Grupo 9: Reconocimiento del estado de los semáforos de vías públicas

Nombre de los integrantes del proyecto:

Braganza Zorilla Samuel Mauricio
Godoy Saavedra Axel Jeremy
Pinargote Figueroa Jorge Anibal

Problema

La problemática encontrada parte de inconvenientes presentados en la vía pública, en muchos casos es muy complicado controlar el tráfico lo cual puede ocasionar accidentes, ya que en la mayoría de las vías los únicos agentes reguladores del tránsito son los semáforos, los cuales en muchas ocasiones pueden ser obviados por muchos de los conductores causando accidentes leves o inclusive encadenar accidentes a gran escala, se puede inferir que en múltiples ocasiones los conductores pierden atención por algún agente externo aunque se espera que ellos estén totalmente concentrados en manejar, pero en muchas ocasiones eso se puede complicar entonces sería necesario tener una señalización que permita identificar el estado de semáforos cercanos por los cuales se pretende transitar.

Otro problema que se puede identificar claramente sobre la identificación de semáforos en la vía pública al conducir son aquellas personas que no reconocen los colores correctamente lo cual se conoce como daltonismo, dichas personas pueden llegar a tener serios problemas si es que poseen una licencia y dicha enfermedad comienza a desarrollarse en ellos podría convertirse en un gran inconveniente en un futuro cercano.

En base a las problemáticas planteadas previamente podemos notar que todo recae sobre problemas que atentan contra la integridad de las personas, entonces se podría crear una interfaz de una aplicación web para validar el funcionamiento de la IA que permita reconocer los colores de los semáforos como los sistemas presentados en [1], [2], y que emita señales diferentes dependiendo del estado identificado, la inteligencia artificial sería el componente clave ya que permitiría analizar el entorno en busca de semáforos a través de una cámara e informaría sobre el estado de ellos en caso de encontrar alguno, se podría determinar que las técnicas de inteligencia a aplicar trabajan como agentes de reconocimiento [3], [6], [12].

Objetivo

Reconocer los colores de semáforos en la vía pública mediante conceptos de inteligencia artificial para aportar positivamente a la circulación de vehículos

Modelo de análisis y definición del problema

Se representa el problema mediante un diagrama de actividades, donde se describe una situación común que pasan los conductores en general, cuando se observa el semáforo es determinante reconocer su estado actual para poder proceder de manera correcta o acorde, sin embargo muchas personas pueden tener problemas de visión y podrían causar accidentes. Además se presenta. Además se presentan el diagrama de casos de uso con los actores y sus posibles acciones en la problemática planteada, el diagrama de clases para representar a los elementos del problema como objetos y sus respectivas relaciones, por último se presenta el diagrama de secuencia para denotar cómo interactuarían los conductores en una situación real al tener que identificar el estado del semáforo.

