PROYECTO DEEP LEARNING:

Clasificación de imágenes

DIEGO ALEXANDER BERNALES VALDIVIA

Data Science

Presentanción del caso: Conjunto de datos de imagen de tipo de vehículo (versión 1): VTID1

CONTEXTO

Los dispositivos de grabación para recopilar las imágenes formaban parte de un sistema de videovigilancia ubicado en la Universidad Loei Rajabhat en la provincia de Loei, Tailandia. El proceso de recolección se llevó a cabo durante el día durante cuatro semanas entre julio y diciembre de 2018. Se instalaron dos cámaras en la puerta principal de la universidad.

Data Science

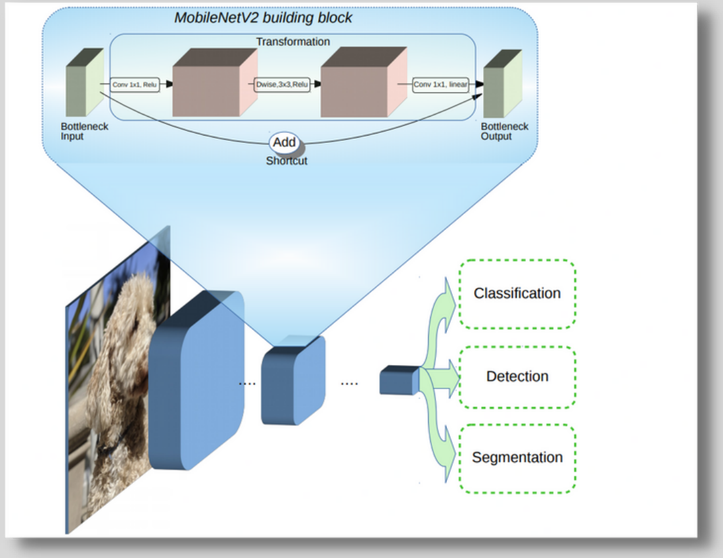
OBJETIVOS

El principal objetivo del uso de un conjunto de datos de imágenes fue examinar los cinco tipos de vehículos de motor que eran los más utilizados en Tailandia (sedán, hatchback, pick-up, SUV y motocicletas).

CONJUNTO DE DATOS

Conjunto de datos llamado "Conjunto de datos de imagen de tipo de vehículo (VTID)" tenía un total de 1410 imágenes que podían separarse en tipos de vehículos de la siguiente manera; 400 sedanes, 478 camionetas, 129 SUV, 181 hatchbacks y otras 122 imágenes de motocicletas. Cada imagen se recopiló utilizando la resolución de 224x224 píxeles.

MOBILENET V2



MobileNet-v2 es una red neuronal convolucional. Podemos cargar una versión preentrenada de la red entrenada en más de un millón de imágenes de la base de datos de ImageNet.

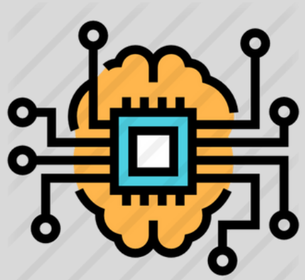
En conjunto, la arquitectura de MobileNetV2 contiene la capa inicial de convolución completa con 32 filtros, seguida de 19 capas de cuello de botella residuales.

La red preentrenada puede clasificar imágenes en 1000 categorías de objetos, como teclado, mouse, lápiz y muchos animales.

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Data Science

Procesos del proyecto



EXPLORACIÓN DE LA DATA

CONSTRUCCIÓN DE DATASETS

Dividir la data en 3 datasets: Train, Validation y Test.
Realizar data augmentation y normalización de imágenes

CONSTRUCCIÓN DEL MODELO

Usar transfer learning con Mobilenet-v2 y modificar los outputs del modelo

ENTRENAR EL MODELO

Entrenamiento de 100 épocas y validación con dataset validation.

