

Aprendizaje Profundo Taller 2

Elkin Daniel Prada Gómez
elkind.prada@javeriana.edu.co

William Peña
walexanderpena@javeriana.edu.co

1 Resumen

En el siguiente trabajo se analizará la construcción de dos modelos basados en redes neuronales convolucionales para el procesamiento de imágenes de señales de tránsito, con el fin de poner en práctica los conceptos obtenidos en la asignatura, aprendizaje profundo y evaluar la diferencia entre 2 modelos con diferentes capas y parámetros de procesamiento. Para realizar este trabajo, se hace uso del dataset Traffic Sign de la página Kaggle, y se procede a realizar la construcción del modelo siguiendo las etapas de comprensión del dataset, construcción del dataset, elaboración del modelo, entrenamiento, ajustes y análisis de resultados. Las librerías a usar para la construcción de los modelos son, Keras, Tensorflow de Python. De igual manera, se realizará la construcción en la plataforma Colab para su desarrollo colaborativo.

2 Comprensión del dataset

Para iniciar con la comprensión del dataset, se realiza la descarga del dataset Traffic Sign Classification and Recognition de Kaggle, el cual tiene un peso de 26.96MB y contiene imágenes de prueba y validación sobre señales de tránsito.

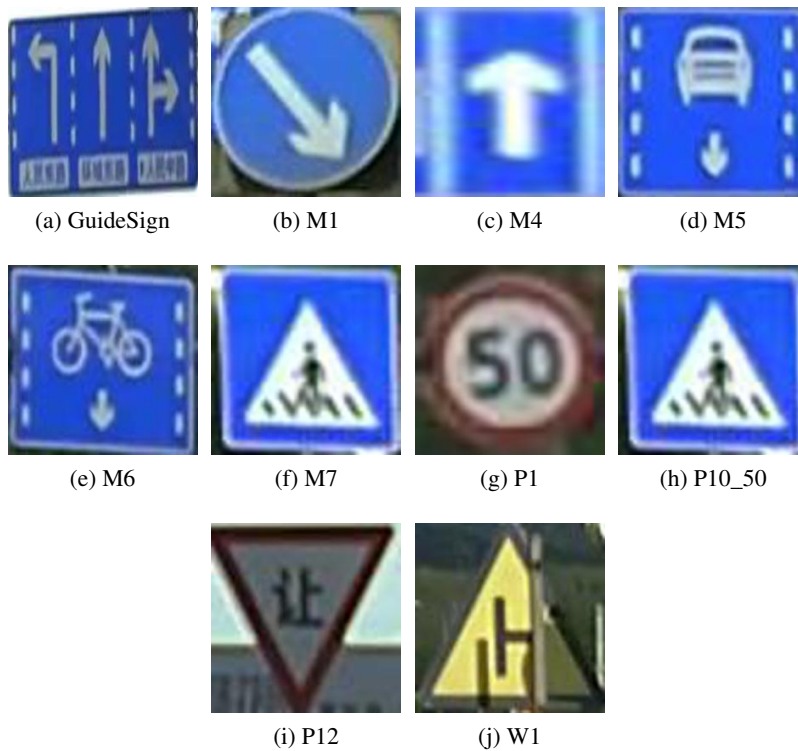
2.1 ¿Qué información presenta el dataset?

El conjunto de datos contiene 6348 etiquetas de categorías etiquetadas manualmente. Las etiquetas incluyen las siguientes 10 categorías correspondientes a diez clases diferentes de señales de tráfico.

Todos los datos han sido divididos manualmente en un conjunto de entrenamiento y un conjunto de pruebas de acuerdo con la proporción, y se proporciona el archivo json de anotación del conjunto de entrenamiento correspondiente. El conjunto de datos contiene dos carpetas de entrenamiento y prueba, así como un archivo train.json unificado para guardar las anotaciones de todos los conjuntos de datos de entrenamiento. Las etiquetas de imagen guardan el diccionario serializado en formato json, y todas las anotaciones se guardan en train.json bajo la clave llamada "annotations" en el archivo.

2.2 ¿Describir las características de las imágenes?

Las imágenes tienen un tamaño de 224 x 224 píxeles y vienen en formato JPG, estas imágenes vienen divididas en diferentes paquetes que llamaremos categorías o clases. Cada imagen tiene una particularidad distinta, ya que la calidad de la imagen cambia así como la posición de la misma, unas vienen centradas, otras no, incluso algunas vienen un poco más opacas o en una orientación distinta.



