Deep Learning aplicado a clasificación de imágenes

CNN - Redes Neuronales Convolucionales

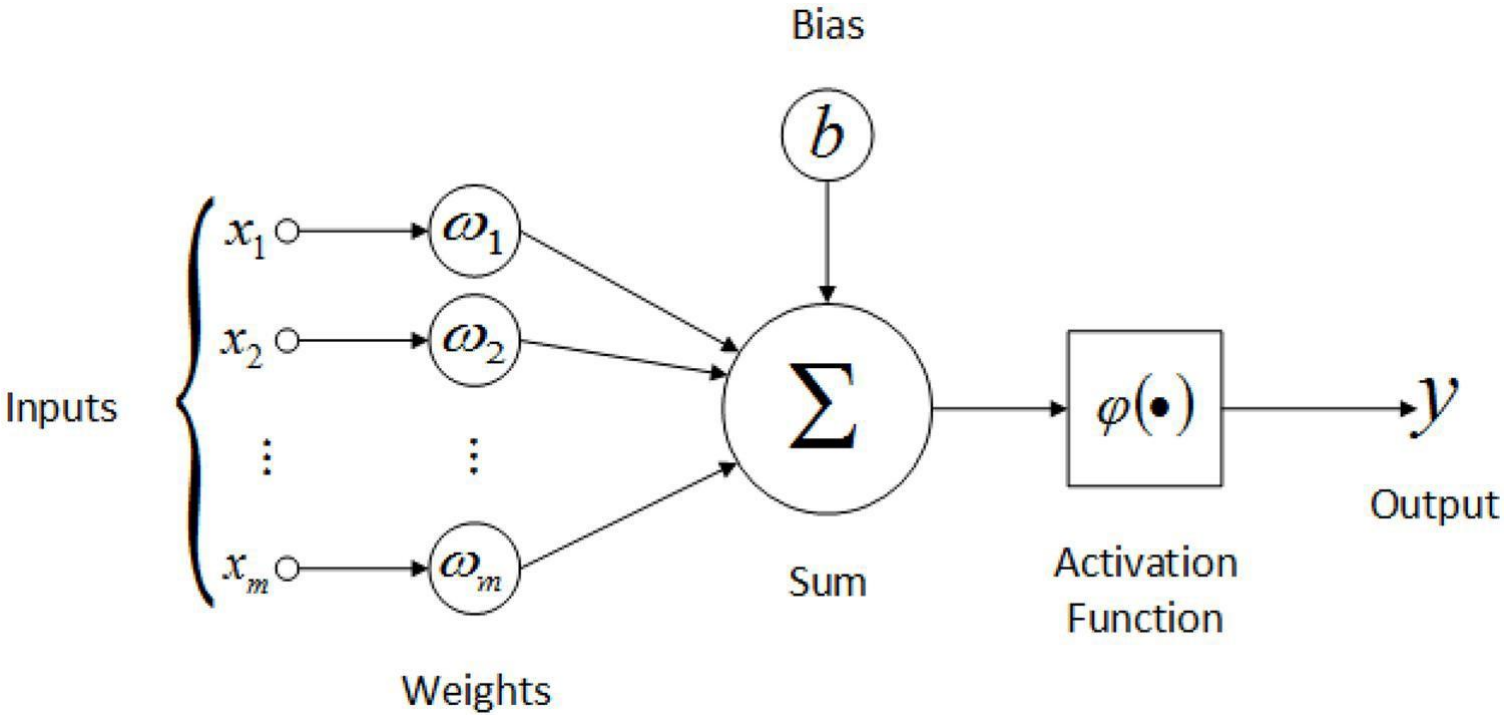
Ing. Mauro Salina

SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

INSTITUTO INGENIERÍA Y AGRONOMÍA | Carrera Ingeniería en Informática

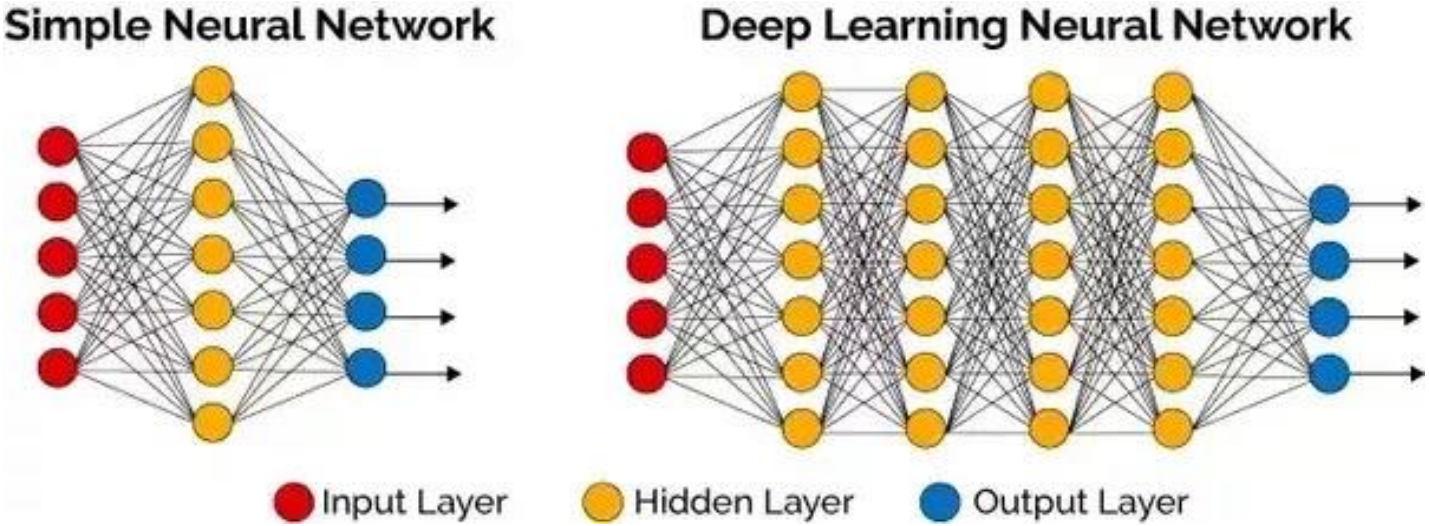
