UNIVERSIDAD PERUANA UNIÓN

FACULTAD DE INGENIERÍA Y ARQUITECTURA

Escuela Profesional de Ingeniería de Sistemas

Modelo de aprendizaje profundo utilizando Transfer Learning para la clasificación del uso de mascarillas faciales en imágenes de baja resolución

Tesis para obtener el Título Profesional de Ingeniero de Sistemas

Autor:

Hebert Yamil Coazaca Bustamante

Pandely Sabina Ali Vilca

Asesor:

Mg. Ferdinand Edgardo Pineda Ancco

Juliaca, diciembre de 2023

DECLARACIÓN JURADA DE ORIGINALIDAD DE TESIS

Yo Mg. Ferdinand Edgardo Pineda Ancco, docente de la Facultad de Ingeniería y Arquitectura, Escuela Profesional de Ingeniería de Sistemas, de la Universidad Peruana Unión.

DECLARO:

Que la presente investigación titulada: **"MODELO DE APRENDIZAJE PROFUNDO UTILIZANDO TRANSFER LEARNING PARA LA CLASIFICACIÓN DEL USO DE MASCARILLAS FACIALES EN IMÁGENES DE BAJA RESOLUCIÓN"** de los autores **Hebert Yamil Coazaca Bustamante** y **Pandely Sabina Ali Vilca**, tiene un índice de similitud de 10% verificable en el informe del programa Turnitin, y fue realizada en la Universidad Peruana Unión bajo mi dirección.

En tal sentido asumo la responsabilidad que corresponde ante cualquier falsedad u omisión de los documentos como de la información aportada, firmo la presente declaración en la ciudad de Juliaca, a los 28 días del mes de diciembre del año 2023.

Mg. Ferdinand Edgardo Pineda Ancco

048

ACTA DE SUSTENTACIÓN DE TESIS

En Puno, Juliaca, Villa Chullunquiani, a 04 día(s) del mes de diciembre del año 2023 siendo las 14:00 horas, se reunieron los miembros del jurado en la Universidad Peruana Unión Campus Juliaca, bajo la dirección del (de la) presidente(a): Mgc Benazir Francis Herrera Yucra el (la) secretario(a): Ing Jenson Daniel Yahambi Aguilar y los demás miembros: Mg Abel Angel Cullen Macalupu, Mg Freddy Abel Huanca Torres y el (la) asesor(a) Mgc Ferdinand Edgardo Pineda Ancco con el propósito de administrar el acto académico de sustentación de la tesis titulado: Modelo de aprendizaje profundo utilizando Transfer Learning para la clasificación del uso de mascarillas faciales en imágenes de baja resolución del(los) bachiller/es: a) Hebert Yamil Zeazaca Bustamante b) Randely Sabina Ali Vilca c) .....

.....conducente a la obtención del título profesional de: .....

Ingeniero de Sistemas
(Denominación del Título Profesional)

El Presidente inició el acto académico de sustentación invitando al (a la) / a (los) (las) candidato(a)/s hacer uso del tiempo determinado para su exposición. Concluida la exposición, el Presidente invitó a los demás miembros del jurado a efectuar las preguntas, y aclaraciones pertinentes, las cuales fueron absueltas por al (a la) / a (los) (las) candidato(a)/s. Luego, se produjo un receso para las deliberaciones y la emisión del dictamen del jurado.

Posteriormente, el jurado procedió a dejar constancia escrita sobre la evaluación en la presente acta, con el dictamen siguiente:

Bachiller (a): Hebert Yamil Zeazaca Bustamante

CALIFICACIÓN	ESCALAS			Mérito
	Vigesimal	Literal	Cualitativa	
Aprobado	18	A-	Muy Bueno	Sobresaliente

Bachiller (b): Randely Sabina Ali Vilca

CALIFICACIÓN	ESCALAS			Mérito
	Vigesimal	Literal	Cualitativa	
Aprobado	18	A-	Muy Bueno	Sobresaliente

Bachiller (c): .....

CALIFICACIÓN	ESCALAS			Mérito
	Vigesimal	Literal	Cualitativa	

(*) *Ver parte posterior*

Finalmente, el Presidente del jurado invitó al (a la) / a (los) (las) candidato(a)/s a ponerse de pie, para recibir la evaluación final y concluir el acto académico de sustentación procediéndose a registrar las firmas respectivas.

Presidente/a

Secretario/a

Asesora

Miembro

Miembro

Bachiller (a)

Bachiller (b)

Bachiller (c)

Agradecimientos

Queremos expresar nuestro agradecimiento a Dios, cuya guía y fortaleza han sido la luz que ha iluminado nuestro camino durante esta travesía académica. A nuestros padres y familia, nuestra gratitud infinita por su inquebrantable apoyo y amor incondicional, que han sido la base de nuestra perseverancia.

A nuestros estimados docentes, quienes con sabiduría y paciencia nos guiaron, les agradecemos por sembrar la semilla del conocimiento en nuestras mentes.

ÍNDICE

Modelo de aprendizaje profundo utilizando Transfer Learning para la clasificación del uso de mascarillas faciales en imágenes de baja resolución

Resumen

La rápida propagación del virus SARS-Cov-2 alrededor del mundo provocó que muchos países impongan reglas obligatorias para evitar los contagios y reducir el impacto del virus en la población. Una de las medidas más importantes fue el uso obligatorio de mascarillas en lugares públicos, sin embargo, esta medida no fue acatada por todas las personas. Entonces aparecieron varias investigaciones de visión por computadora para detectar el uso de mascarillas en rostros de personas, aunque ninguno utilizó imágenes en baja resolución. Generalmente, las grabaciones en escenas exteriores se encuentran en baja resolución, por lo que este trabajo propone un modelo de aprendizaje profundo para la clasificación del uso de mascarillas en imágenes de baja resolución utilizando técnicas de transfer learning. Se recolectaron imágenes de rostros para crear el conjunto de datos "Low-Res Faces in Perú" y se realizaron cuatro experimentos donde se entrenaron distintas arquitecturas incluyendo modelos diseñados por los autores. El modelo del tercer experimento alcanzó el resultado más alto de accuracy con 99,10%, superando a las arquitecturas que tenían una mayor profundidad de capas convolucionales.

