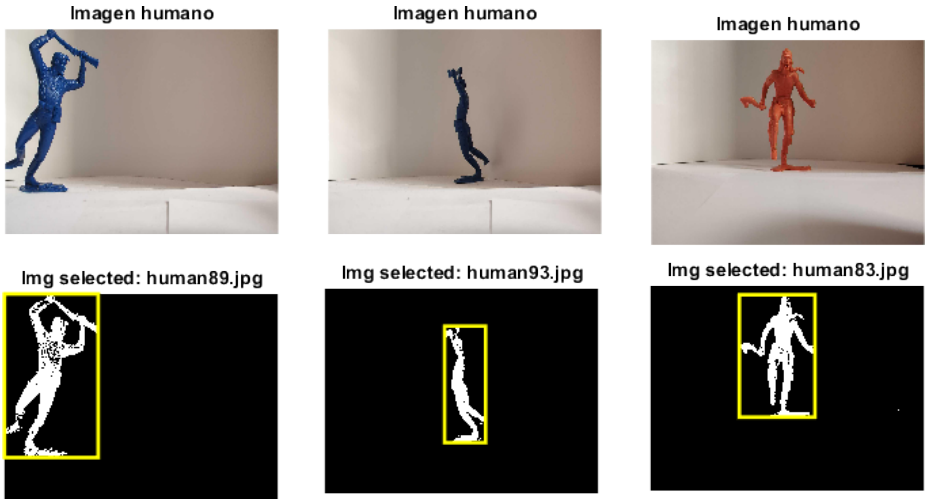


**Fig. 9.** Ejemplos de humanos encerrados en una caja a la cual se le extraen carcterísticas tales como dimensiones y área.

Si había objetos blancos a la izquierda de la imagen binarizada el resultado de la predicción era 'carro'. Lo cual hacía el sistema débil, y que clasificase mal si los objetos no se encontraban en la zona correcta.

**Modelo experimental 2.** Para solucionar el problema encontrado en los experimentos mencionados anteriormente, se buscó la forma en la que el perceptrón identificara características particulares de los objetos a clasificar, tratando de evitar que se centrara en la posición de tales objetos, que fue el caso anterior.

Para este fin, el siguiente paso fue hacer un entrenamiento, donde las imágenes tuvieran más elementos diferenciadores entre sí, es decir, hubiese más variabilidad respecto a las condiciones de iluminación, utilizando más modelos de objetos, variación en el ángulo en profundidad con que se colocaba el objeto respecto a la cámara, y también para poder retirar el sesgo de posición existente se tomaron imágenes con los objetos a diferentes distancias y ubicando el objeto en distintas zonas de la imagen, obteniendo así, variabilidad en la posición, y el tamaño.

Para ello, se tomaron de las imágenes previamente existentes, y se quitaron de ese conjunto las que tenían sombras muy marcadas y por ende, los resultados de la binarización eran muy inadecuados, con datos para entrenamiento. Adicionalmente, se agregaron las nuevas imágenes con mayor variabilidad. Dejando así, un total de 260 imágenes de entrenamiento, 130 de autos, y 130 de personas. Para validación se utilizaron un total de 90 imágenes, 45 de autos, y 45 de personas.

Con estas imágenes se relizó pre procesamiento mayor al ya realizado con la binarización, y en vez de pasarle la imagen aplanada al perceptrón, lo que se envió a la neurona fue el resultado de la extracción de características de cada una de las imágenes tomadas.

Como una primera parte en la identificación de objetos, se tomaron en cuenta 18 características, las cuales se enviaron al perceptrón: circularidad, área, solidez, longitud

**Tabla 1.** Cantidad de imágenes utilizadas para el entrenamiento y validación de los modelos experimentales.

| **Modelo experimental 1** | | | | | | |
|---|---|---|---|---|---|---|
| | **Experimento 1** | | | **Experimento 2** | | |
| | **Entrenamiento** | **Validación** | **Pruebas en tiempo real** | **Entrenamiento** | **Validación** | **Pruebas en tiempo real** |
| Autos | 24 | 12 | 10 | 50 | 15 | 10 |
| Personas | 24 | 12 | 10 | 50 | 15 | 10 |
| **Modelo experimental 2** | | | | | | |
| | **Experimento 1** | | | | | |
| | **Entrenamiento** | | **Validación** | **Pruebas en tiempo real** | | **Fuera de distribución** |
| Autos | 130 | | 45 | 30 | | 7 |
| Personas | 130 | | 45 | 30 | | 8 |

del eje mayor, longitud del eje menor, perímetro, excentricidad, diámetro equivalente, tasa de pixeles de la figura dentro de la caja entre los pixeles totales de la caja,longitud sobre el eje $x$ de la caja que encierra al área mayor, longitud sobre el eje $y$ de la caja que encierra al área mayor, primer momento de Hu, segundo momento de Hu, tercero momento de Hu, cuarto momento de Hu, quinto momento de Hu, sexto momento de Hu y séptimo momento de Hu. Al entrenar, el perceptrón no logró converger para dichas características en el número de épocas límite (100,000).

