Revista Científica y Tecnológica UPSE

Detección de Mascarilla para COVID-19 a través de Aprendizaje Profundo usando OpenCV y Cascade Trainer GUI

Mask detection for COVID-19 through of Deep Learning using OpenCV and Cascade Trainer GUI

Luis Chuquimarca Jiménez * https://orcid.org/0000-0003-3296-4309, Santiago Pinzón Tituana https://orcid.org/0000-0003-1055-6507, Anthony Rosales Pincay https://orcid.org/0000-0003-3871-1679.

Universidad Estatal Península de Santa Elena, Ecuador.

Resumen

La pandemia del covid-19 está provocando una crisis de salud a nivel mundial, una de las recomendaciones de los científicos y gobiernos para evitar contagios es el uso de mascarilla. Basados en esta precisión, el presente artículo muestra el desarrollo de un software que permite detectar la mascarilla en distintos escenario usando el lenguaje de programación de Python mediante las librerías de cv2, os, Numpy y Imutils, utilizando redes neuronales convolucionales más eficaces que las redes neuronales comunes, las cuales fueron entrenadas con el software Cascade Trainer GUI, usando diferentes cantidades de bases de datos desde 400 hasta 1400 imágenes para comparar los distintos tipos de precisión del sistema de detección de la mascarilla. Sin embargo, de la primera base de datos no se obtuvo una buena presión por una baja cantidad de falsos positivos, por lo cual a medida que se usa más datos la precisión fue aumentando considerablemente hasta obtener una precisión de 92 % con mascarilla y un 100% sin mascarilla.

Palabras clave:

Aprendizaje de máquina, Aprendizaje profundo, redes convolucionales, falsos positivos.

Abstract

The pandemic of covid-19 is causing a health crisis worldwide, one of the recommendations of scientists and governments to avoid contagion is the use of masks. Based on this, this paper shows the development of a software to detect the mask in different scenarios using Python programming language through cv2, os, Numpy and Imutils libraries, using convolutional neural networks more efficient than common neural networks, which were trained with Cascade Trainer GUI software, using different data-base quantities from 400 to 1400 images to compare the different types of accuracy of the mask detection system. However, the first database did not have a good pressure due to a low number of false positives, so as more data was used, the accuracy increased considerably until an accuracy of 92% with mask and 100% without mask was obtained.

Keywords:

Machine Learning, Deep Learning, convolutional networks, false positives.

Recibido: abril 6/2021 **Aceptado:** mayo 7/2021 **Publicado:** junio 25/ 2021

Forma de citar: Chuquimarca Jiménez, L.; Pinzón Tituana, S.; Rosales Pincay, A. (2021). Detección de Mascarilla para COVID-19 a través de Aprendizaje Profundo usando OpenCV y Cascade Trainer GUI. Revista Científica y Tecnológica UPSE, 8 (1) pág. 68-73. DOI: 10.26423/rctu.v8i1.572

* *Autor para correspondencia:* lchuquimarca@upse.edu.ec

1. Introducción

En el año 2020, considerando los riesgos de la pandemia del Covid-19, los países tomaron decisiones para lograr controlar la propagación de este virus, las principales acciones para controlarlo fue la utilización obligatoria de la mascarilla.

Científicos comprobaron que el uso regular de mascarillas minimiza la transmisión del covid-19 [1]. En distintos países como Francia las autoridades implementaron un sistema de inteligencia artificial para el control de mascarillas en los medios de transporte.

El presente artículo permite el desarrollo de un software que sea capaz de identificar el uso de mascarillas en tiempo real, en distintas situaciones mediante el uso de inteligencia artificial [2].

La inteligencia artificial, Aprendizaje de Máquina y Aprendizaje Profundo ofrecen herramientas viables contra la pandemia actual mediante el uso de software implementado en dispositivo de bajo costo; en el presente artículo se presenta una de ellas.

Dentro del Aprendizaje de Máquina, este se utiliza para la clasificación de los fragmentos de las imágenes procesadas por el software IDLE.

La técnica de Aprendizaje Profundo se utiliza para resaltar los elementos principales mediante el uso de varias capas en diferentes niveles especializados.