2.3 Indicar conjuntos de entrenamiento y pruebas

Al descargar el dataset, se puede evidenciar que son separadas las imágenes en dos paquetes, `train_dataset` para las imágenes de entrenamiento y `test_dataset` para las imágenes de prueba. Sumando la cantidad de las imágenes de ambas carpetas, dan un total de 6024 imágenes de entrenamiento y 324 imágenes de prueba. Sin embargo, no vienen en la proporción que indica el taller, ya que en este caso tenemos 95% imágenes de entrenamiento y 5% imágenes de prueba aproximadamente.

2.4 Indicar las clases de clasificación

El conjunto de datos, viene en 10 clases diferentes, las cuales son: "GuideSign", "M1", "M4", "M5", "M6", "M7", "P1", "P10_50", "P12", "W1". Como podemos observar a continuación:

```
In [53]: class_list = os.listdir("dataset/train_dataset/train")
         class_list

Out[53]: ['M6', 'W1', 'P10_50', 'P1', 'M7', 'GuideSign', 'P12', 'M4', 'M5', 'M1']
```

Figure 1: Se cuentan las imágenes de cada paquete

```
00135.jpg 00293.jpg 00450.jpg 00615.jpg 00771.jpg 00935.jpg 01096.jpg 01258.jpg 01416.jpg 01574.jpg 01739.jpg 01899.jpg 02065.jpg 02233.jpg 02397.jpg 02558.jpg 02711.jpg 02873.jpg 03032.jpg 03191.jpg 03354.jpg
00136.jpg 00294.jpg 00451.jpg 00616.jpg 00772.jpg 00936.jpg 01097.jpg 01257.jpg 01417.jpg 01575.jpg 01740.jpg 01900.jpg 02070.jpg 02234.jpg 02398.jpg 02559.jpg 02712.jpg 02874.jpg 03033.jpg 03192.jpg 03355.jpg
00137.jpg 00295.jpg 00452.jpg 00617.jpg 00773.jpg 00937.jpg 01098.jpg 01259.jpg 01418.jpg 01576.jpg 01741.jpg 01901.jpg 02071.jpg 02235.jpg 02399.jpg 02560.jpg 02713.jpg 02875.jpg 03034.jpg 03193.jpg 03356.jpg
00138.jpg 00296.jpg 00453.jpg 00618.jpg 00774.jpg 00938.jpg 01099.jpg 01260.jpg 01419.jpg 01577.jpg 01742.jpg 01902.jpg 02072.jpg 02236.jpg 02400.jpg 02561.jpg 02714.jpg 02876.jpg 03035.jpg 03194.jpg 03357.jpg
00139.jpg 00297.jpg 00454.jpg 00619.jpg 00775.jpg 00939.jpg 01100.jpg 01261.jpg 01420.jpg 01578.jpg 01743.jpg 01903.jpg 02073.jpg 02237.jpg 02401.jpg 02562.jpg 02715.jpg 02877.jpg 03036.jpg 03195.jpg 03358.jpg
00140.jpg 00298.jpg 00455.jpg 00620.jpg 00776.jpg 00940.jpg 01101.jpg 01262.jpg 01421.jpg 01579.jpg 01744.jpg 01904.jpg 02074.jpg 02238.jpg 02402.jpg 02563.jpg 02716.jpg 02878.jpg 03037.jpg 03196.jpg 03359.jpg
00141.jpg 00299.jpg 00456.jpg 00621.jpg 00777.jpg 00941.jpg 01102.jpg 01263.jpg 01422.jpg 01580.jpg 01745.jpg 01905.jpg 02075.jpg 02239.jpg 02403.jpg 02564.jpg 02717.jpg 02879.jpg 03038.jpg 03197.jpg 03360.jpg
00142.jpg 00300.jpg 00457.jpg 00622.jpg 00778.jpg 00942.jpg 01103.jpg 01264.jpg 01423.jpg 01581.jpg 01746.jpg 01906.jpg 02076.jpg 02240.jpg 02404.jpg 02565.jpg 02718.jpg 02880.jpg 03039.jpg 03198.jpg 03361.jpg
00143.jpg 00301.jpg 00458.jpg 00623.jpg 00779.jpg 00943.jpg 01104.jpg 01265.jpg 01424.jpg 01582.jpg 01747.jpg 01907.jpg 02077.jpg 02241.jpg 02405.jpg 02566.jpg 02719.jpg 02881.jpg 03040.jpg 03199.jpg 03362.jpg