Debido a ello, se bajó la dificultad y nos quedamos únicamente con 3 características: Área y longitud del área en el eje $x$ y en el eje $y$, haciendo mención que estas características se extraen directamente por medio del procesamiento de imágenes, encapsulando los objetos y obteniendo las áreas de las cajas en las cuales se encapsulaban, estas cajas pueden verse en las figuras 8 y 9 (Es importante mencionar que al tener imágenes binarizadas, el encapsulamiento de objetos es óptimo, ya que identifica los espacios blancos del conjunto negro).

Con estas características logró converger en 18,162 épocas, en 1 hora 1 minuto y 54 segundos. El modelo logró ajustarse a todas las imágenes de entrenamiento, y todas las de validación, obteniendo un 100 % de rendimiento para ambos conjuntos. En el cuadro 1, es posible resumir la cantidad de imágenes utilizadas para el entrenamiento y ejecución de los dos modelos experimentales descritos.

## 6. Experimentación en tiempo real

Al realizar las pruebas en tiempo real con los pesos obtenidos para el 2do. modelo, se observó una exactitud del 100 %, cuando el total de pruebas realizadas fue de 30 para cada objeto, es decir 60 pruebas en tiempo real. Aun después de ello, se le pasaron al sistema imágenes que salían completamente de la distribución de las imágenes de entrenamiento, con cámaras muy diferentes, bajo condiciones de iluminación distinta, incluso iluminación artificial, y el modelo, para 15/15 de estas imágenes dio solución correcta.

En la figura 10, se puede apreciar el tipo de condiciones en donde se realizó la obtención de imágenes de entrenamiento, validación y prueba en tiempo real. Con lo que se deja en claro que el experimento tuvo condiciones controladas con la intención de que se pudiera hacer un buen binarizado y fuera clasificable para un perceptrón.

**Fig. 10.** Representación de coches y personas en escenario controlado.

## 7. Conlusiones

En este proyecto se realizó la implementación de un sistema controlador de un semáforo inteligente, capaz de detectar a una persona, o a un auto, y ceder el paso, conforme al objeto presentado frente a la cámara de detección, para esto se utilizaron dos lenguajes de programación distintos, los cuales brindan bondades, en uno de los casos, para manejo de datos, y pre-procesamiento de imágenes, y en el segundo caso, rapidez de cálculo para la toma de una decisión mediante el procesamiento de la información. Durante el desarrollo de este proceso, surgió una situación interesante al momento de realizar el entrenamiento de la neurona.

Al binarizar la imagen en el primer modelo experimental, el perceptrón logró identificar un patrón, sin embargo, este patrón no era el esperado ya que identificaba las posiciones con respecto a la línea peatonal, es decir, si el objeto estaba delante de la línea peatonal, lo identificaba como una persona; pero si el objeto estaba detrás de la línea peatonal, lo identificaba como un vehículo. Esta situación dio como resultado una falsa interpretación de las circunstancias a evaluar por la neurona.

Sin embargo, este error permitió la generación de una idea diferente para la solución del problema, en vez de considerar las posiciones de los objetos, se consideró la extracción de las características de los objetos, como el tamaño y forma de cada uno de ellos, lo que desemboco en que el perceptrón identificara correctamente una persona o un vehículo, es importante mencionar, que para la obtención de estas características no fueron necesario un cálculo o procesamiento complejo de las imágenes, ya que, según los experimentos, basta con distinguir las distancias que ocupan en el espacio, así como el área de cada objeto.

Las características relacionadas a los tamaños y formas de los objetos, fueron un punto primordial en la caracterización realizada por la neurona, estas características fueron extraídas gracias a las cajas en las que se encapsularon los objetos.

Bajo el problema planteado, un procesamiento sencillo por medio de una sola neurona, es adecuado, debido a que una persona siempre tendrá un área dentro del espacio más reducida que un vehículo, por lo que, para este proyecto, no fue necesario generar redes de extracción de características más complejas y de mayor tiempo de ejecución. En este artículo se pudo realizar experimentación con el sistema creado fuera del ámbito virtual, y comprobando su respectivo funcionamiento, con pruebas en tiempo real a través de la tarjeta de desarrollo Arduino.

## 8. Trabajo futuro

Uno de los fines de este proyecto, es la búsqueda de sistemas inteligentes, en este caso un semafóro, que puedan ser utilizados en entornos en los que no se tenga una capacidad alta de procesamiento computacional, por lo que se buscará que el siguiente paso sea la implementación de todo el proceso en una tarjeta de desarrollo de bajo costo, que pueda ejecutar la propuesta de forma autónoma e independiente, tal como una tarjeta Raspberry Pi.

