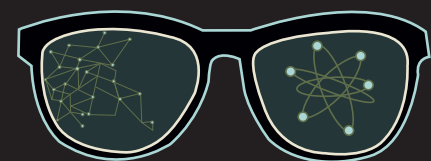




CHEAT SHEET



REDES N. CONVOLUCIONALES

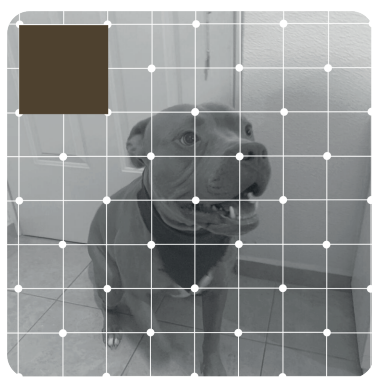
Las redes neuronales convolucionales o RNC, son un tipo especial de redes neuronales ya que procesan datos que tienen la forma de cuadrícula, o matrices si hablamos de manera más formal. El nombre de redes convolucionales se les da debido a su operación matemática principal la “convolución” la cual es un tipo especial de operación lineal, y es típicamente denotada por un asterisco. Como podemos ver en la siguiente ecuación:

$$s(t) = (x * w)(t)$$

Donde: **x**- Entrada
w- Parámetros
t- Tiempo

ENTRADAS A LA RNC

La imagen entra con una dimensión definida previamente, pasan por las operaciones de convolución y activación entre cada capa convolucional. Supongamos que tenemos la siguiente imagen (izquierda) las entradas a la RNC sería una sección (derecha).



FUNCIONAMIENTO

El funcionamiento se describe de la siguiente forma: tenemos una imagen y un kernel.



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

Imagen o matriz

1	0	-1
1	0	-1
1	0	-1

Kernel = filtro



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

1	0	-1
1	0	-1
1	0	-1

-5			

Se toma una parte de la matriz (imagen) y se multiplica por el kernel, el resultado se almacena en una nueva matriz. Este proceso se repite hasta terminar con todos los elementos de la imagen y podemos decir que aplicamos una convolución.



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9



1	0	-1
1	0	-1
1	0	-1



-5	-4		

MI PRIMERA RED N CONVOLUCIONAL

```
[1] from tensorflow.keras.layers import Conv2D, Flatten, Dense
(x_ent, y_ent), (x_val, y_val)= tf.keras.datasets.cifar10.load_data()
```

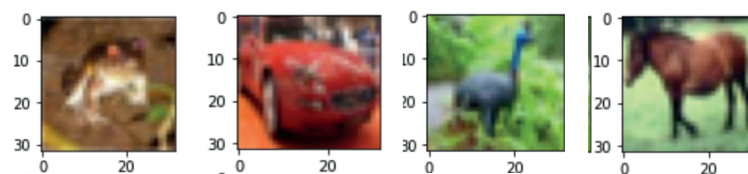
```
mi_red= Sequential([
    Dense(10, activation= 'relu'),
[2]   Dense(3, activation= 'relu'),
    Dense(10, activation= 'softmax')
])
```

```
mi_red.compile(optimizer= 'SGD',
[3]               loss= 'sparse_categorical_crossentropy',
               metrics= ['accuracy'])
```

```
mi_red.fit(x_ent, y_ent, epochs= 20,
[4]         validation_data= (x_val, y_val)
        )
```

METIENDONOS AL CODIGO

Importando la librería de tensorflow y la base de datos cifar10, que contiene imágenes random:



Se diseña la red convolucional con tres capas, de 10, 3 y 10 neuronas y las funciones de activación para cada capa.

Para medir el error del modelo le asignamos la función pérdida **sparse_categorical_crossentropy**, para que el modelo aprenda utilizamos el optimizador **SGD** y como métrica extra medimos la exactitud (**accuracy**).

Creamos el modelo con 10 épocas y pasamos los datos de entrenamiento y validación.