00144.jpg 00302.jpg 00459.jpg 00624.jpg 00780.jpg 00944.jpg 01105.jpg 01266.jpg 01425.jpg 01583.jpg 01748.jpg 01908.jpg 02078.jpg 02242.jpg 02406.jpg 02567.jpg 02720.jpg 02882.jpg 03041.jpg 03200.jpg 03363.jpg
00145.jpg 00303.jpg 00460.jpg 00625.jpg 00781.jpg 00945.jpg 01106.jpg 01267.jpg 01426.jpg 01584.jpg 01749.jpg 01909.jpg 02079.jpg 02243.jpg 02407.jpg 02568.jpg 02721.jpg 02883.jpg 03042.jpg 03201.jpg 03364.jpg
00146.jpg 00304.jpg 00461.jpg 00626.jpg 00782.jpg 00946.jpg 01107.jpg 01268.jpg 01427.jpg 01585.jpg 01750.jpg 01910.jpg 02080.jpg 02244.jpg 02408.jpg 02569.jpg 02722.jpg 02884.jpg 03043.jpg 03202.jpg 03365.jpg
00147.jpg 00305.jpg 00462.jpg 00627.jpg 00783.jpg 00947.jpg 01108.jpg 01269.jpg 01428.jpg 01586.jpg 01751.jpg 01911.jpg 02081.jpg 02245.jpg 02409.jpg 02570.jpg 02723.jpg 02885.jpg 03044.jpg 03203.jpg 03366.jpg
00148.jpg 00306.jpg 00463.jpg 00628.jpg 00784.jpg 00948.jpg 01109.jpg 01270.jpg 01429.jpg 01587.jpg 01752.jpg 01912.jpg 02082.jpg 02246.jpg 02410.jpg 02571.jpg 02724.jpg 02886.jpg 03045.jpg 03204.jpg 03367.jpg
00149.jpg 00307.jpg 00464.jpg 00629.jpg 00785.jpg 00949.jpg 01110.jpg 01271.jpg 01430.jpg 01588.jpg 01753.jpg 01913.jpg 02083.jpg 02247.jpg 02411.jpg 02572.jpg 02725.jpg 02887.jpg 03046.jpg 03205.jpg 03368.jpg
00150.jpg 00308.jpg 00465.jpg 00630.jpg 00786.jpg 00950.jpg 01111.jpg 01272.jpg 01431.jpg 01589.jpg 01754.jpg 01914.jpg 02084.jpg 02248.jpg 02412.jpg 02573.jpg 02726.jpg 02888.jpg 03047.jpg 03206.jpg 03369.jpg
00151.jpg 00309.jpg 00466.jpg 00631.jpg 00787.jpg 00951.jpg 01112.jpg 01273.jpg 01432.jpg 01590.jpg 01755.jpg 01915.jpg 02085.jpg 02249.jpg 02413.jpg 02574.jpg 02727.jpg 02889.jpg 03048.jpg 03207.jpg 03370.jpg
00152.jpg 00310.jpg 00467.jpg 00632.jpg 00788.jpg 00952.jpg 01113.jpg 01274.jpg 01433.jpg 01591.jpg 01756.jpg 01916.jpg 02086.jpg 02250.jpg 02414.jpg 02575.jpg 02728.jpg 02890.jpg 03049.jpg 03208.jpg 03371.jpg
00153.jpg 00311.jpg 00468.jpg 00633.jpg 00789.jpg 00953.jpg 01114.jpg 01275.jpg 01434.jpg 01592.jpg 01757.jpg 01917.jpg 02087.jpg 02251.jpg 02415.jpg 02576.jpg 02729.jpg 02891.jpg 03050.jpg 03209.jpg 03372.jpg
00154.jpg 00312.jpg 00469.jpg 00634.jpg 00790.jpg 00954.jpg 01115.jpg 01276.jpg 01435.jpg 01593.jpg 01758.jpg 01918.jpg 02088.jpg 02252.jpg 02416.jpg 02577.jpg 02730.jpg 02892.jpg 03051.jpg 03210.jpg 03373.jpg
00155.jpg 00313.jpg 00470.jpg 00635.jpg 00791.jpg 00955.jpg 01116.jpg 01277.jpg 01436.jpg 01594.jpg 01759.jpg 01919.jpg 02089.jpg 02253.jpg 02417.jpg 02578.jpg 02731.jpg 02893.jpg 03052.jpg 03211.jpg 03374.jpg
00156.jpg 00314.jpg 00471.jpg 00636.jpg 00792.jpg 00956.jpg 01117.jpg 01278.jpg 01437.jpg 01595.jpg 01760.jpg 01920.jpg 02090.jpg 02254.jpg 02418.jpg 02579.jpg 02732.jpg 02894.jpg 03053.jpg 03212.jpg 03375.jpg
00157.jpg 00315.jpg 00472.jpg 00637.jpg 00793.jpg 00957.jpg 01118.jpg 01279.jpg 01438.jpg 01596.jpg 01761.jpg 01921.jpg 02091.jpg 02255.jpg 02419.jpg 02580.jpg 02733.jpg 02895.jpg 03054.jpg 03213.jpg 03376.jpg