Universidad Nacional ARTURO JAURETCHE

Neurona Artificial



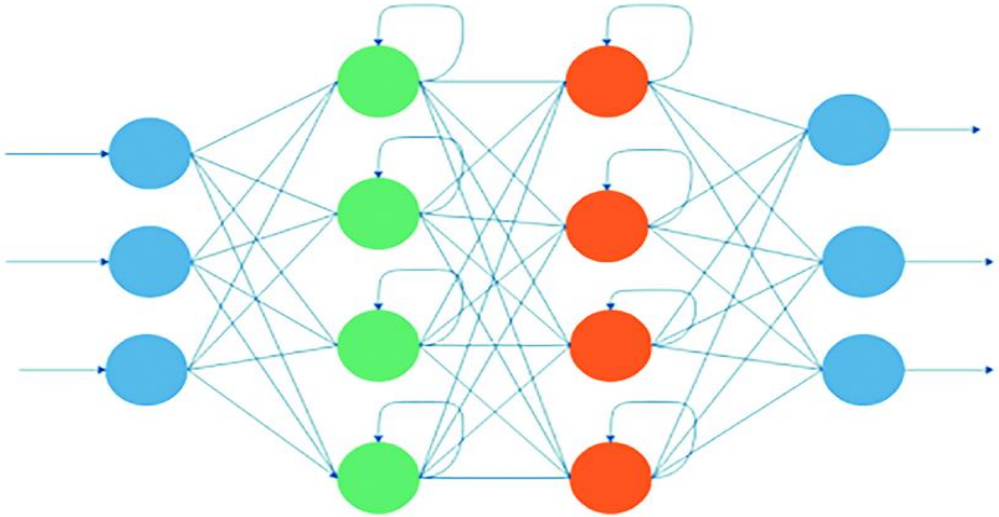
Arquitectura de red neuronal artificial

- DNN – Red Neuronal Profunda, cuenta con una capa de entrada, 2 o más capas ocultas, y una capa de salida.
- Todas las capas están completamente conectadas (red neuronal densa).