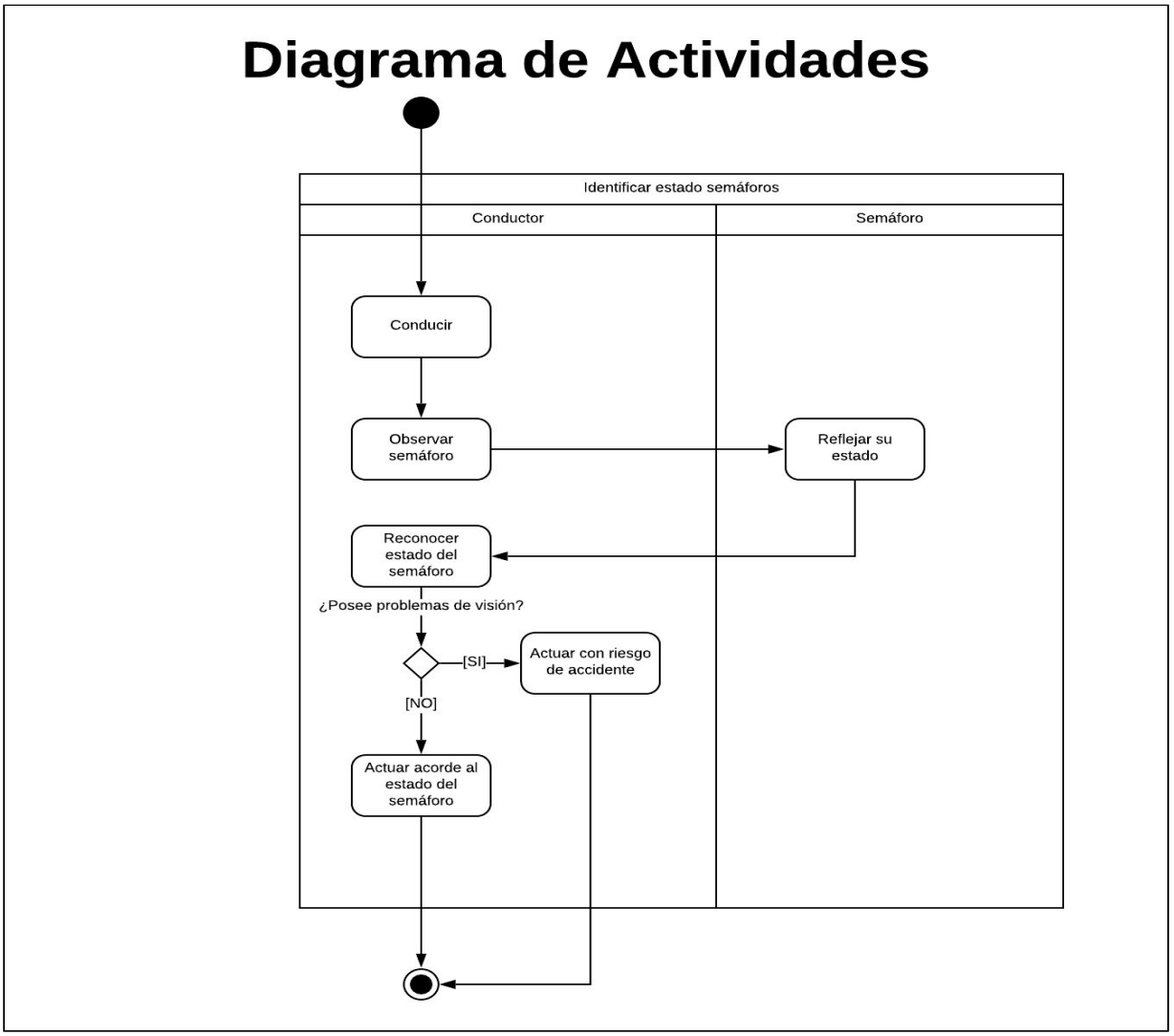

Figura 1: Diagrama de actividades problemática

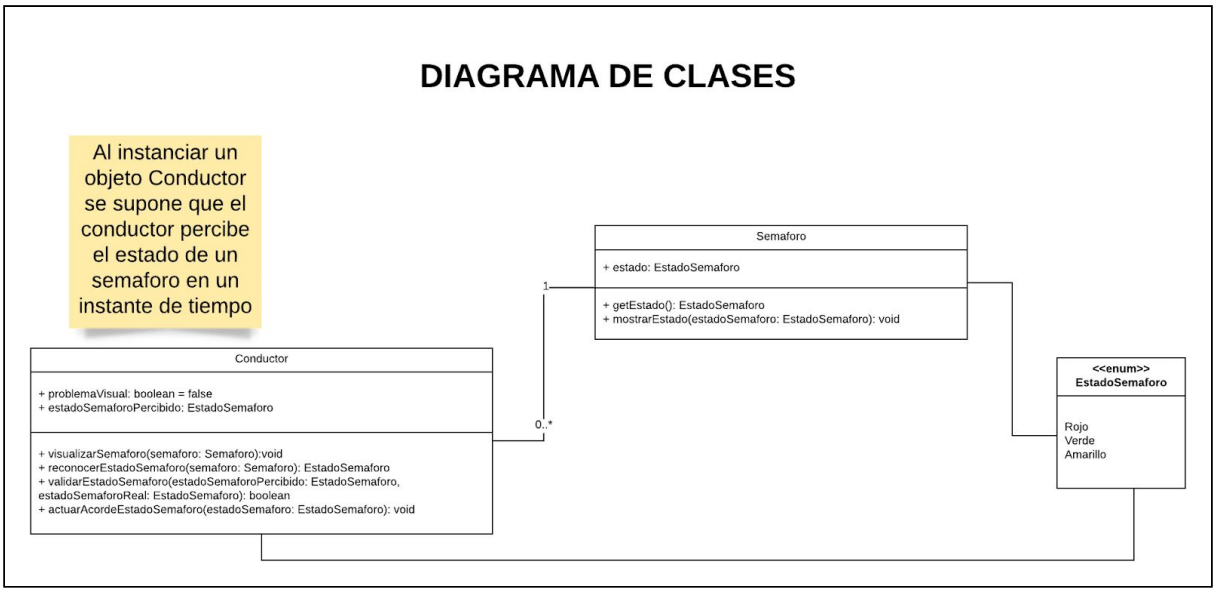

Figura 2: Diagrama de clases problemática

Figura 3: Diagrama de casos de uso problemática

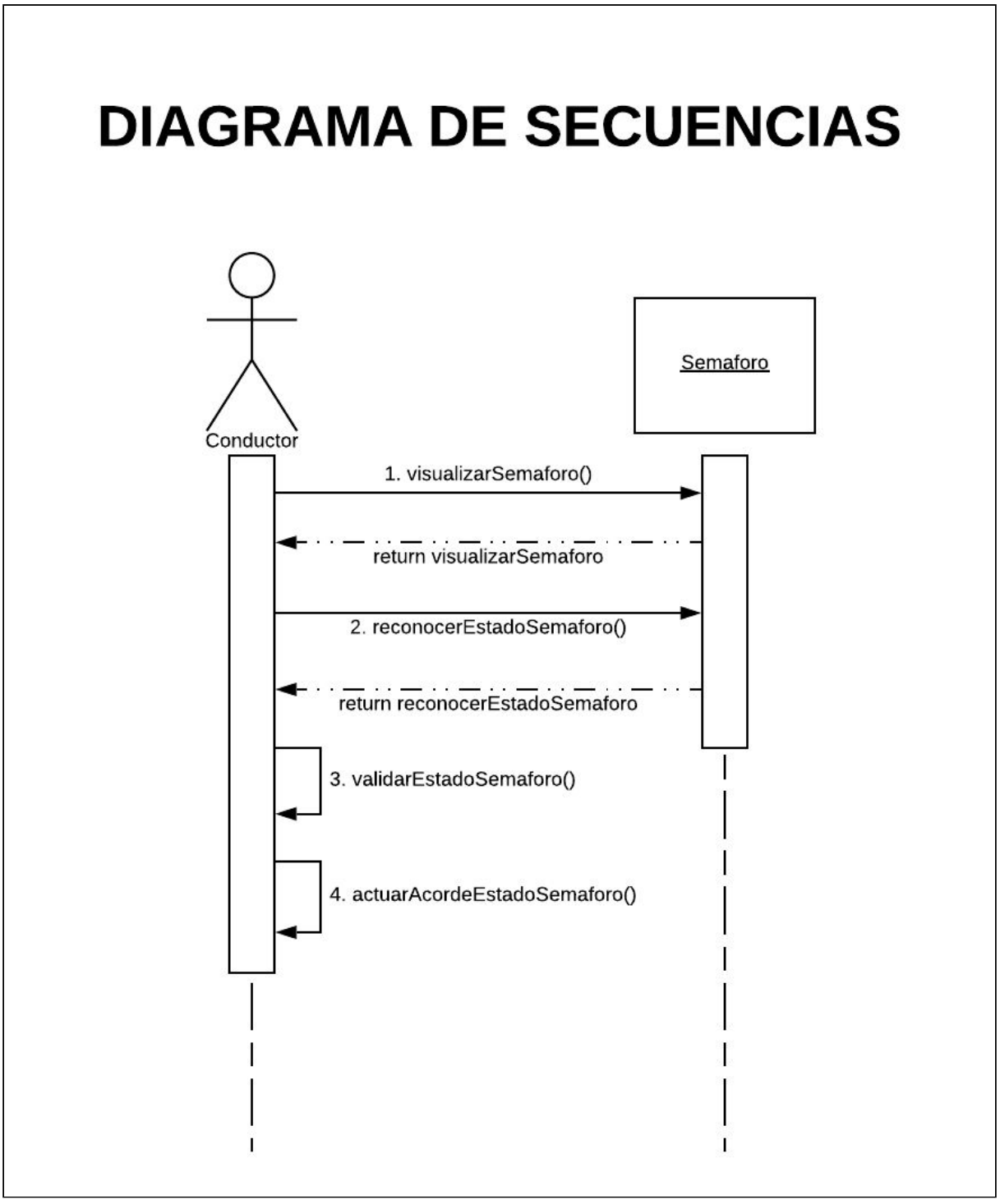

Figura 4: Diagrama de secuencias problemática

Descripción de solución

Nuestra propuesta consiste en la implementación de una red neuronal convolucional que nos permita reconocer el estado de los semáforos de la vía pública a una distancia considerable. La entrada será un conjunto de datos de imágenes del estado del semáforo captados con una cámara tomada desde los vehículos e imágenes genéricas tomando en cuenta que la imagen capturada debe presentar el

menor ruido posible, y a partir de estas imágenes nuestra solución podría decirle al usuario el estado en el que se encuentra el semáforo en la calle, el motivo real por el cual se eligió una red neuronal convolucional es por su característica principal de shift invariant que les permite aprender a través de los patrones concretos encontrados en la imagen de entrada, entonces si se desea probar el modelo implementado con una variación de la imagen ya sea rotada o transpuesta en diferentes posiciones el modelo podrá reconocer los patrones aprendidos, y generará los resultados correctos si el entrenamiento fue acertado como se expone en [13].