00158.jpg 00316.jpg 00473.jpg 00638.jpg 00794.jpg 00958.jpg 01119.jpg 01280.jpg 01439.jpg 01597.jpg 01762.jpg 01922.jpg 02092.jpg 02256.jpg 02420.jpg 02581.jpg 02734.jpg 02896.jpg 03055.jpg 03214.jpg 03377.jpg
00159.jpg 00317.jpg 00474.jpg 00639.jpg 00795.jpg 00959.jpg 01120.jpg 01281.jpg 01440.jpg 01598.jpg 01763.jpg 01923.jpg 02093.jpg 02257.jpg 02421.jpg 02582.jpg 02735.jpg 02897.jpg 03056.jpg 03215.jpg 03378.jpg
00160.jpg 00318.jpg 00475.jpg 00640.jpg 00796.jpg 00960.jpg 01121.jpg 01282.jpg 01441.jpg 01599.jpg 01764.jpg 01924.jpg 02094.jpg 02258.jpg 02422.jpg 02583.jpg 02736.jpg 02898.jpg 03057.jpg 03216.jpg 03379.jpg
00161.jpg 00319.jpg 00476.jpg 00641.jpg 00797.jpg 00961.jpg 01122.jpg 01283.jpg 01442.jpg 01600.jpg 01765.jpg 01925.jpg 02095.jpg 02259.jpg 02423.jpg 02584.jpg 02737.jpg 02899.jpg 03058.jpg 03217.jpg 03380.jpg
00162.jpg 00320.jpg 00477.jpg 00642.jpg 00798.jpg 00962.jpg 01123.jpg 01284.jpg 01443.jpg 01601.jpg 01766.jpg 01926.jpg 02096.jpg 02260.jpg 02424.jpg 02585.jpg 02738.jpg 02900.jpg 03059.jpg 03218.jpg 03381.jpg
00163.jpg 00321.jpg 00478.jpg 00643.jpg 00799.jpg 00963.jpg 01124.jpg 01285.jpg 01444.jpg 01602.jpg 01767.jpg 01927.jpg 02097.jpg 02261.jpg 02425.jpg 02586.jpg 02739.jpg 02901.jpg 03060.jpg 03219.jpg 03382.jpg
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M45 ls | wc -l
1206
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M45 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M5 cd M5/
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M55 ls | wc -l
113
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M55 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M6 cd M6/
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M65 ls | wc -l
134
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M65 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M7 cd M7/
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M75 ls | wc -l
469
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/M75 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P15 cd P1
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P15 ls | wc -l
249
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P15 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P10_50 cd P10_50/
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P10_505 ls | wc -l
95
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P10_505 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P12 cd P12/
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P125 ls | wc -l
95
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/P125 cd ..
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/W1 cd W1/
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/W15 ls | wc -l
145
pr4d4pr4d4-GF63-Thin-185C-~/Desktop/taller2/dataset/train_dataset/train/W15
```