Palabras clave: *baja resolución, redes neuronales convolucionales, covid19, transfer learning, mascarilla, aprendizaje profundo.*

Deep Learning Model using Transfer Learning for Face Mask detection in challenging low resolution images from public outdoor areas

Abstract

The rapid spread of the SARS-CoV-2 virus around the world caused many countries to impose mandatory rules to avoid contagion and reduce the impact of the virus on the population. One of the most important measures was the mandatory use of face masks in public places, however, this measure was not followed by all people. Then, several computer vision investigations appeared to detect the use of masks on people's faces, although none of them used low-resolution images. Generally, recordings in outdoor scenes are in low resolution, so this paper proposes a deep learning model for the classification of face mask use in low resolution images using transfer learning techniques. Images of faces were collected to create the dataset "Low-Res Faces in Peru" and four experiments were performed where different architectures were trained including models designed by the authors. The model from the third experiment achieved the highest accuracy result with 99.10%, outperforming architectures with a deeper convolutional layer structure.

Keywords: low resolution, convolutional neural networks, covid19, deep learning, transfer learning

1. Introducción

La pandemia global provocada por el virus SARS-CoV-2 fue uno de los eventos más importantes y disruptivos en la historia reciente de la humanidad. Desde su primera aparición reportada en diciembre del 2019, el virus se propagó muy rápidamente y en pocos meses estaba presente en muchos países alrededor del mundo. Ante esta situación la Organización Mundial de la Salud (OMS) se vio en la necesidad de coordinar una respuesta internacional y declarar el contagio de la enfermedad como una situación de emergencia global (WHO Director-General's Opening Remarks at the Media Briefing on COVID-19 - 11 March 2020, s/f). Tres años después del brote (diciembre del 2022), la pandemia de COVID-19 ha provocado un total aproximado de 6.7 millones de muertes a nivel global, y el número acumulado de casos confirmados se acerca a los 660 millones de acuerdo a Johns Hopkins University (Johns Hopkins University, s/f).

Para evitar la propagación del virus y prevenir los focos de contagio, el gobierno del Perú estableció un reglamento de bioseguridad que lamentablemente no fue respetado por una parte considerable de la población (Rosas, 2021). Entre las principales medidas recomendadas por organismos internacionales estaban el lavado de manos, el distanciamiento social y el uso de mascarillas faciales (Advice for the Public on COVID-19 – World Health Organization, s/f). El resultado del incumplimiento se veía reflejado en las cifras publicadas por el Ministerio de Salud del Perú, donde se reportó la mayor tasa de fallecidos durante la segunda Ola de COVID-19 con un promedio de 828 decesos por día. Asimismo, a inicios de la tercera ola se registró un promedio de 61,800 casos nuevos por día (Instituto Nacional de Salud y Centro Nacional de Epidemiología, Prevención y Control de Enfermedades-MINSA, s/f).

El caso de Perú también sucedió en otros países donde fue difícil la aplicación de las medidas sanitarias impuestas debido a la gran cantidad de personas en las calles y la falta de recursos para hacer cumplir las medidas. Una de las faltas más graves fue el no uso de mascarillas faciales.

Se ha comprobado que el uso adecuado de mascarillas faciales puede evitar la transmisión de enfermedades respiratorias, ya que las mismas funcionan como un filtro físico que evita la salida o entrada de partículas respiratorias que una persona emite cuando habla, tose o estornuda (Milton et al., 2013; Lai et al., 2012). En el estudio de Garcia (2020) se afirma que incluso si una mascarilla no está totalmente ajustada o es

de material casero, aún puede retener parte de las partículas infecciosas que viajan en el aire a fin de que no alcancen a otra persona. En la investigación de Eikenberry et al. (2020) se concluye que el beneficio a la comunidad que ofrece el uso de mascarilla es mucho mayor cuando se aplica junto a otras medidas de protección y si se cumplen de manera masiva.

En los últimos años, la investigación en el campo de Visión por Computadora se ha popularizado en gran medida y se ha evidenciado su utilidad para solucionar una gran variedad de problemas en la vida del ser humano. Dentro de este campo, las redes neuronales convolucionales (CNN) se han convertido en una herramienta clave para la clasificación de imágenes y detección de objetos; ya que, a diferencia de otras técnicas, las CNN poseen capas que pueden extraer diferentes características de las imágenes y aprender patrones complejos. Otra ventaja de este tipo de redes neuronales es que permite aprovechar la información aprendida por un modelo en la resolución de un problema y aplicarla a la resolución de una tarea nueva. Esta técnica es conocida como transfer learning y se utiliza comúnmente por medio de modelos pre-entrenados con conjuntos de datos muy grandes, ya que es relativamente complicado tener un conjunto de datos con una cantidad de imágenes suficiente para entrenar una red neuronal convolucional desde cero (CS231n Convolutional Neural Networks for Visual Recognition, s/f).

En trabajos previos se elaboraron modelos con el objetivo de detectar rostros con mascarillas como el trabajo de Singh et al. (2021), donde se utilizó transfer learning en las arquitecturas de Yolo versión 3 y Faster Region-Based Convolutional Neural Networks (Faster R-CNN), obteniendo buenos resultados en ambos casos considerando la indisponibilidad de un conjunto de datos de gran escala. En el trabajo de Loey et al. (2021) se desarrolló un modelo híbrido de transfer learning enfocado en la detección de mascarillas faciales compuesto por dos componentes. El primero se encargaba de la extracción de características mediante el modelo ResNet50, luego el segundo componente realizaba la clasificación del uso de mascarillas faciales utilizando el modelo Support Vector Machine (SVM). Se utilizaron las imágenes de tres conjuntos de datos diferentes: Real-World-Masked Face Dataset (RMFD), Simulated Masked Face Dataset (SMFD) y Labeled Faces in the Wild (LFW). Finalmente, la evaluación del modelo en cada conjunto de datos resultó en puntajes de accuracy (exactitud) superiores a 99%.