Nuestra solución es la implementación de una red neuronal convolucional que recibe como entrada imágenes tomadas desde el interior de un vehículo simulando cómo ve el conductor un semáforo e imágenes genéricas. Las imágenes son clasificadas por su clase, las cuales son amarillo, verde y rojo que representan el estado del semáforo y normalizadas a dimensiones cuadradas, en nuestro caso 50 * 50 pixeles. Contamos con un total de 3374 imágenes, de las cuales 2032 son rojas, 314 amarillas y 1025 verdes. Estas imágenes son usadas para el entrenamiento del modelo. Una vez construido el modelo se desarrolló una interfaz web que permite seleccionar una imagen de prueba para clasificarla, y finalmente mostrar el estado del semáforo.

Figura 5: Imagen genérica para entrenamiento

En base a lo investigado se planteó usar un clasificador kNN el cual de acuerdo a sus siglas en inglés significa K-Nearest-Neighbor se lo define como un algoritmo basado en instancia y usa un modelo supervisado en Machine Learning usado en [5], puede ser utilizado para clasificar nuevas muestras o incluso para predecir nuevos valores, donde su principal función es clasificar valores buscando determinados puntos con mayor similitud, esto se usará en la solución para diferenciar entre los diferentes colores del semáforo que son rojo, amarillo y verde y clasificar los datos de entrada de acuerdo al color de la imagen, además para

detectar los círculos y el color de los píxeles de los datos de entrada se pensaba usar conceptos de visión por computador presentados en [7], [9] , pero luego de evaluar las diferentes soluciones posibles el uso de esta herramienta fue descartada, tomando como solución lo que se expusó previamente.