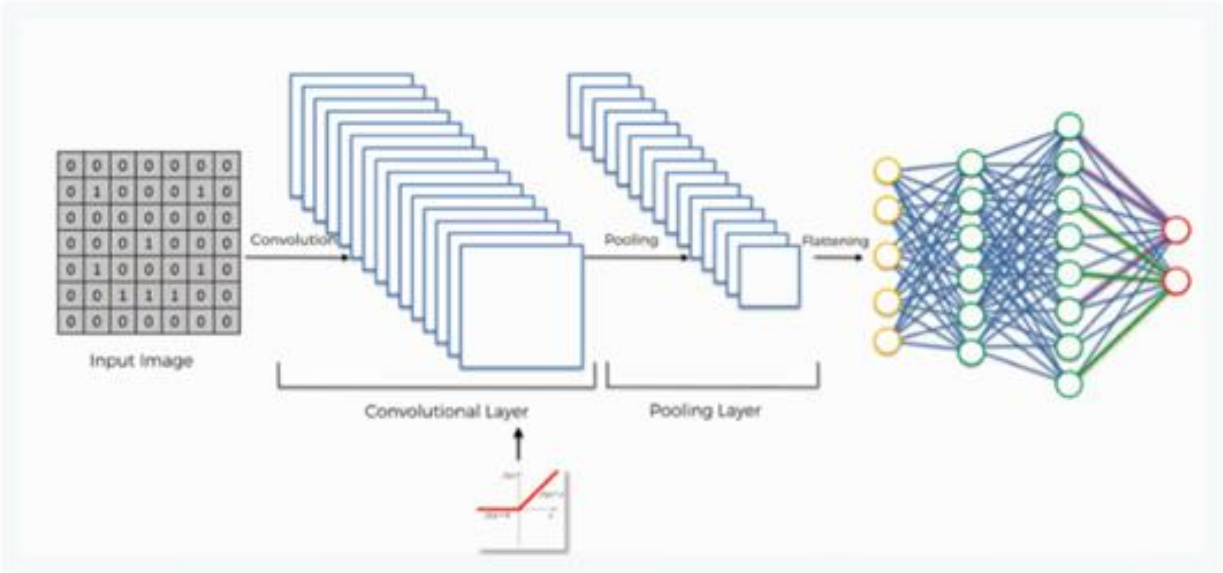
Principales Arquitecturas de redes neuronales

- RNN – Redes Neuronales Recurrentes.
- Las salidas de cada capa oculta se retroalimenta como entradas de la capa.



Principales Arquitecturas de redes neuronales

- CNN – Redes Neuronales Convolucionales, luego de la capa de entrada tiene 1 o mas capas convolucionales en donde se extraen características y luego contiene una red completamente conectada antes de la capa de salida.