| Clases o Categorías | | | |
|---------------------|---------------|---------|-------|
| Categoría | Entrenamiento | Pruebas | Total |
| GuideSign | 1171 | 62 | 1233 |
| M1 | 247 | 14 | 261 |
| M4 | 3206 | 169 | 3375 |
| M5 | 213 | 12 | 225 |
| M6 | 134 | 8 | 142 |
| M7 | 469 | 25 | 494 |
| P1 | 249 | 14 | 263 |
| P10_50 | 95 | 6 | 101 |
| P12 | 95 | 6 | 101 |
| W1 | 145 | 8 | 153 |

3 Construcción del dataset

Para construir el dataset conforme a lo especificado en el taller, es necesario distribuir las imágenes de forma manual, para que sea un 80% de imágenes de entrenamiento y un 20% de imágenes de prueba. Se hace uso de la siguiente serie de comandos de linux para mover las imágenes en las carpetas correspondientes y así cambiar el dataset por 80% - 20% teniendo en cuenta la proporción de distribución de imágenes para que por cada clase se tenga el mismo porcentaje de imágenes respecto a las demás y a todo el dataset. De esta manera, el dataset queda construido de la siguiente forma:

Figure 2: Comandos para mover las imagenes

```

1  ls | wc -l
2  find train_dataset/train/GuideSign/ -maxdepth 1 -type f | head -185 | xargs -I{} mv {} test_dataset/test/GuideSign/
3  find train_dataset/train/M1/ -maxdepth 1 -type f | head -38 | xargs -I{} mv {} test_dataset/test/M1/
4  find train_dataset/train/M4/ -maxdepth 1 -type f | head -506 | xargs -I{} mv {} test_dataset/test/M4/
5  find train_dataset/train/M5/ -maxdepth 1 -type f | head -33 | xargs -I{} mv {} test_dataset/test/M5/
6  find train_dataset/train/M6/ -maxdepth 1 -type f | head -20 | xargs -I{} mv {} test_dataset/test/M6/
7  find train_dataset/train/M7/ -maxdepth 1 -type f | head -74 | xargs -I{} mv {} test_dataset/test/M7/
8  find train_dataset/train/P1/ -maxdepth 1 -type f | head -39 | xargs -I{} mv {} test_dataset/test/P1/
9  find train_dataset/train/P10_50/ -maxdepth 1 -type f | head -14 | xargs -I{} mv {} test_dataset/test/P10_50/
10 find train_dataset/train/P12/ -maxdepth 1 -type f | head -14 | xargs -I{} mv {} test_dataset/test/P12/
11 find train_dataset/train/W1/ -maxdepth 1 -type f | head -23 | xargs -I{} mv {} test_dataset/test/W1/

```

| Clases o Categorías | | | |
|---------------------|---------------|---------|-------|
| Categoría | Entrenamiento | Pruebas | Total |
| GuideSign | 986 | 247 | 1233 |
| M1 | 209 | 52 | 261 |
| M4 | 2700 | 675 | 3375 |
| M5 | 180 | 45 | 225 |
| M6 | 114 | 28 | 142 |
| M7 | 395 | 99 | 494 |
| P1 | 210 | 53 | 263 |
| P10_50 | 81 | 20 | 101 |
| P12 | 81 | 20 | 101 |
| W1 | 122 | 31 | 153 |

4 Elaboración del modelo

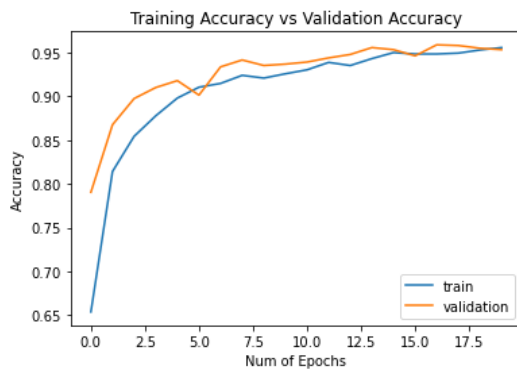
El modelo se elabora con una Red Neuronal Convolutacional. En este caso, una vez se extraen las características y se seleccionan las clases con las que se desea trabajar, se procede a construir el modelo e iniciar un entrenamiento de clasificación y de pruebas. En este caso, para el modelo 1, se tienen en cuenta las siguientes características de la arquitectura, que serán datos que usaremos como parámetros al construir la red neuronal apoyados por las librerías de Python.