Otros conceptos que se utilizaron en menor proporción de lo esperado fueron el uso de técnicas de visión por computador para poder identificar desde las imágenes de entrada los colores del semáforo como las presentadas en [8], [11], donde se puede destacar las siguientes: detección del ROI, binarización, filtrado de imágenes, reconocimiento de blobs a partir de ROI, normalización, uso de otros espacios de color como el HSV usando threshold, detección de puntos claves usando SURF, se puede acotar que junto con estas técnicas se podría usar un modelo DNN como SNN o Yolo que se describe posteriormente, los cuales se basan en deep learning, a pesar de esto si se utilizaron librerías para hacer un preprocesamiento simple en las imágenes de entrada al modelo.

Otra solución posible planteada que también fue descartada en el camino del desarrollo de la solución fue el uso del algoritmo Yolo utilizado en [10], este es un algoritmo de detección de objetos que usa un enfoque distinto, es una red neuronal convolucional inteligente que permite su uso en tiempo real, se divide la imagen en regiones y se predicen las cajas que bordean y probabilidades de existencia de una clase. Entre sus ventajas está que es extremadamente veloz, en su versión extendida se ha llegado a identificar 45 frames por segundo y en su versión más pequeña se ha logrado un total de 155 frames por segundo, también durante la etapa del entrenamiento se encarga de observar toda la imagen y extrae información de todas las clases que se requieran.

Al efectuar la búsqueda de datos locales sobre vías públicas o carreteras a nivel local se pudo notar que no existe la información sobre estos datos puesto que no son expuestos a conocimiento público por parte de autoridades, por lo tanto se propone usar datasets extranjeros donde ya se tienen datos de diferentes vías públicas donde se aprecian semáforos en diferentes imágenes, en nuestro caso los datos a utilizar para crear el modelo son representados como imágenes puesto que a partir de ellas se procederá a identificar los colores de los semáforos. Se identificaron diferentes datasets internacionales, pero estos poseían una cantidad elevada de ruido que confunden al modelo que se pretende desarrollar, por esta razón se decidió usar el dataset presentado en [4], el cual contiene 2631 imágenes genéricas con una resolución de 224 x 224 píxeles, donde todas poseen un semáforo o más con el fin que el entrenamiento sea lo más correcto posible, además se incluyeron imágenes locales tomadas desde un carro en diferentes puntos de la ciudad para comprobar cómo funciona el modelo con datos locales, con esto se buscaría que el modelo se ajuste a las necesidades de nuestro país, es

importante resaltar que estas imágenes fueron recortadas al área donde se encontraba el semáforo para facilitar el entrenamiento del modelo, incluyendo esas imágenes nuestro datos de entrada fueron un total de 3371 imágenes como se detalló previamente.

En términos generales la solución necesita de componentes esenciales para su funcionamiento, como lo es, una forma de reconocer todos los semáforos de una imagen en sus diferentes ángulos y alturas, necesariamente debería de prepararse u obtenerse un dataset con los puntos donde se encuentra el objeto de estudio, luego en caso de existir varios semáforos, deberá implementarse un algoritmo de selección del mejor candidato y escoger su color, finalmente se debe traducir la salida en un formato entendible por el usuario final.

Modelos completos de diseño de la solución

El modelo de la solución se muestra en un diagrama de actividades, representa 4 etapas, inicialmente se recolectarán las imágenes que actuarán como la entrada de datos para la red neuronal convolucional a implementar, después se tienen 2 capas de convolución con función Relu donde se efectúa la detección de patrones, posteriormente se aplica la capa pooling para refinar el patrón encontrado, luego se repite el mismo proceso usando 2 capas de convolución y una de pooling, sobre esta se aplica la técnica de dropout para tratar el overfitting, seguidamente se utiliza la técnica flatten para pasar del modelo 3D a uno 1D que será la entrada de la red MLP donde internamente se utilizará una capa con una función de activación Relu sobre la cual también se usa el dropout, para finalizar se clasifica los resultados obtenidos de la red neuronal para compararlo con la salida esperada.

En el segundo gráfico se muestra la arquitectura general del modelo que se usará para la solución, las entradas de la CNN serán imágenes que reflejen el estado de diferentes semáforos, además como se mencionó previamente este modelo se encuentra formado por dos capas convolucionales, donde en cada capa se efectúan dos convoluciones y un *pooling,* el objetivo de estas capas es reducir la mayor cantidad de *features* posibles para así retener información relevante para el modelo, para obtener resultados coherentes en la etapa de entrenamiento, posteriormente se establece el clasificador el cual se basa en una MLP donde se usa de forma específica una capa con una función Relu y la salida se obtiene usando la función Softmax, se utiliza esta función para expresar la salida ya que esta se basa en una distribución probabilística que permitirá determinar a qué clase pertenece el ejemplo con el cual se está entrenando, donde las clases en este caso serían 3 correspondientes a los estados posibles del semáforo.