Para detectar la forma, bordes y esquinas de la mascarilla se utiliza una red neuronal convolucional, mediante un entrenamiento de la red para optimizar el proceso.

2. Materiales y métodos

En la implementación de un software que detecte el uso de mascarilla en tiempo real con base a una red neuronal, es indispensable contar con un IDE, usando el método deductivo.

El lenguaje con el que se realiza el proyecto es Python el cual necesita de algunos framework compatibles con los lenguajes ya mencionados como son OpenCV y Cascade Trainer GUI, además se debe tener disponibles las siguientes librerías como son cv2, numpy, imutils y os.

Luego se crea una base de datos con imágenes positivas y negativas, es decir se crea dos carpetas: una para las imágenes positivas en las cuales como característica principal las personas deben estar usando mascarilla; y, la segunda carpeta debe constar de imágenes de personas sin mascarillas, tomando en consideración que entre más imágenes, mayor precisión y exactitud se tiene en la detección de objetos, por lo que no se tiene mayor cantidad de falsos positivos, dando así una mejor sensibilidad a la hora de detectar (ver Figura 1).

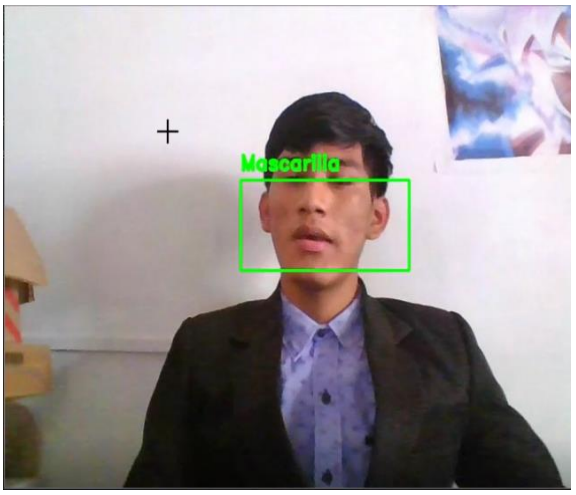

Figura 1. Falso positive.

2.1. Aprendizaje de Máquina

Aprendizaje de Máquina se subdivide en distintas categorías, una de ellas es el aprendizaje sin supervisar que tiene como objetivo que la máquina encuentre en qué categoría se encuentran los datos de entrenamiento (ver Figura 2). Por ejemplo, esta técnica se utiliza para determinar distintas emociones luego clasificar los datos fuera del grupo de entrenamiento.

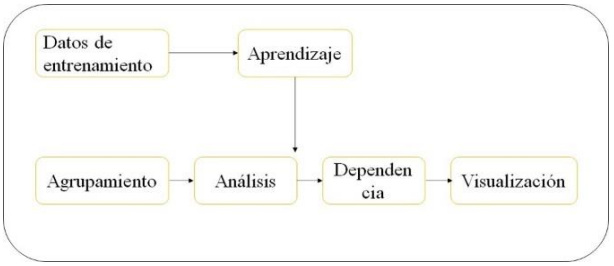

Figura 2. Aprendizaje de Máquina sin supervisar

Aprendizaje supervisado tiene como objetivo que la máquina asigne etiquetas, los datos de entrenamiento se subdividen en etiquetados y no etiquetados (ver Figura 3). Por ejemplo, en la fotografía de una placa de vehículo, la máquina etiqueta los números y las letras de la fotografía. También se puede hacer una combinación de aprendizaje no supervisado y aprendizaje supervisado (ver Figura 4).

Para poder representar los datos como un grupo de objetos denominados instancias, cada instancia se especifica usando números llamados atributos. En cambio, para clasificar fragmentos de alguna imagen se presentan como fragmentos verdadero o fragmentos desecho [4].