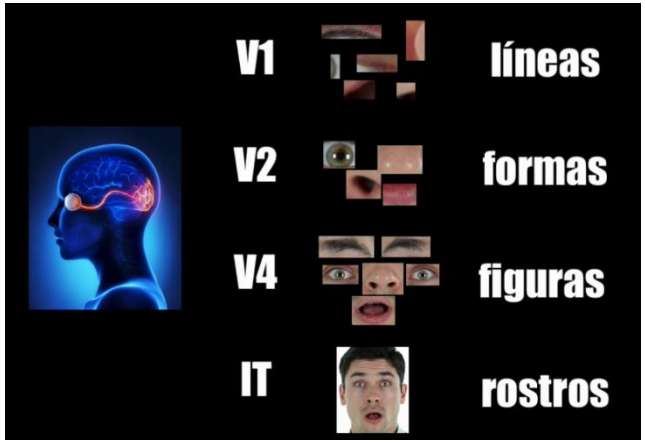


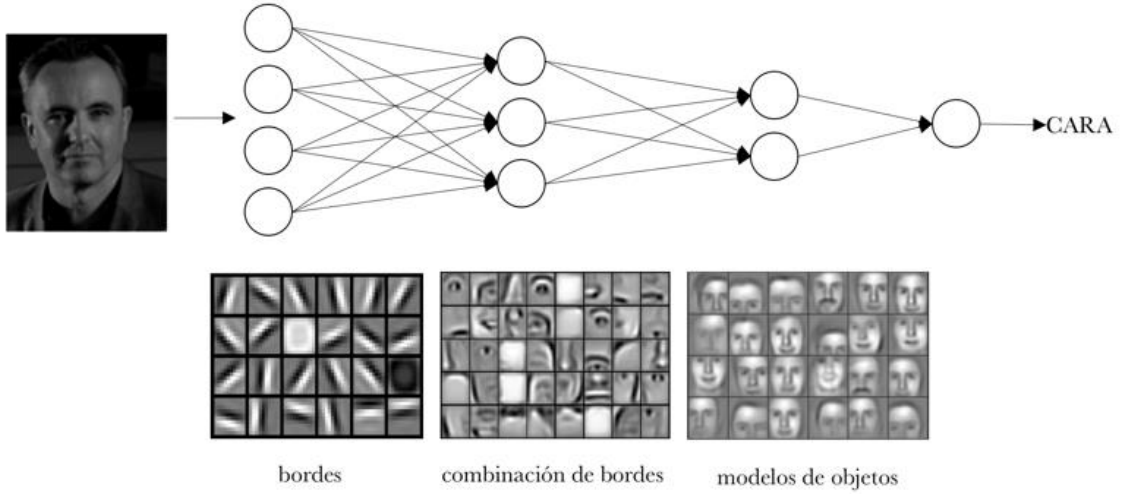
Redes Neuronales Convolucionales

SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

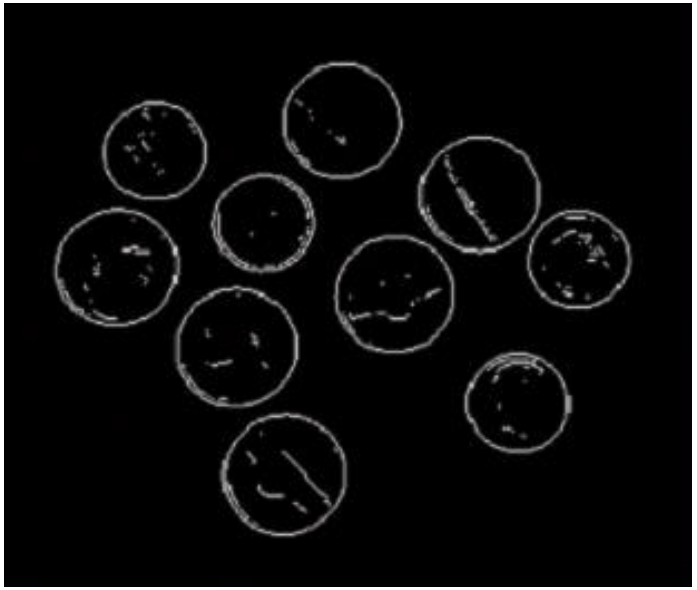
- Basada en el funcionamiento de la corteza visual y el cerebro humano.
- Tiene muy buenos resultados en el procesamiento de voz, texto y sobre todo imágenes (clasificación, detección de objetos).
- Requiere un número menor de conexiones entre las capas de neuronas.
- Se puede dividir en dos grandes partes:
  - Aprendizaje de características (learning feature)
    Convolución – Activación – Pooling (Agrupación)
  - Clasificación
    Flatten-FullyConected-Salida(Softmax)