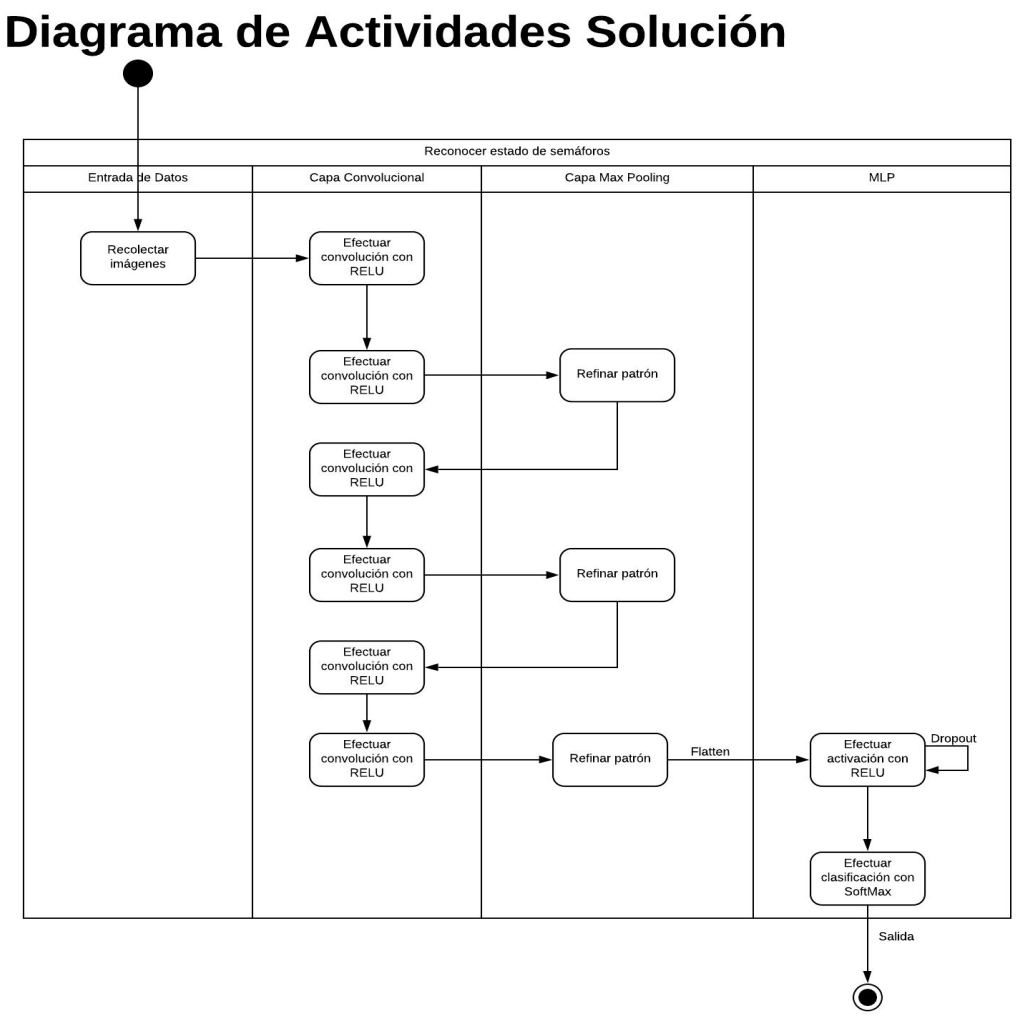

Figura 6: Diagrama de actividades solución

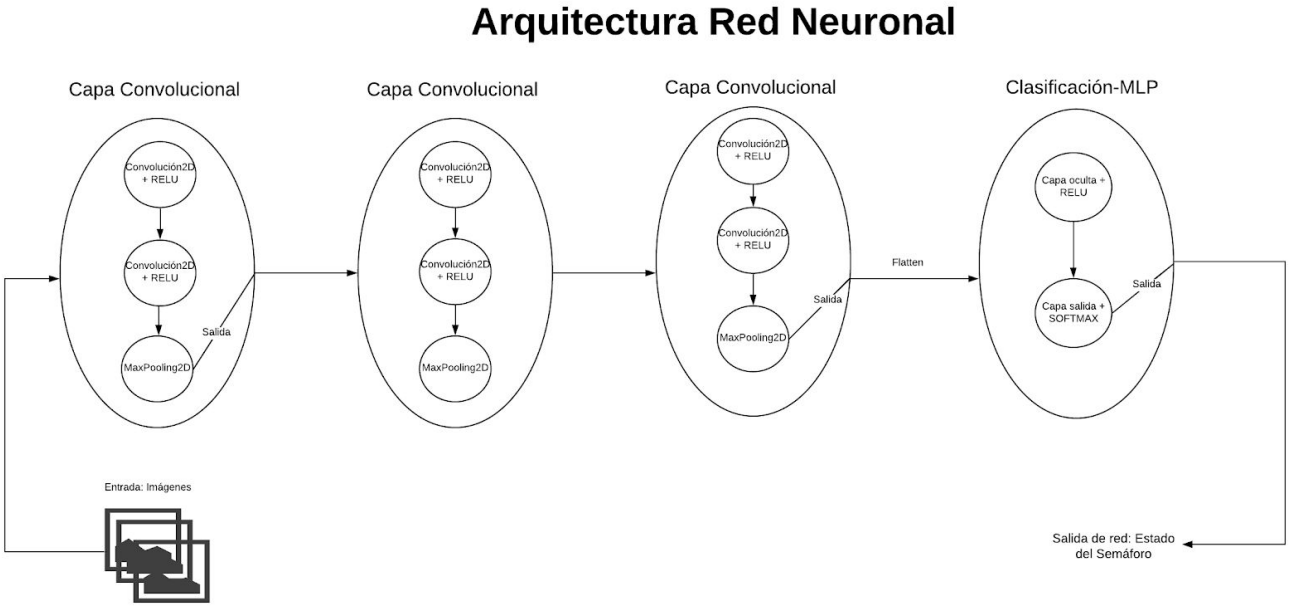

Figura 7: Arquitectura red neuronal

Descripción de la implementación

El lenguaje de programación seleccionado para implementar la solución propuesta es Python, debido a las librerías que ofrece para desarrollar aplicaciones basadas en inteligencia artificial como es el caso de Tensor Flow y Keras las cuales serán usadas para el desarrollo de este proyecto, su función principal será el desarrollo del modelo, su entrenamiento, validación y pruebas finales, entre los componentes que se destacan de esta librerías se encuentran *Sequential, Dense, Dropout, Flatten, Conv2D, MaxPooling2D,* entre otros, el principal motivo de la elección de estas librerías es por la variedad de funcionalidades que brindan, facilidades para programar y efectuar diversas operaciones en pocos pasos. Otro de los paquetes que se usarán del lenguaje de programación corresponden a aquellos que serán utilizados para realizar ciertos cálculos de visión por computador donde destaca *opencv, numpy y pandas.* La plataforma utilizada para la implementación y ejecución del código será una máquina local debido a que contamos con los suficientes recursos computacionales suficientes para ejecutar el proyecto sin complicaciones, luego de evaluar las plataformas de ejecución se descartó la opción de migrar el proyecto a Google Colab, a pesar que su plataforma es completa y con gran cantidad de recursos uno de los equipos de cómputo de los miembros del grupo fue suficiente para trabajar en el proyecto, pero en caso que se trabaje en un proyecto de mayor escala podría usarse esta plataforma.

Los hiperparametros utilizados para el desarrollo del modelo incluyen los siguientes:

- Número de épocas que corresponde a la cantidad de veces que se ejecutará el entrenamiento del modelo propuesto, fue seleccionado porque en base al estado del arte este parámetro es fundamental para identificar máximos globales y no estancarse en mínimos locales.
- Tamaño del lote el cual divide en lotes los datos de entrenamiento del modelo para evaluar en base al tamaño de lotes establecido en cada iteración con el fin de conocer el comportamiento del error del modelo desarrollado.
- Proporción de prueba el cual se establece como un porcentaje para determinar la cantidad de datos de entrada que se usarán en entrenamiento y pruebas, permite segmentar los datos de entrada de forma específica.