- Capa convolucional con 32 detectores de características, tamaño de kernel de 5 x 5 y una función de activación lineal rectificada (ReLU) de paso 1 con padding.
- Capa de max pooling de tamaño 5x5
- Capa de Flattening
- Capa full conectada con 100 neuronas y función de activación lineal rectificada (ReLU).
- Capa de salida del mismo número de las clases a clasificar y función de activación softmax

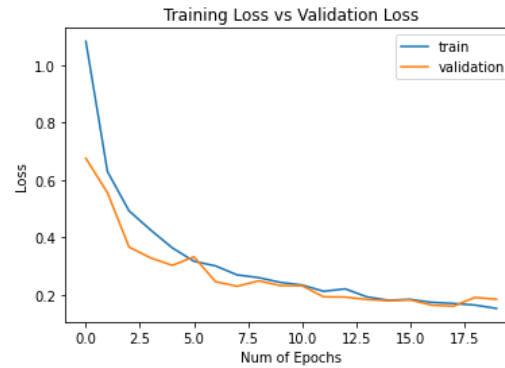
5 Entrenamiento

El modelo se entrenó de acuerdo a las consideraciones del taller, en este caso, durante 20 épocas con un tamaño de bloque de 200 (Batch), los resultados obtenidos se muestran a continuación, luego de la etapa de entrenamiento y pruebas.

A continuación, Podemos apreciar en las gráficas de validación el desempeño del algoritmo a lo largo de cada una de las épocas con respecto a nuevos datos en el proceso de entrenamiento, y la validación loss donde revisamos la medida de valuación de la función de perdida, es decir, el error del modelo en la fase de validación.



(a) Gráfica de entrenamiento y pruebas (Accuracy)



(b) Gráfica de entrenamiento y pruebas (Loss)

6 Elaboración del modelo 2

El modelo empleado para la red neuronal convolucional tiene la siguiente configuración

- Capa convolucional con 48 detectores de características, tamaño de kernel de 3x3, funcion de activación lineal rectificada (ReLU) de paso 1 y con padding.
- Capa de max pooling de tamaño 2x2
- Capa convolucional con 96 detectores de características, tamaño de kernel de 3x3, funcion de activación lineal rectificada (ReLU) de paso 1 y con padding.
- Capa convolucional con 96 detectores de características, tamaño de kernel de 3x3, funcion de activación lineal rectificada (ReLU) de paso 1 y con padding.
- Capa de max pooling de tamaño 2x2
- Capa de Flattening
- Capa full conectada con 100 neuronas y funcion de activación lineal rectificada (ReLU)
- Capa full conectada con 100 neuronas y funcion de activación lineal rectificada (ReLU)
- Capa de salida del mismo numero de las clases a clasificar y función de activación softmax.

```

Model: "sequential_1"
_____
Layer (type)                 Output Shape              Param #
-----
conv2d_2 (Conv2D)            (None, 32, 32, 48)       1344
max_pooling2d_2 (MaxPooling2D) (None, 16, 16, 48)       0
conv2d_3 (Conv2D)            (None, 16, 16, 96)       41568
max_pooling2d_3 (MaxPooling2D) (None, 8, 8, 96)         0
flatten_1 (Flatten)          (None, 6144)              0
dense_3 (Dense)              (None, 100)               614500
dense_4 (Dense)              (None, 100)               10100
dense_5 (Dense)              (None, 10)                1010
_____
Total params: 668,522
Trainable params: 668,522
Non-trainable params: 0
_____
None

```

Figure 3: Parámetros del modelo CNN en la implementación con TF y Keras

7 Entrenamiento 2

El modelo se entrenó de acuerdo a las consideraciones del taller, en este caso, durante 20 épocas con un tamaño de bloque de 200 (Batch), los resultados obtenidos se muestran a continuación, luego de la etapa de entrenamiento y pruebas:

- loss: 0.0619
- accuracy: 0.9818
- val_loss: 0.2320
- val_accuracy: 0.9467

De acuerdo con estos resultados la red neuronal convolucional ha logrado una alta precisión en el conjunto de entrenamiento, con una precisión del 98.18% y una pérdida del 0.0619, por su parte la precisión en el conjunto de validación es ligeramente menor, pero aún así es alta, con una precisión del 94.67% y una pérdida del 0.2320.

La diferencia entre la precisión del conjunto de entrenamiento y la precisión del conjunto de validación indica que la red podría estar sufriendo de sobreajuste (overfitting), es decir, que está memorizando demasiado los datos de entrenamiento y no está generalizando bien para nuevos datos, puede ser necesario evaluar la red en un conjunto de datos completamente nuevo para confirmar su capacidad de generalización y evitar el sobreajuste.