**Agradecimientos.** Este proyecto fue apoyado por los proyectos PAPIIT IN105219, PAPIME PE100221 y PIAPI 2053 de la UNAM y de la FES-C.

## Referencias

1. Arora, M., Banga, V. K.: Real time traffic light control system using morphological edge detection and fuzzy logic. In: Proceedings of the 2nd International Conference on Electrical, Electronics and Civil Engineering, pp. 28–29 (2012)
2. Ash, R., Ofri, D., Brokman, J., Friedman, I., Moshe, Y.: Real-time pedestrian traffic light detection. In: Proceedings of the IEEE International Conference on the Science of Electrical Engineering in Israel, pp. 1–5 (2018) doi: 10.1109/ ICSEE.2018.8646287
3. Chao, K. H., Lee, R. H., Wang, M. H.: An intelligent traffic light control based on extension neural network. In: Proceedings of the International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, pp. 17–24 (2008) doi: 10.1007/978-3-540-85563-7_8
4. Gallant, S. I.: Perceptron-based learning algorithms. In: Proceedings of the IEEE Transactions on Neural Networks, vol. 50, no. 2, pp. 179–191 (1990) doi: 10.1109/72.80230
5. Ghazal, B., ElKhatib, K., Chahine, K., Kherfan, M.: Smart traffic light control system. In: Proceedings of the Third International Conference on Electrical, Electronics, Computer Engineering and their Applications, pp. 140–145 (2016) doi: 10.1109/EECEA.2016.7470780
6. Kulkarni, G. H., Waingankar, P. G.: Fuzzy logic based traffic light controller. In: Proceedings of the International Conference on Industrial and Information Systems, pp. 107–110 (2007)
7. Liu, Y., Liu, L., Chen, W. P.: Intelligent traffic light control using distributed multi-agent Q learning. In: Proceedings of the 20th International Conference on Intelligent Transportation Systems, pp. 1–8 (2017) doi: 10.1109/ITSC.2017.8317730

8. Mehan, S.: Introduction of traffic light controller with fuzzy control system. International Journal of Electronics & Communication Technology, vol. 2, no. 3, pp. 119–122 (2011)
9. Miz, V., Hahanov, V.: Smart traffic light in terms of the cognitive road traffic management system based on the internet of things. In: Proceedings of IEEE East-West Design and Test Symposium, pp. 1–5 (2014) doi: 10.1109/EWDTS. 2014.7027102
10. Rizwan, J. M., Krishnan, P. N., Karthikeyan, R., Kumar, S. R.: Multi layer perception type artificial neural network based traffic control. Indian Journal of Science and Technology, vol. 9, no. 5, pp. 1–6 (2016) doi: 10.17485/ijst/ 2016/v9i5/87267
11. Salehi, M., Sepahvand, I., Yarahmadi, M.: TLCSBFL: A traffic lights control system based on fuzzy logic. International Journal of u-and e-Service, Science and Technology, vol. 7, no. 3, pp. 27–34 (2014) doi: 10.14257/ijunesst.2014.7.3.03
12. Tubaishat, M., Shang, Y., Shi, H.: Adaptive traffic light control with wireless sensor networks. In: Proceedings of the 4th IEEE Consumer Communications and Networking Conference, pp. 187–191 (2007) doi: 10.1109/CCNC.2007.44
13. Turky, A. M., Ahmad, M. S., Yusoff, M. Z., Hammad, B. T.: Using genetic algorithm for traffic light control system with a pedestrian crossing. In: Proceedings of the International Conference on Rough Sets and Knowledge Technology, vol. 5589, pp. 512–519 (2009) doi: 10.1007/978-3-642-02962-2_65
14. Wei, H., Zheng, G., Yao, H., Li, Z.: IntelliLight: A reinforcement learning approach for intelligent traffic light control. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2496–2505 (2018) doi: 10.1145/3219819.3220096
15. Wen, W.: A dynamic and automatic traffic light control expert system for solving the road congestion problem. Expert Systems with Applications, vol. 34, no. 4, pp. 2370–2381 (2008) doi: 10.1016/j.eswa.2007.03.007
16. Wiering, M. A., Veenen, J., Vreeken, J., Koopman, A: Intelligent traffic light control. Utrecht University: Information and Computing Sciences (2004)
17. Zhang, S., Abdel-Aty, M., Yuan, J., Li, P.: Prediction of pedestrian crossing intentions at intersections based on long short-term memory recurrent neural network. Transportation research record, vol. 2674, no. 4, pp. 57–65 (2020) doi: 10.1177/0361198120912422