EVALUAR EL MODELO Y ANALIZAR LAS PROBABILIDADES DE SALIDA

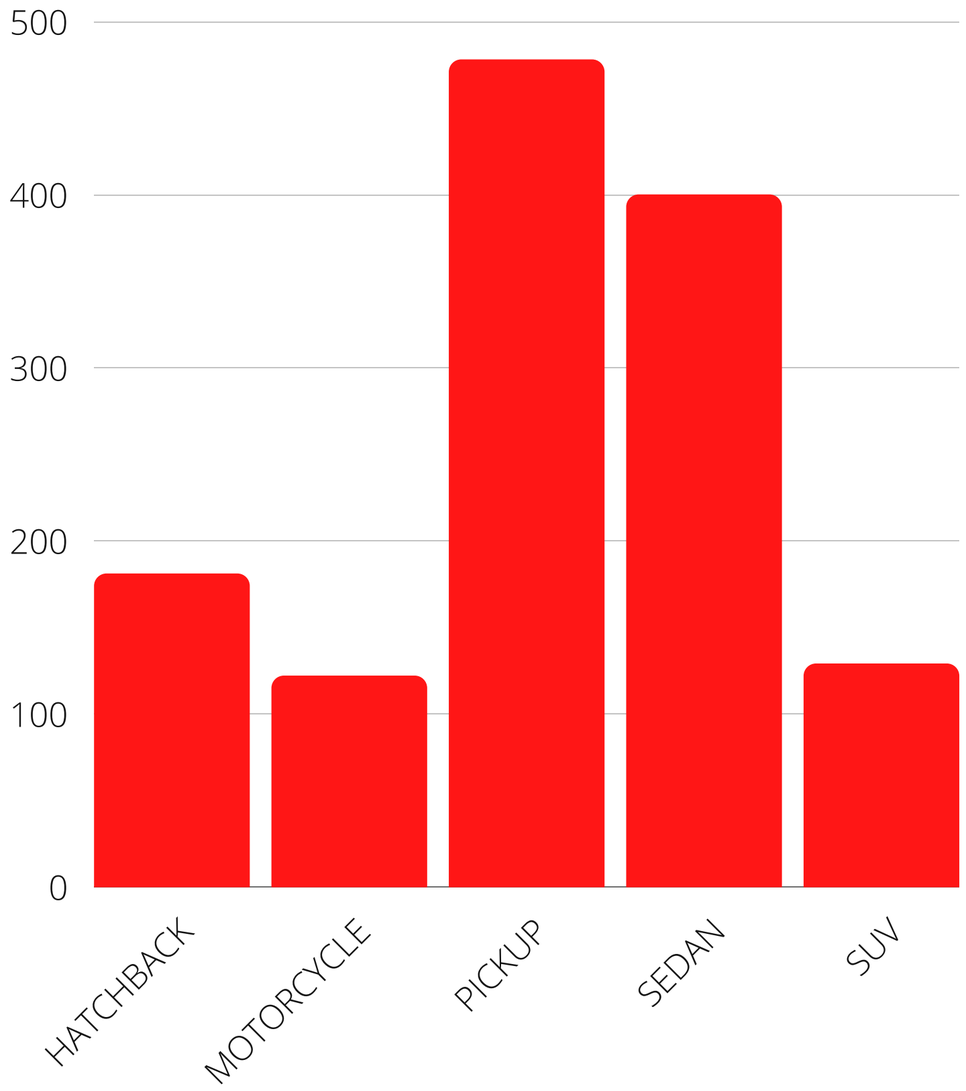
Utilizar el dataset Test para probar el modelo con data completamente nueva para el modelo.

Exploración de la data:

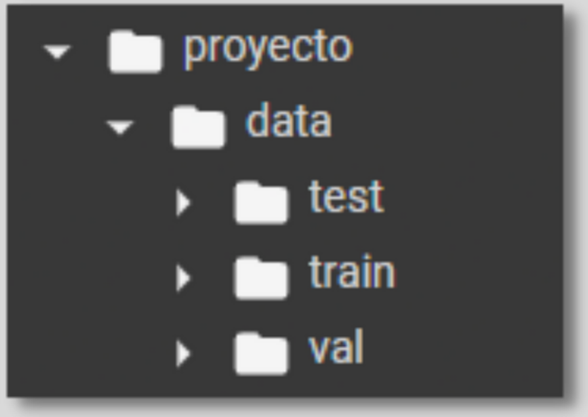
```
def imshow(img):
    img = img / 2 + 0.5
    plt.imshow(np.transpose(img, (1, 2, 0)))
dataiter = iter(data_loader)
images, labels = dataiter.next()
fig = plt.figure(figsize=(20, 4))
for idx in np.arange(20):
    ax = fig.add_subplot(2, 20/2, idx+1, xticks=[], yticks=[])
    imshow(images[idx])
    ax.set_title(classes[labels[idx]])
```


```
#Conteo de imagenes por cada clase
labels = {}
for label in classes:
  labels[label] = 0
data_loader = DataLoader(dataset, batch_size=1, shuffle=True)
for data in data_loader:
  img, label = data
  labels[classes[label.item()]] += 1
print(labels)
```

```
{'hatchback': 181, 'motorcycle': 122, 'pickup': 478, 'sedan': 400, 'suv': 129}
```