En la investigación de Suresh et al. (2021) se propone un sistema de detección del uso de mascarillas en tiempo real utilizando OpenCV y la arquitectura de MobileNet. Se utilizó un conjunto de datos de 3918 imágenes y se realizaron pruebas en grabaciones para demostrar el buen desempeño del modelo. Rahman et. al (2020) propone un sistema para identificar si una persona no está usando una mascarilla y posteriormente informar a la autoridad correspondiente en el contexto de una ciudad inteligente. El modelo de CNN propuesto fue diseñado por los autores y estaba conformado por tres pares de capas convolucionales sumado a un grupo de capas densas y de dropout. Se emplearon en total 1539 imágenes, dando como resultado un accuracy del 98,7% y un AUC (Area Under the Curve) de 0,985 en el conjunto de imágenes de prueba. Por otro lado, en la investigación de Oumina et. al. (2020) se utilizaron 3 modelos pre-entrenados de CNN para clasificar imágenes de rostros y detectar si una persona usa o no mascarilla. Se modificó la última capa de redes neuronales densas en cada modelo con el fin de implementar algoritmos más complejos como SVM y K-Nearest Neighbors (K-NN). Se obtuvieron resultados satisfactorios para la detección de rostros enmascarados y el puntaje más alto de accuracy se obtuvo combinando los modelos MobileNetV2 y SVM.

En la mayoría de los trabajos anteriores se utilizaron conjuntos de datos de imágenes de alta resolución tales como fotografías de stock (Real-World Masked Face Dataset, s/f) e imágenes obtenidas a través de herramientas como Web Scraping y superposición de imágenes (Simulated Masked Face Dataset, s/f). Sin embargo, las grabaciones en escenas exteriores se encuentran en ambientes no controlados, donde las condiciones de luz son variables y comúnmente no se emplean cámaras profesionales, lo que resulta en videos de baja resolución. Por tal motivo, en el presente trabajo se diseñó un algoritmo de CNN sencillo y ligero, capaz de clasificar imágenes de rostros en baja resolución en dos clases: Con mascarillas y Sin mascarilla.

Se elaboró un conjunto de datos denominado como “Low-Res Faces in Peru” (LR-FP) y se aplicaron las imágenes al modelo de CNN diseñado por los autores. Los resultados obtenidos fueron aceptables, se alcanzó un accuracy de 99.10% que superó a las arquitecturas que utilizaron transfer learning. Esto demuestra la capacidad de las redes neuronales convolucionales para clasificar imágenes en baja resolución a pesar de la baja profundidad de capas convolucionales y la limitada cantidad de imágenes de entrenamiento.

El presente documento está organizado de la siguiente manera: La sección 2 presenta la metodología de aprendizaje profundo utilizada, la sección 3 explica los experimentos realizados, la sección 4 muestra los resultados obtenidos y por último, en la sección 5 se presentan las conclusiones del trabajo de investigación.

2. Metodología

En esta sección, se detalla la metodología utilizada en este estudio, la cual se basa en gran medida en la investigación de Oumina et. al. [15] donde debido a escasa cantidad de datos utilizan estrategias de aumento de datos para incrementar la diversidad de los datos disponibles para los modelos de entrenamiento. Además de la comparación realizada entre 3 redes neuronales convolucionales profundas como selectores de características.

A. Elaboración del conjunto de datos

Para la construcción del conjunto de datos “Low-Res Faces in Peru” se seleccionaron videos de la plataforma Youtube en los que aparecían personas caminando en lugares públicos de la ciudad de Lima. De los videos obtenidos se tomaron capturas de los rostros de las personas, con y sin mascarillas. El requisito para el trabajo fue que los videos tuvieran una resolución menor a 720p, lo que se consideró como baja resolución. Dentro de las imágenes destaca las escenas donde la iluminación es deficiente y la sombra oculta varias partes de los rostros de las personas, como se muestra en la Figura 1(A). También se consideraron rostros que se encontraban ocluidos por accesorios personales como lentes, gorros y protectores faciales, en la Figura 1(B). La Figura 1(C) muestra la variación de las imágenes en cuanto a la posición de los rostros y la mirada de las personas. Así mismo, se identificaron escenarios donde los rostros tenían desenfoque debido al movimiento constante de las personas y la deficiente calidad de la cámara.

Figura 1

Muestra de imágenes en el dataset

A B C

El conjunto de imágenes final contiene un total de 2000 imágenes donde 1000 pertenecen a la clase Con mascarilla y 1000 a la clase Sin Mascarilla.

B. Preprocesamiento de imágenes

Las imágenes del conjunto de datos poseen diferentes escalas en consecuencia del acercamiento de los rostros de las personas al dispositivo de grabación. Por esto las imágenes fueron redimensionadas a un tamaño específico de 224x224 píxeles con el fin de estandarizar la entrada de datos del modelo. Además, se aplicó un proceso de normalización para optimizar el reconocimiento de las imágenes en la red neuronal.

Debido a la baja cantidad de imágenes en el conjunto de datos, se aplicó técnicas de Aumento de Datos para generar nuevas imágenes de entrenamiento. Este proceso se llevó a cabo aplicando transformaciones a las imágenes originales, como rotaciones aleatorias y reflejos horizontales.

Todas las imágenes del conjunto de datos están en formato PNG y poseen el espacio de color RGB.

C. Fase 1: Diseño del modelo de CNN

El desarrollo del modelo se dividió en 2 fases. En la primera fase se presentan tres arquitecturas de deep learning diseñadas por los autores, lo que implicó la definición de la estructura de las redes neuronales, la selección de las capas adecuadas y los hiperparámetros correspondientes. Estos modelos fueron entrenados con el conjunto de datos "Low-Res Faces in Peru" sin conocimientos previos, lo que significa que los pesos

de las redes se inicializaron aleatoriamente y se ajustaron iterativamente a través del algoritmo de optimización durante el proceso de entrenamiento. A continuación, se describen las características de cada modelo.

a) Modelo 1

La arquitectura del primer modelo consta de tres capas convolucionales. Se aplicó Max Pooling a cada capa convolucional. Mediante este proceso se disminuye el tamaño de la representación y a su vez se reduce el número de parámetros. Por consecuencia, el cálculo es simplificado para la red. Después de la segunda capa de convolución se encuentra la capa de Batch Normalization que normaliza los datos antes de que pasen por la función de activación. Finalmente, se aplica el proceso de aplanamiento para obtener el vector que ingresará a la red neuronal totalmente conectada. La arquitectura descrita se ilustra en la Tabla 1.