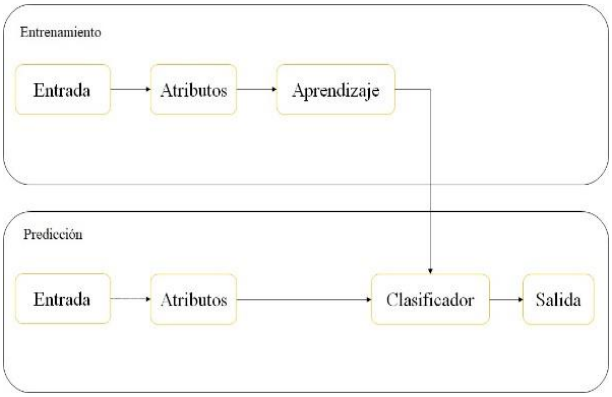

Figura 3. Aprendizaje de Máquina supervisado

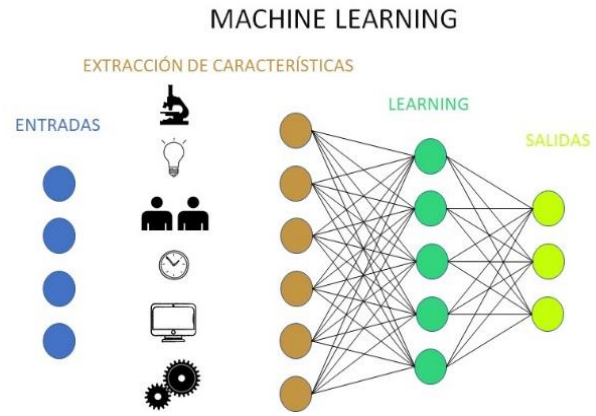

Figura 4. Aprendizaje de Máquina

2.2. Aprendizaje Profundo

El Aprendizaje de Máquina está limitado en muchos aspectos en la actualidad, para suplir el Aprendizaje de Máquina surge el Aprendizaje Profundo que ha tomado mucha fuerza en la actualidad.

El Aprendizaje Profundo es un conjunto de métodos que se le confiere a una máquina para que se alimente con datos brutos para que automáticamente detecte y clasifique los tipos de datos. Aprendizaje Profundo son métodos de aprendizaje con múltiples niveles de representación.

La formación de módulos simples no lineales, donde cada uno de los módulos convierte la representación a un nivel diferente y con la formación de varias conversiones de este tipo se memoriza las funciones complejas, para la clasificación de los distintos datos, las capas superiores resaltan los aspectos más relevantes para la exclusión de los aspectos no relevantes.

Una imagen se representa por una matriz de pixeles, en la primera capa se detecta la presencia o ausencia de bordes y se determinada ubicación.

En la segunda capa se detecta la disposición particular de bordes de las pequeñas variaciones de la ubicación en las imágenes.

En la tercera capa se reconstruye las partes del objeto a detectar y las siguientes capas se detecta el objeto (ver Figura 5) [5].

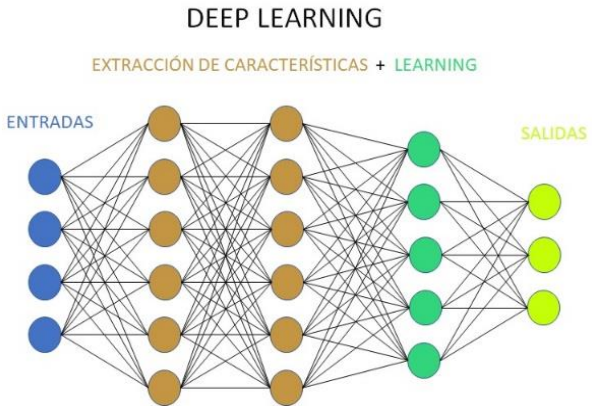

Figura 5. Aprendizaje Profundo

2.3. Redes Neuronales Convolucionales

Actualmente, las redes neuronales convolucionales son muy utilizadas en distintos ámbitos como son las ciencias de datos, visión por computadora, medicina, entre otras. En el tratamiento de imágenes tienen un éxito rotundo gracias a capacidades actuales de cómputo. Para entender cómo los humanos procesan los datos visuales se involucra desarrollar un proceso muy complejo. Durante varios años se desarrollaron sistemas ineficientes, recientemente las redes neuronales convolucionales dieron un gran avance en la detección de objetos y categorización de los mismo alcanzado un nivel parecido al de los humanos.