- Proporción de validación el cual puede ser definido como el porcentaje de las imágenes de entrenamiento restantes que se usarán para el proceso de validación, al igual que el anterior permite segmentar y categorizar los datos de entrada de forma específica consiguiendo que el entrenamiento del modelo sea ordenado y puedan obtenerse las características deseadas.
- Tamaño de las imágenes de entrada puesto que como se conoce estas tienden a variar por cada imagen, entonces para que el modelo aprenda en base a un patrón constante se necesitaba que todas sean del mismo tamaño, de igual forma al efectuar pruebas las imágenes debían ser redimensionadas al valor establecido para que el resultado sea el esperado.

Los parámetros presentados previamente pueden definirse como generales para todo el modelo implementado que son usados para todas las etapas de entrenamiento, por otro lado se tienen hiper parámetros que fueron usados en las capas convolucionales y mlp que se detallan a continuación:

- Tamaño de los filtros de convolución, tamaño de filtros para el pooling y número de filtros los cuales fueron muy relevantes para extraer las características deseadas de los datos de entrada.
- Optimización de Adam es un descenso de gradiente estocástico que se basa en la estimación adaptativa de momentos de primer y segundo orden. Según estudios, el método es definido como computacionalmente eficiente, requiere poca memoria, es invariante al reajuste diagonal de gradientes y es muy adecuado para problemas que son grandes en términos de datos o parámetros.
- Número de neuronas para la red completamente conectada el cual correspondía a un número que podía cambiar de acuerdo a las necesidades del modelo, fue seleccionado porque si se altera este valor los resultaron cambian totalmente entonces fue establecido a un valor constante de acuerdo a las necesidades del modelo del proyecto
- Se usó cross entropy categorical para la función de loss puesto que permite clasificar entre dos o más clases, lo cual era lo deseado en el proyecto ya que se requería que el modelo reconozca la clase a la que pertenece en base a un proceso de clasificación.
- Las funciones de activación también jugaron un papel importante en cada capa, en las convolucionales se utilizó la función ‘Relu’ porque es ideal para extraer las características de las imágenes de entrada, también fue usada en la capa oculta de la MLP, pero en la capa de salida se decidió el uso de Softmax puesto que es una función ideal para efectuar la clasificación lo cual era lo esperado en nuestro modelo que reconozca entre la clase 0, 1, 2, que corresponde a a rojo, amarillo y verde respectivamente.