Tabla 1

Red Neuronal Convolucional 1

Layer	Type	Kernel size
1	Convolutional2D	5
2	MaxPooling2D	2
3	Convolutional2D	3
4	BatchNormalization2D	-
5	MaxPooling2D	2
6	Convolutional2D	3
7	MaxPooling2D	2
8	Flatten	-
9	FullyConnected	-
10	FullyConnected	-

b) Modelo 2

Se aumentó la profundidad del primer modelo con dos capas convolucionales adicionales, lo que resultó en una arquitectura de 5 capas convolucionales. Se aplicó Max Pooling a cada capa convolucional y se añadió Batch Normalization cada 2 capas convolucionales. La arquitectura se puede visualizar en la Tabla 2.

Tabla 2

Red Neuronal Convolucional 2

Layer	Type	Kernel size
1	Convolutional2D	5
2	MaxPooling2D	2
3	Convolutional2D	3
4	BatchNormalization2D	-
5	MaxPooling2D	2
6	Convolutional2D	3
7	MaxPooling2D	2
8	Convolutional2D	3
9	BatchNormalization2D	-
10	MaxPooling2D	2
11	Convolutional2D	3
12	MaxPooling2D	2
13	Flatten	-
14	FullyConnected	-
15	FullyConnected	-

c) Modelo 3

El tercer modelo consta de 3 capas convolucionales y 3 capas totalmente conectadas. A diferencia de los modelos anteriores, la dimensión de la imagen de entrada se redujo a 64x64 píxeles y las capas convolucionales utilizan un tamaño de kernel más pequeño. En la parte final del modelo se encuentran tres capas totalmente conectadas que utilizan la información de las características extraídas para realizar la clasificación de la imagen en dos clases (Con mascarillas y Sin mascarilla). En la Tabla 3 se presenta la arquitectura del modelo.

Tabla 3

Red Neuronal Convolucional 3

Layer	Type	Kernel size
1	Convolutional2D	3
2	MaxPooling2D	2
3	Convolutional2D	2
4	MaxPooling2D	2
5	Convolutional2D	2
6	MaxPooling2D	2
7	Flatten	-
8	FullyConnected	-
9	FullyConnected	-
10	FullyConnected	-

D. Fase 2: Transfer Learning

Para la segunda fase se utilizó la técnica de transfer learning en tres arquitecturas de redes neuronales convolucionales ampliamente reconocidas: VGG-16, ResNet18 y AlexNet. El objetivo principal fue aprovechar el conocimiento previo de estas arquitecturas pre-entrenadas en conjuntos de datos masivos y aplicarlo a la clasificación del uso de mascarillas faciales en imágenes de baja resolución. Para ello, se tomó la estructura de cada modelo, se modificó la capa final con el fin de obtener una salida con dos clases: Con Mascarillas y Sin Mascarilla. Por último, se entrenaron estas redes adaptadas utilizando el conjunto de datos "Low-Res Faces in Peru" y se ajustaron los pesos de las capas previas mediante técnicas de backpropagation.

A continuación, se presentan las tres arquitecturas utilizadas en esta fase:

- VGG-16: Es una arquitectura de red neuronal convolucional con 16 capas de profundidad. Utiliza filtros de convoluciones pequeñas (3x3) para extraer características de las imágenes y lograr una alta precisión en la clasificación de imágenes.
- Resnet18: Es una red neuronal convolucional con 18 capas de profundidad que destaca por su eficiencia y rendimiento en tareas de clasificación de imágenes. Utiliza conexiones residuales para evitar el problema de la degradación del

rendimiento que ocurre cuando se agregan más capas a una red neuronal. Resnet18 es una arquitectura relativamente pequeña en comparación con otras redes neuronales profundas.

- AlexNet: Es una arquitectura de red neuronal convolucional que consta únicamente de ocho capas, incluyendo cinco capas de convolución y tres capas densas. AlexNet también utiliza la técnica de Dropout para prevenir el sobreajuste durante el entrenamiento. Es una de las primeras redes neuronales convolucionales profundas que demostró su capacidad para lograr una alta precisión en la clasificación de imágenes en el desafío ImageNet Large Scale Visual Recognition Competition.

E. Evaluación del modelo

Las imágenes del conjunto de datos se dividieron en 3 subconjuntos de datos por cada clase (Con mascarilla y Sin mascarilla) 1600 para entrenamiento, 80 para validación y 320 para prueba. El subconjunto de datos de validación fue utilizado durante el entrenamiento para supervisar el desempeño del modelo en cada época con imágenes que el modelo no conocía.

Al finalizar el entrenamiento, se utilizó el subconjunto de prueba para evaluar la capacidad del modelo en la clasificación del uso de mascarillas en imágenes de baja resolución. Después se organizaron los resultados en tablas de datos y se realizó una comparación entre los modelos diseñados por los autores y los modelos que utilizaron transfer learning. El accuracy fue la métrica principal para evaluar la capacidad de generalización de los modelos.

3. Experimentos

A. Fase 1

a) Experimento 1: Para el experimento 1 se utilizó el modelo de 3 capas convolucionales. El método de optimización fue la Gradiente Descendiente Estocástica (SGD) y la función de pérdida Cross Entropy. Además, se usó una taza de aprendizaje de 0.001 y el tamaño del lote de entrenamiento se estableció en 64.

El modelo fue entrenado con tres diferentes cantidades de épocas para encontrar los mejores resultados aumentando el tiempo de entrenamiento de forma progresiva.

Además, en esta fase se emplearon 3 diferentes técnicas de normalización de lotes: Batch Normalization tradicional, BatchGroup y BatchGroup con Weight Standardization. En la Tabla 4 se pueden visualizar los resultados del entrenamiento con 30, 50 y 80 épocas. En todos los casos se puede notar que el average loss (puntaje promedio de pérdida) así como el accuracy mejoran durante el transcurso de 30 a 50 épocas y empeora al llegar a 80 épocas, esto indicar la existencia de overfitting (sobreajuste) en el entrenamiento.

Como se muestra en la Tabla 4, el puntaje más alto de accuracy fue de 92,50% y se obtuvo en dos ocasiones, sin embargo, el average loss fue más bajo utilizando Batch Normalization con 50 épocas.