Las redes neuronales convolucionales pueden clasificar y reconocer objetos por su forma, para lograr reconocer las formas aun cuando las señales son eliminadas, para esto debe detectar los rasgos distintivos de la imagen como sus esquinas y bordes, hasta reconocer rasgos más complejos de la imagen.

Las primeras capas de la red cuentan con una sensibilidad parecida a la de los humanos, esto también depende de cuántos datos se brinda para el entrenamiento de la red, entre más datos mejor para que no dé falsos positivo (ver Figura 6) [6].

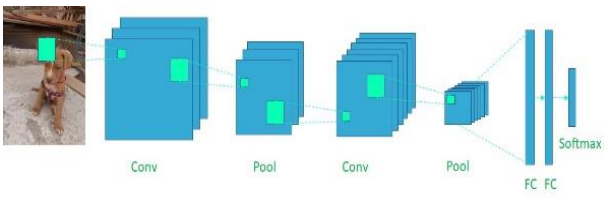

Figura 6. Redes neuronales convolucionales

2.4. Python

Python en un lenguaje de programación de alto nivel de interpretación, es decir que no es compilado, sino que

ejecuta directamente las líneas de código. Se puede decir que es el sucesor del lenguaje de programación ABC, el cual tiene una forma de multiparadigma; es decir, que soporta diferentes tipos de programación, como son: programación orientada a objetos, programación imperativa y programación funcional.

En diciembre 1989 Guido Van Rossum creó Python, el nombre proviene de Monty Python. Una de las características principales de Python es tener un lenguaje de programación fácil, intuitivo, potente, amigable, desarrollado en menor tiempo, uno de los lenguajes más usados últimamente y de mayor eficiencia en aplicaciones actuales [7].

2.5. Numpy

Numpy es una librería matemática que permite trabajar con matrices (N-Array) en el procesamiento de imágenes, por lo general necesita de un código para recorrer píxel por píxel para trabajar con pixeles 2D, ya que los bucles son muy ineficientes y tardan más, facilitando mucho este proceso, que permite grandes cantidades de matrices multidimensionales y aparte cuenta con distintas funciones para el procesamiento de imágenes; cabe mencionar que utiliza lenguaje C [8]. Además, Numpy almacena y accede a los datos eficientemente de un N-array, las matrices tienen el mismo tipo de datos para todos sus elementos y aparte cada elemento ocupa la misma cantidad de bits en la memoria. Por ejemplo, los videos son descompuestos en matrices de cuatro dimensiones de la siguiente forma (T, M, N, 3) [9].

2.6. Imutils

La librería de Imutils en un conjunto de funciones que sirven para el procesamiento de imágenes como cambio de tamaño, clasificación de contornos, esqueletización, rotación, visualización, entre otras (ver Figura 7) [10].

Figura 7. Funciones para procesar imágenes

2.7. Cv2

La librería de OpenCV cuenta con varios paquetes, en sí son cuatro, tiene que seleccionar uno de estos ya que si instala varios tendrá problemas a la hora de ejecutar su código ya que los cuatro paquetes de importan como cv2 [11].

2.8. Os

Os permite usar funciones dependientes del sistema operativo, permite leer, escribir, manipular la ruta del destino del archivo y crear archivos temporales [12].

2.9. OpenCV

OpenCV fue presentada por Intel en el año 2000, es una librería de visión artificial y open source, programada en lenguaje C y C++. Se considera como una multiplataforma, es decir, que cuenta con versiones para Windows, Linux y Mac OS X. Además, cuenta con un campo muy amplio donde se puede aplicar tanto para la investigación como prácticas estudiantiles, con más de 500 funciones para el procesamiento de imágenes desde el reconocimiento facial hasta visión robótica y muchas otras más. Tiene varios paquetes para usuarios avanzados como HighGUI y CvCam, para usuarios de nivel medio cuenta con Hawk y para usuarios de Matlab OpenCV Toolbox [13].

2.10. Cascade Trainer GUI

Cascade Trainer GUI es un software que solo tiene versión para Windows que sirve para entrenar el código, para esto es necesario crear una carpeta con dos subcarpetas una con el nombre de "p" donde se encuentran las imágenes positivas y la otra subcarpeta con el nombre de "n" donde se encontrarán las imágenes negativas. Como se especificó previamente, es una herramienta complementaria para usar con OpenCV, cuenta con una interfaz gráfica muy intuitiva para el usuario (ver Figura 8) [14].