Construcción de Datasets:




```
[ ]  dataset_sizes = {d: len(image_datasets[d]) for d in DATASETS}
     class_names = image_datasets['train'].classes

     dataset_sizes

     {'test': 133, 'train': 1046, 'val': 131}

[ ]  class_names

     ['hatchback', 'motorcycle', 'pickup', 'sedan', 'suv']
```

DATA AUGMENTATION: para aumentar el tamaño del dataset train. Tambien pasaremos a tensor todas las imagenes y se le aplicará una normalización.

```
[ ]  mean_nums = [0.485, 0.456, 0.406]
     std_nums = [0.229, 0.224, 0.225]

     transforms = {'train': T.Compose([
       T.RandomResizedCrop(size=256),
       T.RandomRotation(degrees=15), # aplicamos rotationes de 15 grados
       T.RandomHorizontalFlip(),
       T.ToTensor(),
       T.Normalize(mean_nums, std_nums) #Normalizamos para obtener datos dentro de un rago y reducir la asimetría, esto ayuda
     ]), 'val': T.Compose([
       T.Resize(size=256),
       T.CenterCrop(size=224),
       T.ToTensor(),
       T.Normalize(mean_nums, std_nums)
     ]), 'test': T.Compose([
       T.Resize(size=256),
       T.CenterCrop(size=224),
       T.ToTensor(),
       T.Normalize(mean_nums, std_nums)
     ]),
     }
```

Data Science

Construcción del Modelo:

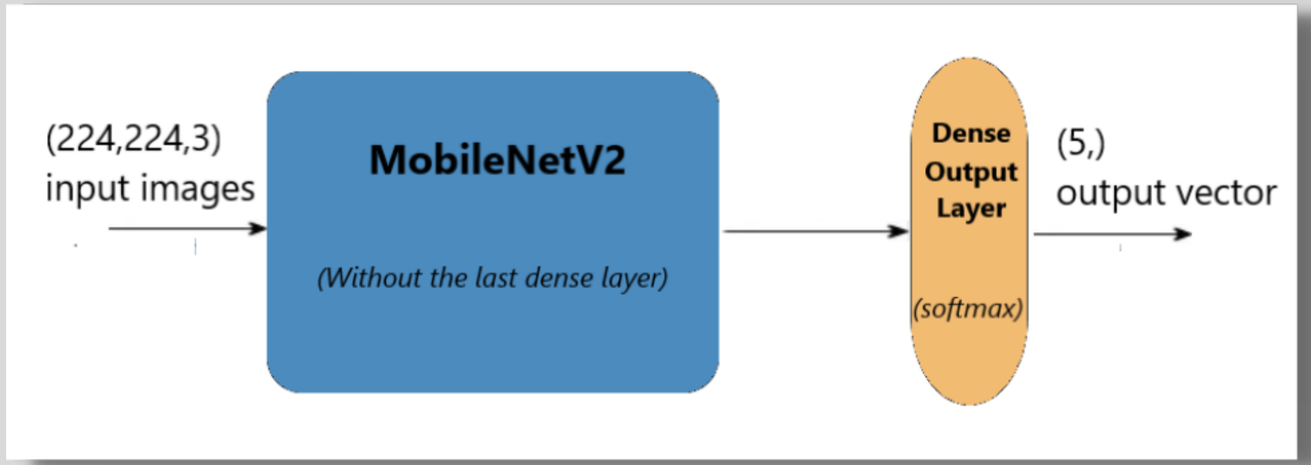
```python
def create_model(n_classes):
  model = models.mobilenet_v2(pretrained=True)

  num_ftrs = model.classifier[1].in_features
  model.classifier[1] = nn.Linear(num_ftrs, n_classes)

  return model.to(device)
```

```python
base_model = create_model(len(class_names))
```