Tabla 4

Resultados del Modelo 1

		Nro. de épocas		
		30	50	80
BatchGroup	Accuracy	80,94%	85,31%	82,50%
	Avg. Loss	0,4308	0,3441	0,4821
BatchGroup & Weight	Accuracy	90,94%	92,50%	88,13%
Stardardization	Avg. Loss	0,2841	0,3385	0,3197
Batch Normal	Accuracy	91,56%	92,50%	89,38%
	Avg. Loss	0,4445	0,1953	0,4326

b) Experimento 2: En el segundo experimento, se utilizó el modelo de 5 capas convolucionales. Se utilizó el mismo método de optimización, función de pérdida, taza de aprendizaje y tamaño de lote del experimento anterior. Los resultados obtenidos se muestran en la Tabla 5. El modelo que implementó BatchGroup & Weight Standardization obtuvo el mejor accuracy con 95,00% en el conjunto de datos de prueba con 30 épocas de entrenamiento. Asimismo, también obtuvo el menor average loss con 0,3808.

Tabla 5

Resultados del Modelo 2

		Nro. de épocas		
		30	50	80
BatchGroup	Accuracy	89,69%	82,50%	89,38%
	Avg. Loss	0,5726	0,6371	0,6283
BatchGroup & Weight	Accuracy	95,00%	82,50%	82,19%
Stardardization	Avg. Loss	0,3808	0,5707	0,5024
Batch Normal	Accuracy	90,62%	91,88%	94,56%
	Avg. Loss	0,5617	0,5138	0,4063

c) Experimento 3: Para este experimento se utilizó el optimizador Adam y la función de pérdida Cross Entropy. Se mantuvieron los valores de la tasa de aprendizaje y el tamaño de lote. Otro cambio importante fue la reducción a 64x64 píxeles en la capa de entrada del modelo. En la Tabla 6 se presentan los resultados obtenidos después del entrenamiento en 30, 50 y 80 épocas, donde el mayor puntaje de accuracy (99,10%) se alcanzó con 30 y 50 épocas. No obstante, el average loss fue menor (0,0312) al utilizar 30 épocas. Se pueden observar mejores resultados en comparación a los modelos presentados anteriormente. Además, se observa que el incremento de épocas de entrenamiento no afectó considerablemente al puntaje de accuracy.

Tabla 6

Resultados del Modelo 3

	Nro. de épocas		
	30	50	80
Accuracy	99,10%	99,10%	98,40%
Avg. Loss	0,0312	0,0763	0,1288

B. Fase 2

a) Experimento 4: Para el entrenamiento de los 3 modelos de transfer learning, se utilizó SGD cómo método de optimización y Cross Entropy como función de pérdida. La taza de aprendizaje se estableció en 0.001 y el tamaño del lote de entrenamiento fue 64. En

la Tabla 7 se muestran los resultados de VGG-16 obtenidos en 30, 50 y 80 épocas de entrenamiento. Se observa que el puntaje de accuracy no aumentó significativamente a medida que se utilizaban más épocas, del mismo modo, el average loss no tuvo una reducción importante en las épocas más altas. El mejor puntaje de accuracy alcanzado fue 90.94% con 80 épocas de entrenamiento y el average loss obtenido fue 0,3940.

Tabla 7

Resultados del VGG-16

		Nro. de épocas		
		30	50	80
VGG-16	Accuracy	90,31%	90,31%	90,94%
	Avg. Loss	0,3880	0,3829	0,3940

Por otro lado, Resnet18 no obtuvo puntajes muy altos de accuracy respecto a los otros 2 modelos de transfer learning. En la Tabla 8 se puede apreciar que el puntaje de accuracy disminuye a medida que aumentan las épocas, esto se debe a la existencia de overfitting en el proceso de entrenamiento. El mejor resultado se obtuvo con 30 épocas llegando a 89,39 % de accuracy.

Tabla 8

Resultados del Resnet18

		Nro. de épocas		
		30	50	80
Resnet18	Accuracy	89,38%	80,00%	82,81%
	Avg. Loss	0,4419	0,5204	0,5114

Por último, el modelo que utilizó la arquitectura de AlexNet alcanzó un accuracy de 95.31 %, el puntaje más alto entre los modelos de transfer learning. En la Tabla 9 se pueden visualizar los resultados obtenidos en 30, 50 y 80 épocas de entrenamiento. Al igual que Resnet18 los puntajes disminuyeron cuando la cantidad de épocas aumentaba, por lo que fueron necesarias sólo 30 épocas para conseguir el mejor resultado.

Tabla 9

Resultados del AlexNet

		Nro. de épocas		
		30	50	80
AlexNet	Accuracy	95,31%	94,38%	93,44%
	Avg. Loss	0,1674	0,1752	0,1708

4. Resultados

De los experimentos realizados en ambas fases, el mejor resultado fue obtenido por el modelo del experimento 3 utilizando el optimizador Adam, la función de pérdida Cross Entropy y 30 épocas de entrenamiento. El puntaje alcanzado fue de 99,10% en accuracy y 0,0312 de average loss.

En la fase 2, el puntaje máximo en accuracy (95,31%) lo obtuvo la arquitectura AlexNet utilizando transfer learning. Esto podría deberse a la estructura más ligera y con menos cantidad de parámetros y capas que posee AlexNet, lo que resulta beneficioso cuando se trabaja con cantidades reducidas de datos como imágenes en baja resolución. No obstante, el modelo diseñado en el experimento 2 con 5 capas convolucionales utilizando BatchGroup & Weight Stardardization tuvo un desempeño similar con 95,00% de accuracy. Aún así, el puntaje de accuracy del experimento 3 fue notablemente mayor con una diferencia de aproximadamente 4 puntos.

5. Discusiones

Se demostró que las arquitecturas que utilizaron transfer learning no fueron muy eficaces para la clasificación del uso de mascarillas en imágenes de rostros en baja resolución, en cambio, una arquitectura mucho más simple, ligera y con menos profundidad alcanzó mejores resultados en cuanto a accuracy y average loss. Es probable que el conocimiento adquirido previamente por el modelo de transfer learning no haya sido efectivo al aplicarse en imágenes de baja calidad. Así mismo, los modelos que poseen una mayor profundidad de capas convolucionales requieren extraer una gran cantidad parámetros de las imágenes, y si las mismas tienen menos información por estar en baja resolución, podría afectar el aprendizaje causando overfitting.