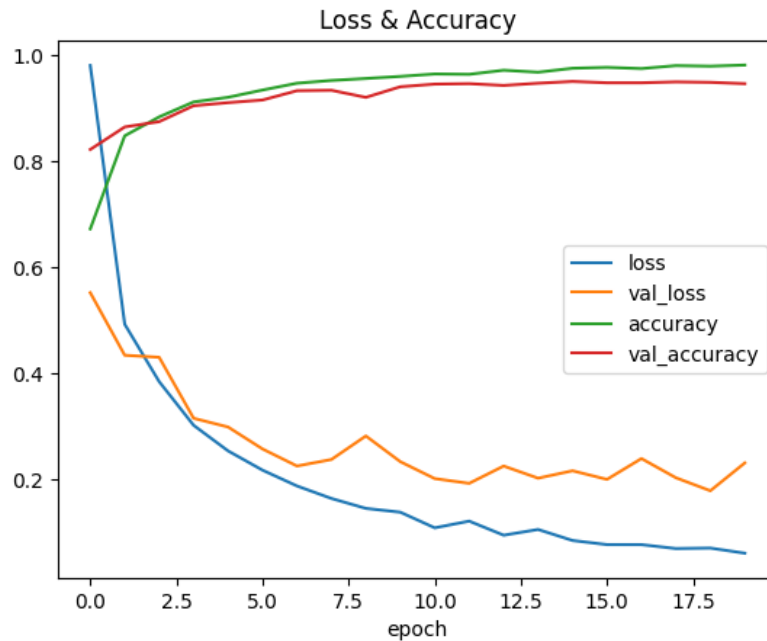


Figure 4: Resultado del proceso de entrenamiento de la CNN

8 Ajustes

No se puede determinar con certeza el balanceo de los datos sólo a partir de la matriz de confusión, ya que ésta sólo muestra la distribución de las predicciones y las etiquetas verdaderas. Sin embargo, se puede notar que algunas clases tienen un número muy pequeño de muestras, con menos de 10 muestras cada una. Esto puede indicar que el conjunto de datos no está equilibrado y puede llevar a una clasificación deficiente en estas clases. Sería importante revisar la distribución de clases en el conjunto de datos y, en caso de que haya un desequilibrio, considerar técnicas de equilibrado de datos, como aumentación de datos o submuestreo/sobremuestreo de clases minoritarias.

9 Análisis de Resultados

9.1 Modelo 1

Para el modelo 1, con la matriz de confusión se visualiza el desempeño de la red supervisada. Podemos observar por cada columna de la matriz, el número de predicciones por clase y en cada fila apreciar las instancias de los valores reales.

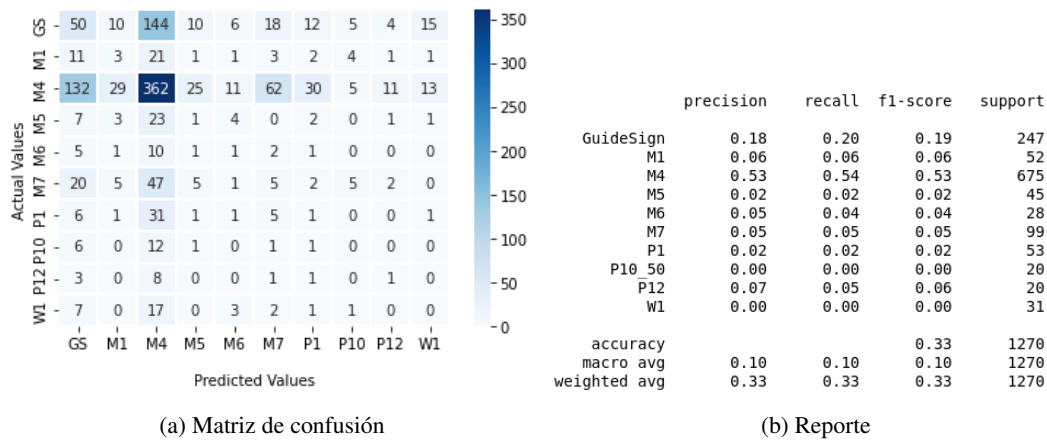
El modelo 1, tiene una precisión baja, aunque detecta algunas clases, se logra confundir con otra, esto quiere decir que el modelo no logra clasificar de forma correcta todos los casos.