CNN

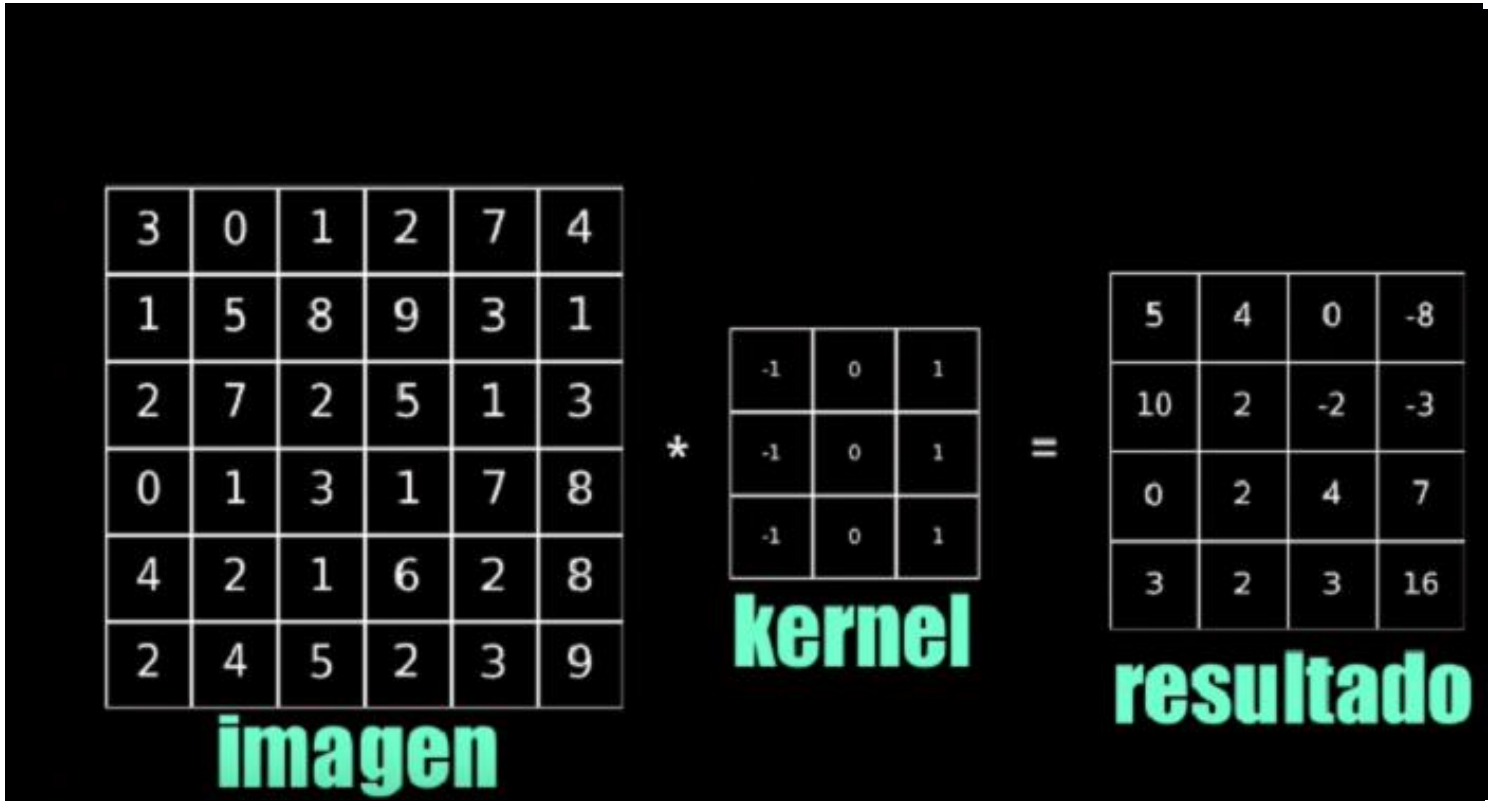

V1
líneas
V2
formas
V4
figuras
IT
rostros


CARA
bordes
combinación de bordes
modelos de objetos

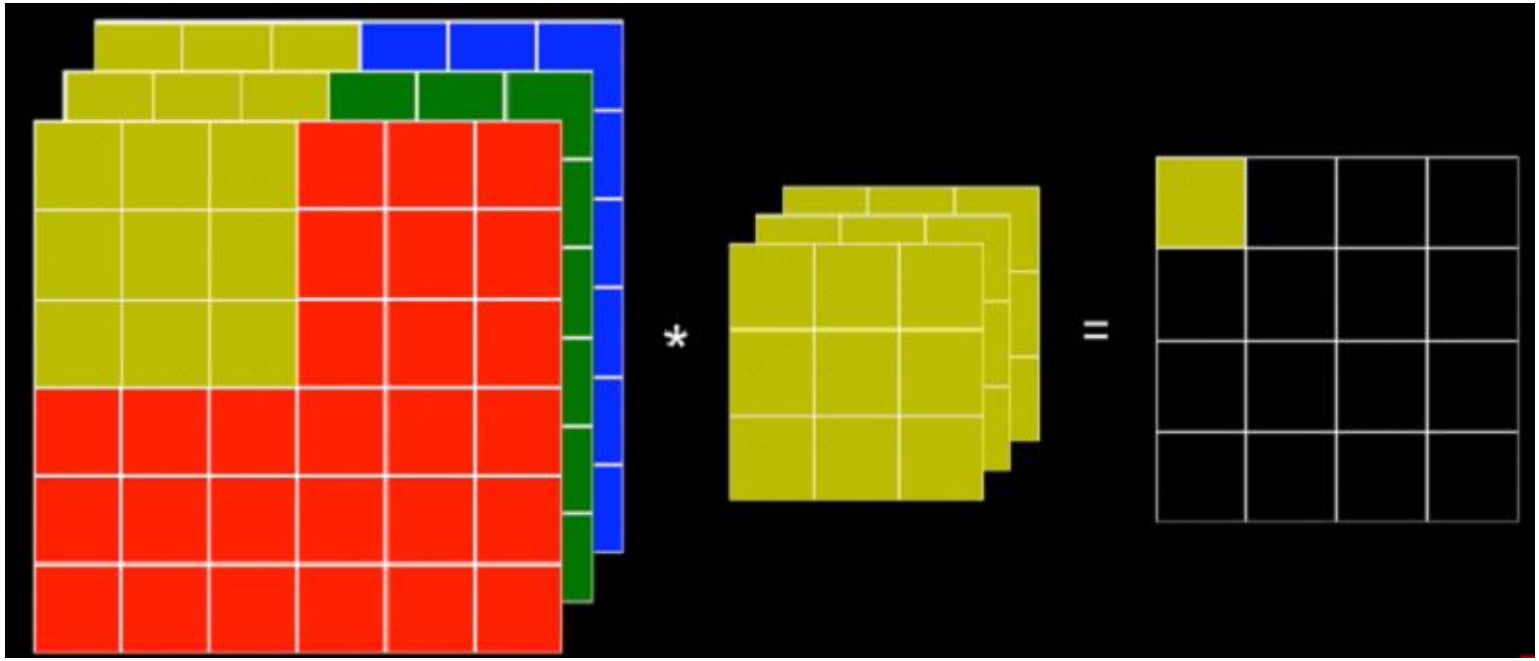
CNN – Filtro o Kernel (imagen escala de grises)