A comparación de otros trabajos, donde es común encontrar más de cinco mil o diez mil imágenes en el conjunto de datos, la limitada cantidad de 800 imágenes de entrenamiento por clase que se utilizó en este trabajo no impidió que el modelo pueda generalizar la información y obtener altos puntajes de accuracy en la etapa de evaluación.

Este trabajo es una base para futuros proyectos de investigación en universidades, también puede servir como antecedente para futuras tesis. El trabajo presenta un aporte para construir una solución de detección de rostros con y sin mascarilla, que podría ser implementada masivamente en lugares públicos u organizaciones a fin de monitorear el uso de mascarillas utilizando cámaras económicas y módulos con capacidad computacional reducida.

La propuesta de los autores, como siguiente trabajo de investigación, es realizar la detección de rostros en tiempo real en imágenes de baja resolución.

6. Conclusiones

Se realizaron varios experimentos con modelos de redes neuronales convolucionales para determinar su capacidad en la clasificación del uso de mascarillas en imágenes de rostros en baja resolución. Se elaboró el conjunto de datos “Low-Res Faces in Peru” con 2000 imágenes extraídas de videos en baja resolución. Del conjunto de datos se reservó un subconjunto de imágenes para la evaluación de los modelos de CNN, donde el modelo propuesto con 3 capas convolucionales y 3 capas densas superó a los modelos que utilizaron transfer learning, alcanzando 99,10% de accuracy y 0,0312 de average loss.

El trabajo demostró que las redes neuronales convolucionales con una arquitectura sencilla, es decir, baja profundidad de capas, pueden hacer una clasificación más que aceptable de imágenes en baja resolución en dos clases, con mascarilla y sin mascarilla. Aun considerando que los primeros modelos diseñados utilizaban diferentes técnicas de normalización de lotes y tenían una complejidad mayor.

7. Referencias

WHO Director-General's opening remarks at the media briefing on COVID-19—11 March 2020. (s/f). Recuperado el 8 de diciembre de 2021, de https://www.who.int/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020

Johns Hopkins University. (s/f). *COVID-19 Dashboard by the Center for Systems Science and Engineering at Johns Hopkins University*. Johns Hopkins Coronavirus Resource Center. Recuperado el 17 de junio de 2021, de https://coronavirus.jhu.edu/map.html

Rosas, Y. (2021, abril 23). *COVID-19 en transporte público: Identifican más de mil paraderos donde hay alto nivel de contagio*. El Comercio. https://elcomercio.pe/lima/transporte/covid-19-en-transporte-publico-identifican-mas-de-mil-paraderos-en-donde-hay-alto-nivel-de-contagio-atu-coronavirus-en-peru-noticia/

Advice for the public on COVID-19 – World Health Organization. (s/f). Recuperado el 26 de abril de 2023, de https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public

Instituto Nacional de Salud y Centro Nacional de Epidemiología, Prevención y Control de Enfermedades-MINSA. (s/f). *Covid 19 en el Perú—Ministerio de Salud*. Recuperado el 26 de abril de 2023, de https://covid19.minsa.gob.pe/sala_situacional.asp

Milton, D. K., Fabian, M. P., Cowling, B. J., Grantham, M. L., & McDevitt, J. J. (2013). *Influenza Virus Aerosols in Human Exhaled Breath: Particle Size, Culturability, and Effect of Surgical Masks*. PLOS Pathogens, 9(3), e1003205. https://doi.org/10.1371/journal.ppat.1003205

Lai, A. C. K., Poon, C. K. M., & Cheung, A. C. T. (2012). *Effectiveness of facemasks to reduce exposure hazards for airborne infections among general populations*. Journal of The Royal Society Interface, 9(70), 938–948. https://doi.org/10.1098/rsif.2011.0537

Garcia, L. P. (2020). *Uso de máscara facial para limitar a transmissão da COVID-19*. Epidemiologia e Serviços de Saúde, 29. https://doi.org/10.5123/S1679-49742020000200021

Eikenberry, S. E., Mancuso, M., Iboi, E., Phan, T., Eikenberry, K., Kuang, Y., Kostelich, E., & Gumel, A. B. (2020). *To mask or not to mask: Modeling the potential for face mask use by the general public to curtail the COVID-19 pandemic*. Infectious Disease Modelling, 5, 293–308. https://doi.org/10.1016/j.idm.2020.04.001

CS231n Convolutional Neural Networks for Visual Recognition. (s/f). Recuperado el 26 de abril de 2023, de https://cs231n.github.io/transfer-learning/

Singh, S., Ahuja, U., Kumar, M., Kumar, K., & Sachdeva, M. (2021). *Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment*. Multimedia Tools and Applications, 80(13), 19753–19768. https://doi.org/10.1007/s11042-021-10711-8

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. https://doi.org/10.1016/j.measurement.2020.108288

K. Suresh, M. Palangappa, & S. Bhuvan. (2021). *Face Mask Detection by using Optimistic Convolutional Neural Network*. 2021 6th International Conference on Inventive Computation Technologies (ICICT), 1084–1089. https://doi.org/10.1109/ICICT50816.2021.9358653

M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud, & J. -H. Kim. (2020). *An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network*. 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 1–5. https://doi.org/10.1109/IEMTRONICS51293.2020.9216386

A. Oumina, N. El Makhfi, & M. Hamdi. (2020). *Control The COVID-19 Pandemic: Face Mask Detection Using Transfer Learning*. 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS), 1–5. https://doi.org/10.1109/ICECOCS50124.2020.9314511

Real-World Masked Face Dataset. (s/f). [Open Source]. Recuperado el 19 de diciembre de 2022, de https://github.com/X-zhangyang/Real-World-Masked-Face-Dataset

Simulated Masked Face Dataset. (s/f). [Open Source]. Recuperado el 19 de diciembre de 2022, de https://github.com/prajnasb/observations

8. Anexos

A. Evidencias de sumisión de la revista

28/09/23, 22:13

Machine Vision and Applications - Receipt of Manuscript 'Deep Learning Model...'