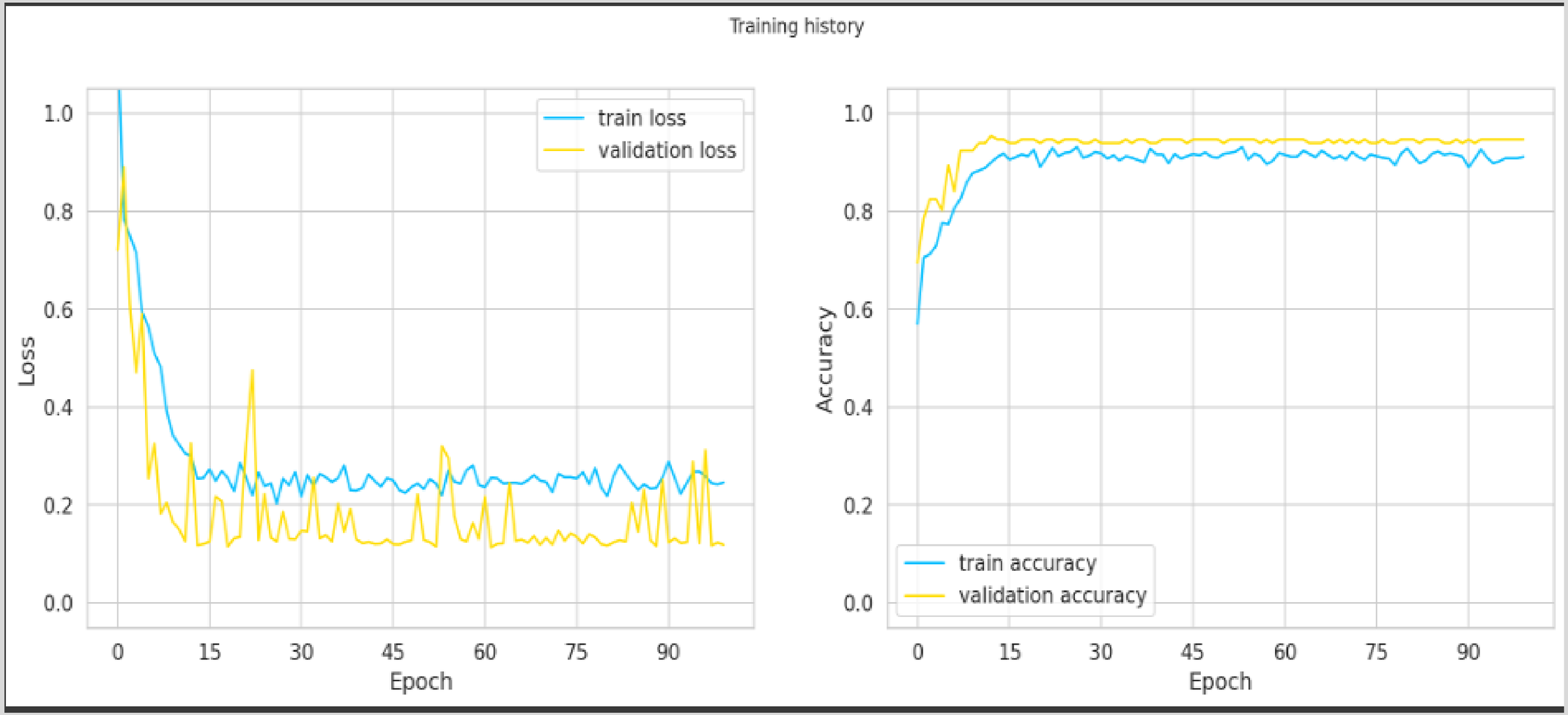
```python
print(base_model.classifier) # podemos ver  que solo se enfoca
```

```
Sequential(
  (0): Dropout(p=0.2, inplace=False)
  (1): Linear(in_features=1280, out_features=5, bias=True)
)
```

Transfer Learning:



Entrenamiento del Modelo:

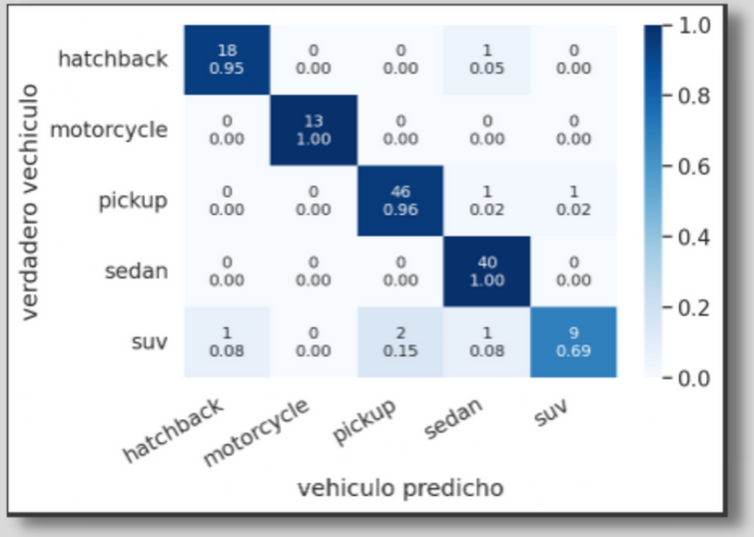


Evaluación del Modelo:

Reporte de clasificación:

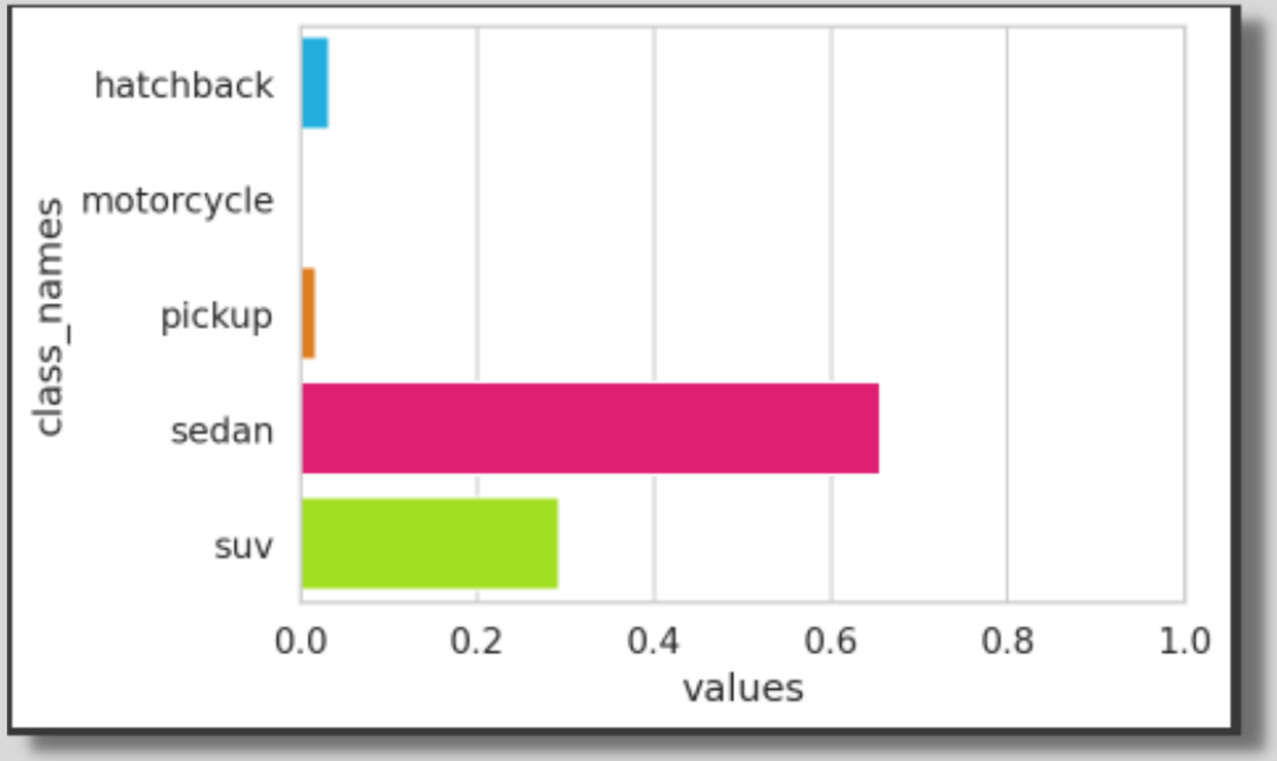
	precision	recall	f1-score	support
hatchback	0.95	0.95	0.95	19
motorcycle	1.00	1.00	1.00	13
pickup	0.96	0.96	0.96	48
sedan	0.93	1.00	0.96	40
suv	0.90	0.69	0.78	13
accuracy			0.95	133
macro avg	0.95	0.92	0.93	133
weighted avg	0.95	0.95	0.95	133

Matriz de Confusión:



Evaluación de probabilidades:

```
pred = predict_proba(base_model, '/content/drive/MyDrive/proyecto/data/test/sedan/PIC_106.jpg' )
pred
```



GRACIAS