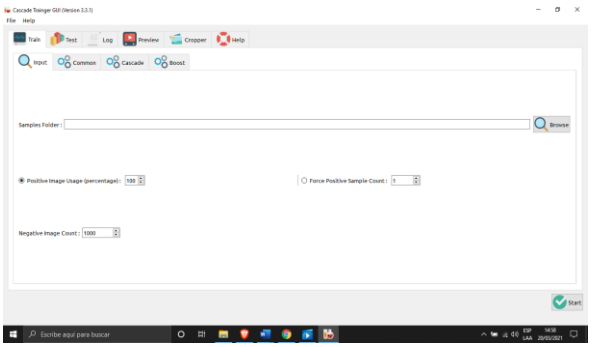

Figura 8. Interfaz gráfico de Cascade Trainer GUI

3. Discusión y resultados

Evaluando el rendimiento de nuestro detector de mascarillas, para calcular el valor de la precisión se utiliza la ecuación 1, donde TP son los verdaderos positivos; es decir, cuándo se detecta correctamente la mascarilla y FP son los falsos negativos que es cuando no detecta la mascarilla o cuando detecta fantasmas.

$$Presicion = \frac{TP}{TP+FP} \qquad \textbf{EC.1}$$

Cabe recalcar que no detecta si está usando correctamente la mascarilla y depende mucho de cuantos datos se proporcione para el entrenamiento de nuestra red neuronal entre más datos mejor funcionará el sistema de detección (ver Figura 9) [15].

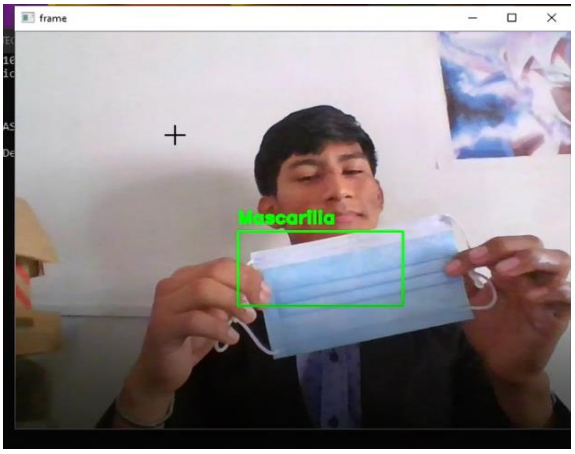

Figura 9. Test de reconocimiento de mascarilla

Para los resultados de la prueba 1 se toma en consideración los parámetros de la Tabla 1, los cuales no son de alta precisión como se indica en la Tabla 2, con la mascarilla cuenta con una precisión de 53% y sin mascarilla cuenta con una precisión del 45%, esto se debe a que existen falsos positivos o conocidos como fantasmas, esto se debe a que al entrenar la red neuronal se tiene muchos datos erróneos.

Tabla 1. Parámetros de la prueba 1

Parámetros	Valores
Imágenes positivas	200
Imágenes negativas	200

Fuente: Autoría propia.

Tabla 2. Resultados de la prueba 1

Métricas	Con mascarilla	Sin mascarilla
Precisión	53%	45%

Fuente: Autoría propia.

Se considera los parámetros de la Tabla 3, para la obtención de los resultados de la prueba 3 indicados en la Tabla 4, en el caso con mascarilla la precisión es de 68% y sin mascarilla es de 72%, la precisión sin mascarilla es mayor que con mascarilla, se reduce la diferencia a un 4%.

Tabla 3. Parámetros de la prueba 3

Parámetros	Valores
Imágenes positivas	400
Imágenes negativas	400

Fuente: Autoría propia.

Tabla 4. Resultados de la prueba 3

Métricas	Con mascarilla	Sin mascarilla
Precisión	68%	72%

Fuente: Autoría propia.