CNN – Filtro o Kernel (imagen escala de grises)



CNN – Filtro o Kernel (imagen color)



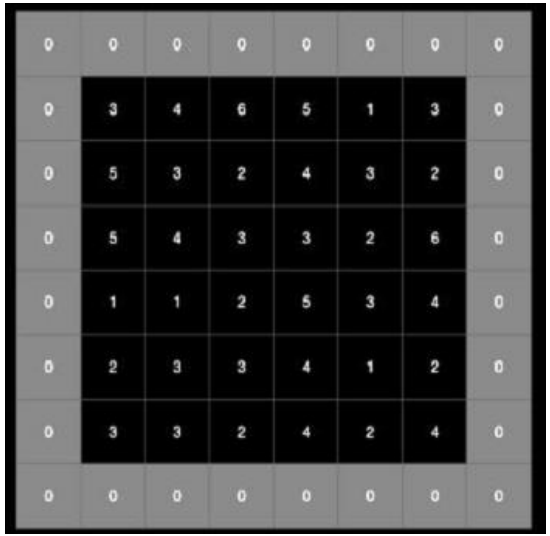
6 x 6 x 3 | 3 x 3 x 3 | 4 x 4 x 1

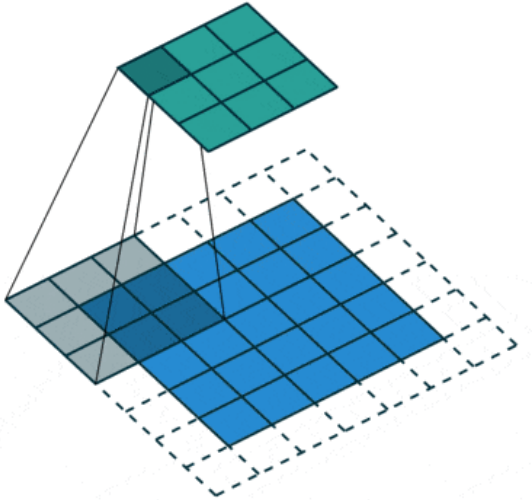
El tamaño de salida está dado por:

$M_{sal} = M_{ent} - K + 1$ **(se aplica a cada dimensión)**

CNN – Padding

- El padding se aplica agregando píxeles de valor cero alrededor de la imagen original.
- Su función:
  - La imagen resultante sea de igual tamaño que la imagen original.
  - Obtener información relevante que se encuentre en las esquinas de las imágenes originales.