Resultados

El principal resultado esperado es que nuestro algoritmo sea capaz de reconocer cualquiera de los colores de los semáforos que cumpla con las características básicas con las cuales se entrenaron, y a su vez se le comunique al usuario de alguna manera el estado del semáforo por el cual pretende circular.

Otro resultado derivado del anterior es que se espera que nuestro algoritmo pueda clasificar correctamente las imágenes que poseen semáforos con respecto a las que no los poseen, con el fin de validar el correcto funcionamiento del modelo planteado para esto como se expuso previamente se usarán datos de entrada que no posean semáforos.

Para evaluar nuestra solución calcularemos el *accuracy,* el cual es definido como la calidad o estado de ser correcto o preciso del modelo propuesto, así como el porcentaje de error. Estos parámetros nos permitirán conocer qué tan preciso es el modelo al evaluarlo con diferentes datos de entrenamiento. Además como parte de los resultados se tendrá una etapa de prueba donde se usarán datos diferentes a los que se usaron para entrenar el modelo, con el fin de mostrar el reconocimiento del estado de los semáforos en una imagen con etiquetas específicas, esta función se presenta en la figura 8

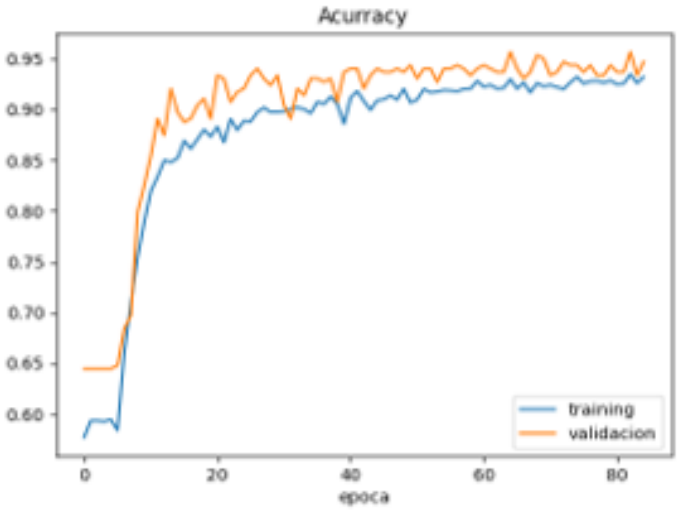

Figura 8: Exactitud del modelo

Dado que luego de 85 épocas el error del modelo no disminuye, entonces concluimos que se había llegado a un mínimo global, es decir, no era posible reducir el error más allá de ese número de épocas, en la figura 9 se presenta el comportamiento de la función loss.

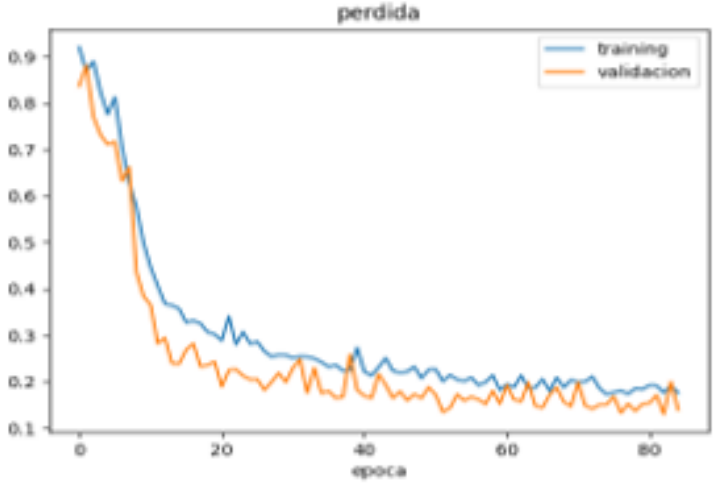

Figura 9: Error del modelo

Conclusiones

La precisión de la solución se encuentra en un rango aceptable, aunque se podría mejorar aumentando la cantidad de datos de entrada para que se encuentren distribuidos uniformemente evitando el sesgo en el modelo.

Efectuando preprocesamiento adicional a las imágenes aplicando máscaras para realzar los diferentes colores que intervienen en el estado de las luces de tráfico, evitando que el algoritmo aprenda características innecesarias.

Para que el modelo sea capaz de reconocer o clasificar el estado del semáforo en escenas completas, es decir, donde el semáforo no sea protagonista es necesario aumentar el número de capas convolucionales para potenciar la extracción de características.

Contribución

Se entrenó el modelo con semáforos locales para adaptarlo a las necesidades del país.

La solución puede ser reutilizada en otros contextos, es decir, utilizar datos de entrada de otras ciudades o regiones para entrenar al modelo.

Se desarrolló una interfaz web para mostrar de forma interactiva los resultados de la solución a la problemática que presenta imágenes representativas en caso de que sea una aplicación real.