Machine Vision and Applications <swathi.venkatesan@springernature.com>
Jue 28/09/2023 21:49
Para:Hebert Coazaca <hebert.coazaca@upeu.edu.pe>
Ref: Submission ID 56867e07-e892-440c-af38-95ef943d0cfa

Dear Dr Coazaca,

Thank you for submitting your manuscript to Machine Vision and Applications.

Your manuscript is now at our initial Technical Check stage, where we look for adherence to the journal's submission guidelines, including any relevant editorial and publishing policies. If there are any points that need to be addressed prior to progressing we will send you a detailed email. Otherwise, your manuscript will proceed into peer review.

You can check on the status of your submission at any time by using the link below and logging in with the account you created for this submission:

https://researcher.nature.com/your-submissions?utm_source=submissions&utm_medium=email&utm_campaign=confirmation-email&journal_id=138

Kind regards,

Editorial Assistant
Machine Vision and Applications

Machine Vision and Applications is a transformative journal. This means it offers a hybrid publication model. When the journal accepts research for publication, the article may be published using either immediate gold open access or the subscription publishing route. For further information please visit https://www.springernature.com/gp/open-research/transformative-journals

B. Resolución de inscripción del perfil de proyecto de tesis aprobado por el Consejo de facultad

RESOLUCIÓN N° 0344-2022/UPeU-FIA-CF-T

Lima, Ñaña 26 de abril de 2022

VISTO:

El expediente de **Coazaca Bustamante, Hebert Yamil**, identificado(a) con Código Universitario N° 201710529 y **Ali Vilca Pandely Sabina**, identificado(a) con Código Universitario N° 201710534, de la Escuela Profesional de Ingeniería de Sistemas de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión;

CONSIDERANDO

Que la Universidad Peruana Unión tiene autonomía académica, administrativa y normativa, dentro del ámbito establecido por la Ley Universitaria N° 30220 y el Estatuto de la Universidad;

Que la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión, mediante sus reglamentos académicos y administrativos, ha establecido las formas y procedimientos para la aprobación e inscripción del perfil de proyecto de tesis en formato artículo y la designación o nombramiento del asesor para la obtención del título profesional;

Que **Coazaca Bustamante, Hebert Yamil** y **Ali Vilca Pandely Sabina**, han solicitado: la inscripción del perfil de proyecto de tesis titulado "Modelo de CNN utilizando Transfer Learning para la clasificación del uso de mascarillas faciales en imágenes de baja calidad tomadas en espacios públicos del Perú" y la designación del Asesor, encargado de orientar y asesorar la ejecución del perfil de proyecto de tesis en formato artículo;

Estando a lo acordado en la sesión del Consejo de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión, celebrada el 26 de abril de 2022, y en aplicación del Estatuto y el Reglamento General de Investigación de la Universidad;

SE RESUELVE:

Aprobar el perfil de proyecto de tesis en formato artículo titulado **"Modelo de CNN utilizando Transfer Learning para la clasificación del uso de mascarillas faciales en imágenes de baja calidad tomadas en espacios públicos del Perú"** y disponer su inscripción en el registro correspondiente, designar a **MSc. Pineda Ancco Ferdinand Edgardo** como ASESOR para que oriente y asesore la ejecución del perfil de proyecto de tesis en formato artículo el cual fue dictaminado por: **Mg. Sullon Macalupu Abel Angel** y **MSc. Huanca Torres Fredy Abel**, otorgándoles un plazo máximo de doce (12) meses para la ejecución.

Regístrese, comuníquese y archívese.

Dra. Erika Inés Acuña Salinas
DECANA

Dr. Santiago Ramírez López
SECRETARIO ACADÉMICO

cc:
-Interesado
Asesor
Dirección General de Investigación
Archivo

C. Código y Fuentes

a) Modelo VGG-16

```
# Definición de la arquitectura de la red neuronal VGG

# Capas convolucionales y funciones de activación ReLU
def features(input):
    conv1_1 = Conv2d(input, 64, kernel_size=(3, 3), padding=(1, 1))
    relu1_1 = ReLU(conv1_1)
    conv1_2 = Conv2d(relu1_1, 64, kernel_size=(3, 3), padding=(1, 1))
    relu1_2 = ReLU(conv1_2)
    pool1 = MaxPool2d(relu1_2, kernel_size=2, stride=2)

    conv2_1 = Conv2d(pool1, 128, kernel_size=(3, 3), padding=(1, 1))
    relu2_1 = ReLU(conv2_1)
    conv2_2 = Conv2d(relu2_1, 128, kernel_size=(3, 3), padding=(1, 1))
    relu2_2 = ReLU(conv2_2)
    pool2 = MaxPool2d(relu2_2, kernel_size=2, stride=2)

    conv3_1 = Conv2d(pool2, 256, kernel_size=(3, 3), padding=(1, 1))
    relu3_1 = ReLU(conv3_1)
    conv3_2 = Conv2d(relu3_1, 256, kernel_size=(3, 3), padding=(1, 1))
    relu3_2 = ReLU(conv3_2)
    conv3_3 = Conv2d(relu3_2, 256, kernel_size=(3, 3), padding=(1, 1))
    relu3_3 = ReLU(conv3_3)
    pool3 = MaxPool2d(relu3_3, kernel_size=2, stride=2)

    conv4_1 = Conv2d(pool3, 512, kernel_size=(3, 3), padding=(1, 1))
    relu4_1 = ReLU(conv4_1)
    conv4_2 = Conv2d(relu4_1, 512, kernel_size=(3, 3), padding=(1, 1))
    relu4_2 = ReLU(conv4_2)
    conv4_3 = Conv2d(relu4_2, 512, kernel_size=(3, 3), padding=(1, 1))
    relu4_3 = ReLU(conv4_3)
    pool4 = MaxPool2d(relu4_3, kernel_size=2, stride=2)

    conv5_1 = Conv2d(pool4, 512, kernel_size=(3, 3), padding=(1, 1))
    relu5_1 = ReLU(conv5_1)
    conv5_2 = Conv2d(relu5_1, 512, kernel_size=(3, 3), padding=(1, 1))
    relu5_2 = ReLU(conv5_2)
    conv5_3 = Conv2d(relu5_2, 512, kernel_size=(3, 3), padding=(1, 1))
    relu5_3 = ReLU(conv5_3)
    pool5 = MaxPool2d(relu5_3, kernel_size=2, stride=2)

    return pool5

# Capa de pooling adaptativo
def avgpool(input):
    return AdaptiveAvgPool2d(input, output_size=(7, 7))

# Capas completamente conectadas y funciones de activación ReLU y dropout
def classifier(input):
    flatten = Flatten(input)
    fc1 = Linear(flatten, 4096)
    relu1 = ReLU(fc1)
    dropout1 = Dropout(relu1, p=0.5)
    fc2 = Linear(dropout1, 4096)
```

```
    relu2 = ReLU(fc2)
    dropout2 = Dropout(relu2, p=0.5)
    fc3 = Linear(dropout2, 1000)
    return fc3

# Modelo completo de VGG
def VGG(input):
    features_output = features(input)
    avgpool_output = avgpool(features_output)
    classifier_output = classifier(avgpool_output)
    return classifier_output

# Utilizando la arquitectura definida
input_data = # Datos de entrada
output = VGG(input_data)
```