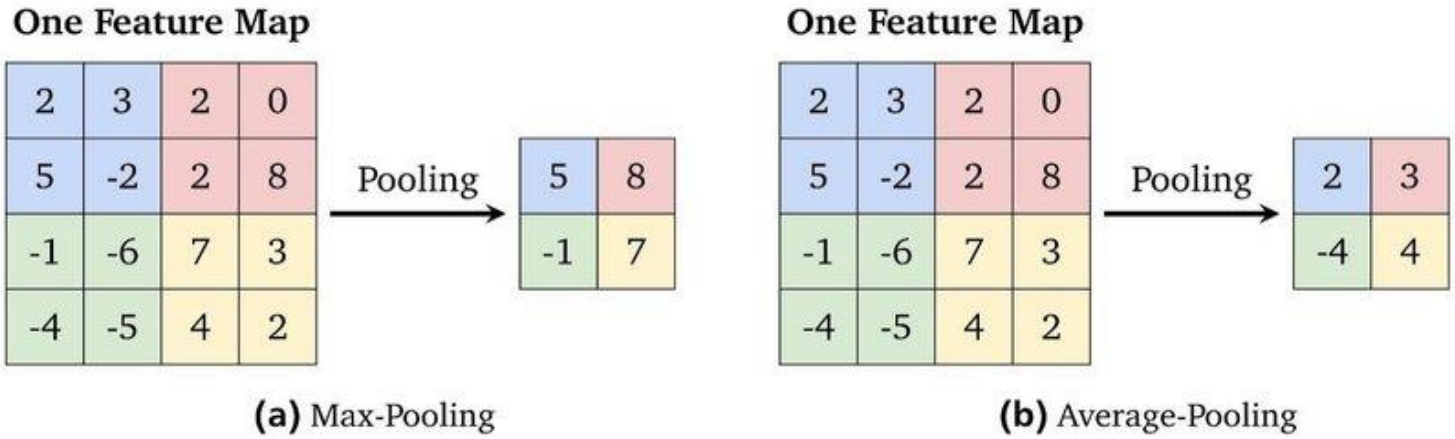


CNN - Kernel

-1	-1	-1
-1	8	-1
-1	-1	-1

CNN – Pooling o subsampling

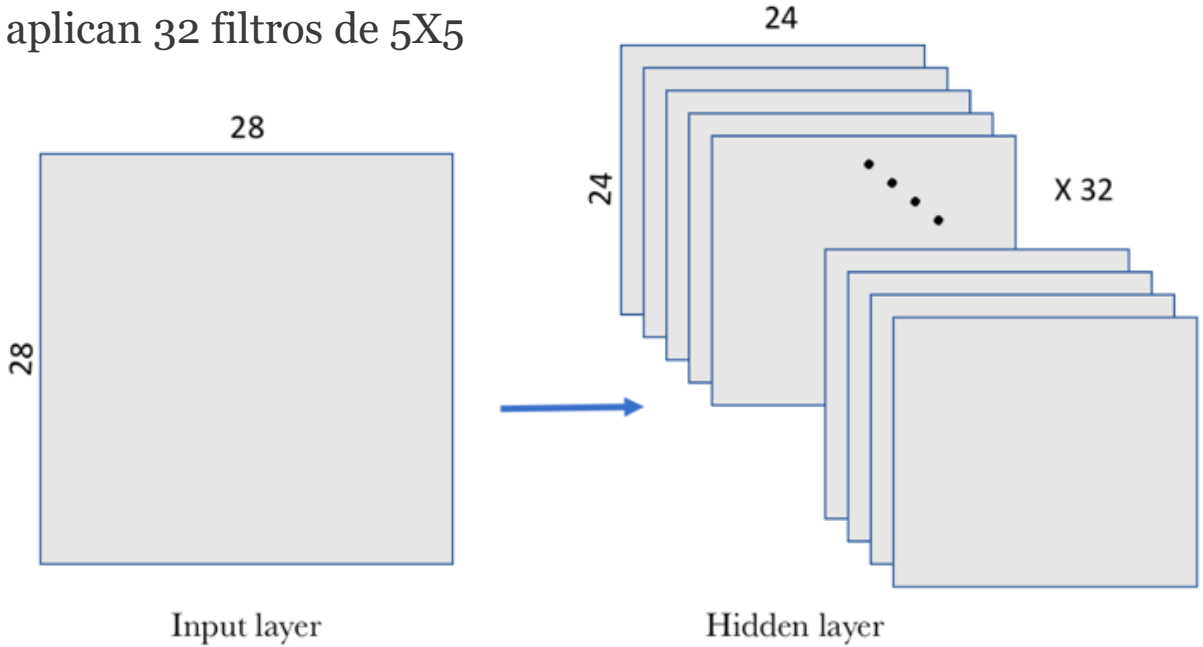
- Reduce el tamaño de las imágenes a procesar
- Agrupa las principales características
- Genera invariabilidad espacial