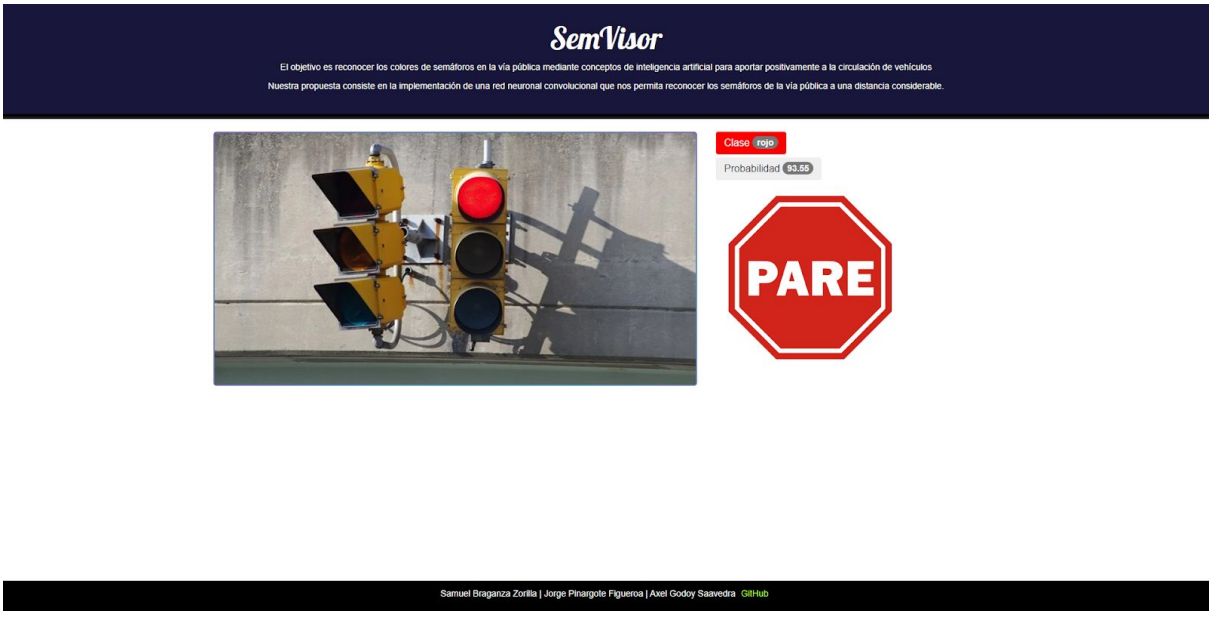

Figura 10: Etapa de Prueba del clasificador con una imágen de entrada de un semáforo con luz roja

Referencias

[1] Y.-C. Chung, J.-M. Wang y S.-W. Chen, «A Vision-Based Traffic Light Detection System at Intersections,» Journal of Taiwan Normal University: Mathematics, Science & Technology, vol. 47, nº 1, pp. 67-86, 2002.

[2] J. Levinson, J. Askeland, J. Dolson y S. Thrun, «Traffic light mapping, localization, and state detection for autonomous vehicles,» 2011 IEEE International Conference on Robotics and Automation, pp. 5784-5791, 2011.

[3] N. Fairfield y C. Urmson, «Traffic Light Mapping and Detection. Traffic Light Mapping and Detection,» 2011 IEEE International Conference on Robotics and Automation, pp. 5421-5426, 2011.

[4] J. Dunn, «Dropbox,» 12 10 2017. [En línea]. Available: https://www.dropbox.com/s/87xark39qyer8df/TLdataset02.zip?dl=0.

[5] S. Bhartiya y L. Reznik, «Traffic Light Detection,» 2016.

[6] J. Gong, Y. Jiang, G. Xiong, C. Guan, G. Tao y H. Chen, «The recognition and tracking of traffic lights based on color segmentation and CAMSHIFT for intelligent vehicles,» 2010 IEEE Intelligent Vehicles Symposium, pp. 431-435, 2010.

[7] T. Ibrayev y R. Sarmukhanov, «Traffic light recognition system for people with color blindness,» Advances in Memristor Circuits and Bioinspired Systems, vol. 1, pp. 80-83, 2015.

[8] T. H.-P. Tran, C. C. Pham, T. P. Nguyen, T. T. Duong y J. W. Jeon, «Real-time traffic light detection using color density,» 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), nº 1-4, 2016.

[9] K. Yoneda, A. Kuramoto, N. Suganuma, T. Asaka, M. Adibaja y R. Yanase, «Robust Traffic Light and Arrow Detection Using Digital Map with Spatial Prior Information for Automated Driving,» Sensors, vol. 20, nº 4, 2020.

[10] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), nº 779-788, 2016.

[11] Y. K. Kim, K. W. Kim y X. Yang, «Real Time Traffic Light Recognition System for Color Vision Deficiencies,» 2007 International Conference on Mechatronics and Automation, nº 76-81, 2007.

[12] L. Possatti, R. Guidolini, V. B. Cardoso, R. F. Berriel, T. Paixão, M. Badue, A. F. De Souza y T. Oliveira-Santos, «Traffic Light Recognition Using Deep Learning

and Prior Maps for Autonomous Cars,» 2019 International Joint Conference on Neural Networks (IJCNN), nº 1-8, 2019.

[13] R. Zhang, «Making Convolutional Networks Shift-Invariant Again,» 2019 Proceedings of the 36 th International Conference on Machine Learning