b) Modelo Restnet18

```
# Definición de la arquitectura de la red neuronal ResNet

# Capa convolucional inicial seguida de batch normalization, ReLU y max pooling
def conv1(input):
    conv1_output = Conv2d(input, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)
    bn1_output = BatchNorm2d(conv1_output, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    relu_output = ReLU(bn1_output)
    maxpool_output = MaxPool2d(relu_output, kernel_size=3, stride=2, padding=1)
    return maxpool_output

# Bloque básico de la capa residual
class BasicBlock:
    def __init__(self, in_channels, out_channels, stride=1):
        self.conv1 = Conv2d(in_channels, out_channels, kernel_size=(3, 3), stride=stride, padding=(1, 1), bias=False)
        self.bn1 = BatchNorm2d(out_channels, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        self.relu = ReLU()
        self.conv2 = Conv2d(out_channels, out_channels, kernel_size=(3, 3), stride=1, padding=(1, 1), bias=False)
        self.bn2 = BatchNorm2d(out_channels, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        self.stride = stride

    def forward(self, x):
        identity = x

        out = self.conv1(x)
        out = self.bn1(out)
        out = self.relu(out)

        out = self.conv2(out)
        out = self.bn2(out)
```

```
        if self.stride != 1:
            identity = Conv2d(identity, out.shape[1], kernel_size=(1, 1),
stride=self.stride, bias=False)
            identity = BatchNorm2d(identity, eps=1e-05, momentum=0.1,
affine=True, track_running_stats=True)

        out += identity
        out = self.relu(out)

        return out

# Capas de la red ResNet
class ResNet:
    def __init__(self):
        self.conv1 = conv1
        self.layer1 = Sequential(BasicBlock(64, 64), BasicBlock(64, 64))
        self.layer2    =    Sequential(BasicBlock(64,    128,    stride=2),
BasicBlock(128, 128))
        self.layer3    =    Sequential(BasicBlock(128,    256,    stride=2),
BasicBlock(256, 256))
        self.layer4    =    Sequential(BasicBlock(256,    512,    stride=2),
BasicBlock(512, 512))
        self.avgpool = AdaptiveAvgPool2d(output_size=(1, 1))
        self.fc = Linear(in_features=512, out_features=1000)

    def forward(self, x):
        x = self.conv1(x)
        x = self.layer1(x)
        x = self.layer2(x)
        x = self.layer3(x)
        x = self.layer4(x)
        x = self.avgpool(x)
        x = Flatten(x)
        x = self.fc(x)
        return x

# Utilizando la arquitectura definida
input_data = # Datos de entrada
resnet_model = ResNet()
output = resnet_model.forward(input_data)
```

c) Modelo Alexnet

```
# Definición de la arquitectura de la red neuronal AlexNet

# Capas convolucionales y funciones de activación ReLU
def features(input):
    conv1  =  Conv2d(input,  64,  kernel_size=(11,  11),  stride=(4,  4),
padding=(2, 2))
    relu1 = ReLU(conv1)
    pool1 = MaxPool2d(relu1, kernel_size=3, stride=2)

    conv2 = Conv2d(pool1, 192, kernel_size=(5, 5), stride=(1, 1), padding=(2,
2))
    relu2 = ReLU(conv2)
    pool2 = MaxPool2d(relu2, kernel_size=3, stride=2)
```

```
    conv3 = Conv2d(pool2, 384, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1))
    relu3 = ReLU(conv3)

    conv4 = Conv2d(relu3, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1))
    relu4 = ReLU(conv4)

    conv5 = Conv2d(relu4, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1,
1))
    relu5 = ReLU(conv5)
    pool5 = MaxPool2d(relu5, kernel_size=3, stride=2)

    return pool5

# Capa de pooling adaptativo
def avgpool(input):
    return AdaptiveAvgPool2d(input, output_size=(6, 6))

# Capas completamente conectadas y funciones de activación ReLU y dropout
def classifier(input):
    dropout1 = Dropout(input, p=0.5)
    fc1 = Linear(dropout1, 4096)
    relu1 = ReLU(fc1)
    dropout2 = Dropout(relu1, p=0.5)
    fc2 = Linear(dropout2, 4096)
    relu2 = ReLU(fc2)
    fc3 = Linear(relu2, 1000)
    return fc3

# Modelo completo de AlexNet
def AlexNet(input):
    features_output = features(input)
    avgpool_output = avgpool(features_output)
    classifier_output = classifier(avgpool_output)
    return classifier_output

# Utilizando la arquitectura definida
input_data = # Datos de entrada
output = AlexNet(input_data)
```

D. Base de datos (Low-Res Faces in Peru)

El conjunto de datos elaborado por los autores se encuentra en una carpeta de OneDrive, el cual será publicada abiertamente cuando el artículo sea publicado en una revista.

URL: https://upeuedupe-my.sharepoint.com/:f:/r/personal/hebert_coazaca_upeu_edu_pe/Documents/Low-Res%20Faces%20in%20Peru%20Dataset?csf=1&web=1&e=KUDkmx