9.2 Modelo 2

```
Evaluate on test data
11/11 [=====] - 1s 121ms/step - loss: 0.1982 - accuracy: 0.9568
test loss, test acc: [0.19824513792991638, 0.9567901492118835]
```

Figure 5: Resultado del proceso de validación de la CNN

En este caso, podemos ver que el modelo tiene un rendimiento muy bueno para la mayoría de las clases, con altos números en la diagonal principal de la matriz. Sin embargo, hay algunas clases donde el modelo tiene dificultades



para hacer predicciones precisas. Por ejemplo, la clase 0 (GuideSign) tiene 60 predicciones correctas y solo una incorrecta, mientras que la clase 2 (M4) tiene 167 predicciones correctas y solo dos incorrectas. Por otro lado, hay clases con un número relativamente pequeño de muestras, como la clase 6 (P12) y la clase 7 (W1), por lo que la capacidad del modelo para clasificar estas clases puede ser menos confiable debido a la falta de datos.

En general, la matriz de confusión muestra que el modelo tiene un rendimiento sólido en la mayoría de las clases, pero puede haber áreas donde se pueda mejorar el rendimiento a través de la recopilación de datos adicionales o la optimización del modelo.

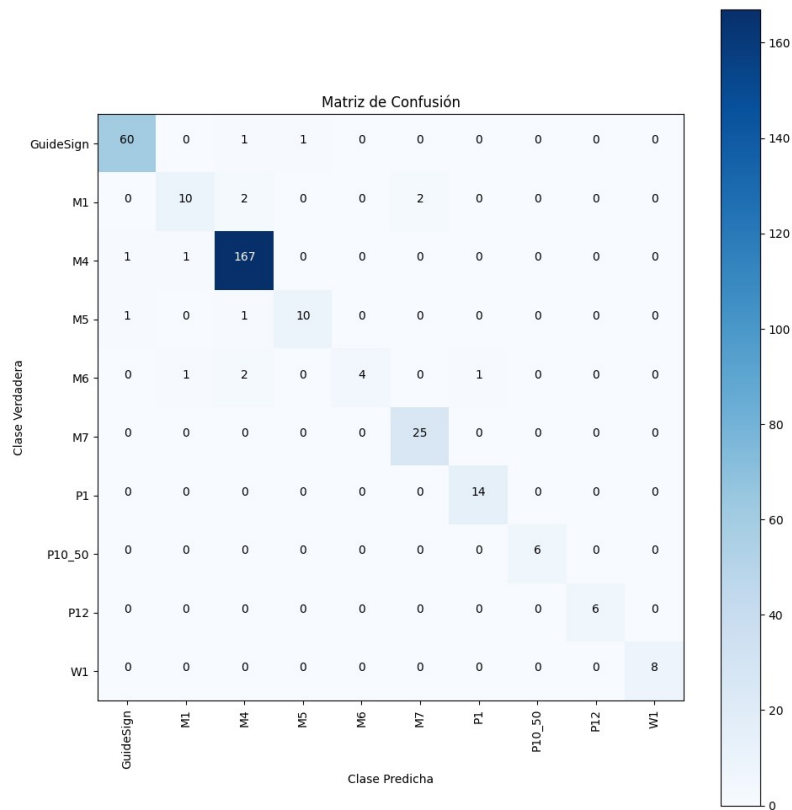


Figure 6: Matriz de confusión de la CNN

10 Conclusiones

Se puede concluir que el modelo tiene un buen desempeño para clasificar algunas de las clases, mientras que para otras clases, la precisión disminuye pero no de manera drástica.

Por ejemplo, podemos ver que la clase 2 (M4) fue clasificada con alta precisión, ya que sólo hubo un error en la clasificación de 169 imágenes. De manera similar, las clases 6 y 7 (P1 y P10_50) fueron clasificadas con precisión perfecta, sin errores. Sin embargo, para algunas otras clases, como la clase 0 (GuideSign), la precisión es menor, ya que hubo 2 errores en la clasificación de 62 imágenes.

Podría ser necesario analizar más a fondo el modelo y los datos de entrenamiento para determinar las posibles causas de estas imprecisiones y tomar medidas para mejorar el modelo.

Referencias

https://www.tensorflow.org/guide/keras/train_and_evaluate?hl=es-419