Los parámetros que se indican en la Tabla 5, son utilizados para obtener los resultados de la prueba 6 indicados en la Tabla 6, los cuales son excelentes de acuerdo con la precisión con mascarilla es de 92% y sin mascarilla es de 100% (ver Figura 10), sigue la tendencia que la precisión más alta es la sin mascarilla, se puede mejorar los resultados usando más datos, la diferencia entre las dos precisiones es de 8%.

Tabla 5. Parámetros de la prueba 6

Parámetros	Valores
Imágenes positivas	700
Imágenes negativas	700

Fuente: Autoría propia.

Tabla 6. Resultado de la prueba 6

Métricas	Con mascarilla	Sin mascarilla
Precisión	92%	100%

Fuente: Autoría propia.

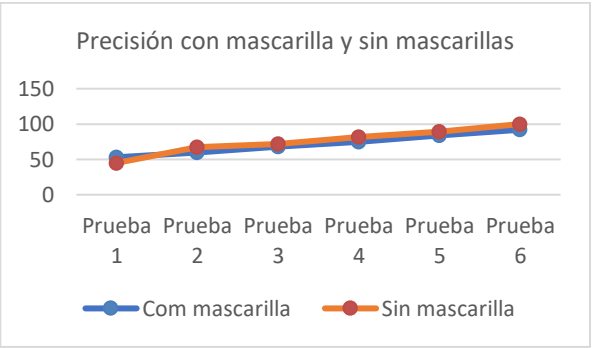

Figura 10. Precisión con mascarilla y sin mascarilla

Figura 11. Test con 1400 imágenes.

4. Conclusiones

En conclusión, se construyó una red neuronal para detectar la mascarilla con una base de datos desde 400 hasta 1400 imágenes, probando su precisión con los distintos parámetros de los grupos de imágenes, creando una red neuronal y aplicando distintas técnicas de procesamiento de imágenes como erosión y dilatación.

Se comprobó el funcionamiento en tiempo real obteniendo excelentes resultados de precisión con mascarilla es de 92% y sin mascarilla es de 100%. Se puede implementar en distintos entornos para controlar el uso de mascarillas puesto que no es muy difícil de implementarlo y su costo es mínimo.

5. Referencias bibliográficas

[1] Andrés, J. M. A., de Castro, M. T. G. V., Vicente-Guijarro, J., Peribáñez, J. B., Haro, M. G., Valencia-Martín, J. L., ... & Montero, G. G. (2020). Mascarillas como equipo de protección individual durante la pandemia de COVID-19: cómo, cuándo y cuáles deben utilizarse. Journal of Healthcare Quality Research, 35(4), 245-252.

[2] Loey, M., Manogaran, G., Taha, M. H. N., & Khalifa, N. E. M. (2021). A hybrid deep transfer learning model with Machine Learning methods for face mask detection in the era of the COVID-19 pandemic. Measurement, 167, 108288.

[3] Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., & Lloyd, S. (2017). Quantum Machine Learning. Nature, 549(7671), 195-202.

[4] Kan, A. (2017). Machine Learning applications in cell image analysis. Immunology and cell biology, 95(6), 525-530.

[5] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. nature, 521(7553), 436-444.

[6] Kubilius, J., Bracci, S., & Op de Beeck, H. P. (2016). Deep neural networks as a computational model for human shape sensitivity. PLoS computational biology, 12(4), e1004896.

[7] Python https://www.python.org/

[8] Bauckhage, C. (2015). NumPy/SciPy Recipes for Image Processing: Creating Fractal Images. researchgate. net, Feb.

[9] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585(7825), 357-362.

[10] Pypi https://pypi.org/project/imutils/

[11] Pypi https://pypi.org/project/OpenCV-python/

[12] Python https://docs.python.org/es/3.10/library/os.html

[13] Arévalo, V., González, J., & Ambrosio, G. (2004). La Librería de Visión Artificial OpenCV. Aplicación a la Docencia e Investigación. Base Informática, 40, 61-66.

[14] Cascade Trainer GUI https://amin-ahmadi.com/cascade-trainer-gui/

[15] Nofallah, S., Mehta, S., Mercan, E., Knezevich, S., May, C. J., Weaver, D., ... & Shapiro, L. (2021). Machine Learning techniques for mitoses classification. Computerized Medical Imaging and Graphics, 87, 101832