CNN – Pooling o subsampling

CNN – Mapas de característica - Stacking

- Se aplican 32 filtros de 5X5

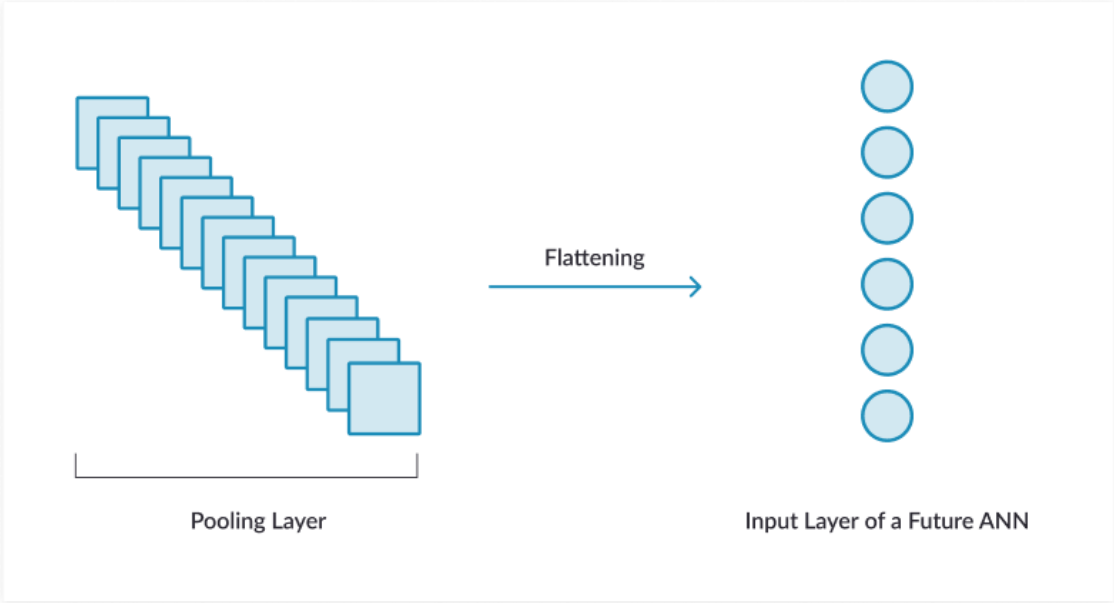


CNN – Clasificación

- Esta capa se conoce como capa completamente conectada.
- Inicia con un aplanamiento.
- Sigue una red neuronal tradicional.
- Por último la capa de salida con activación sigmoide o softmax.

CNN – Flatten o aplanamiento

- Se aplana la última capa de mapas de características formando como resultado un vector 1 x 1 x n, el valor de n dependerá del tamaño de las imágenes generadas por los filtros.



CNN – Fully Connected Layer

- Se forma una red ANN o DNN según las necesidades del problema.

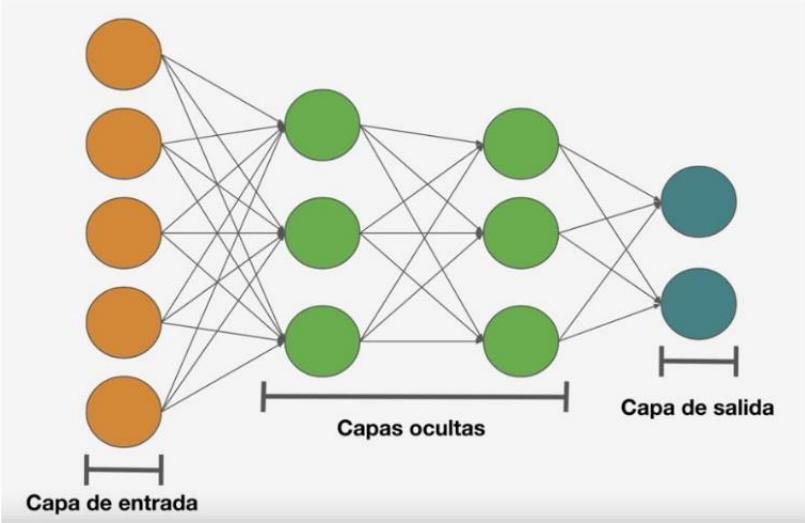
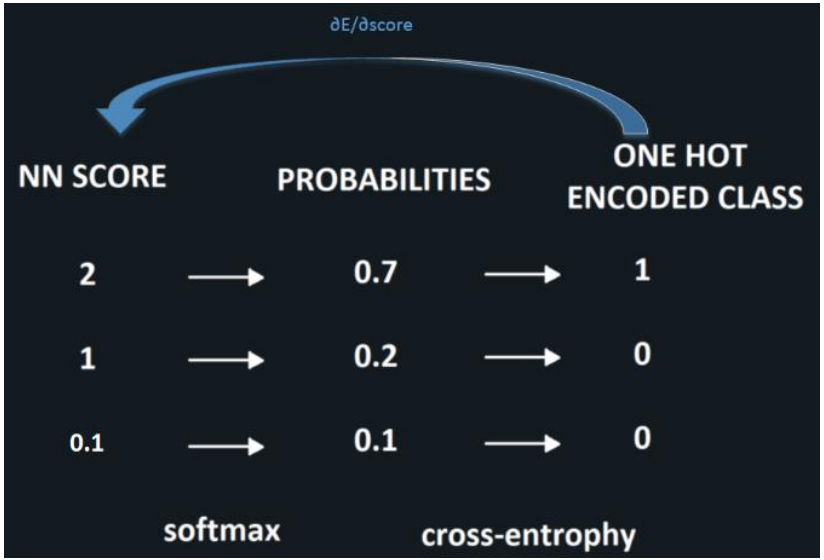


Figura 6: Arquitectura de una DNN

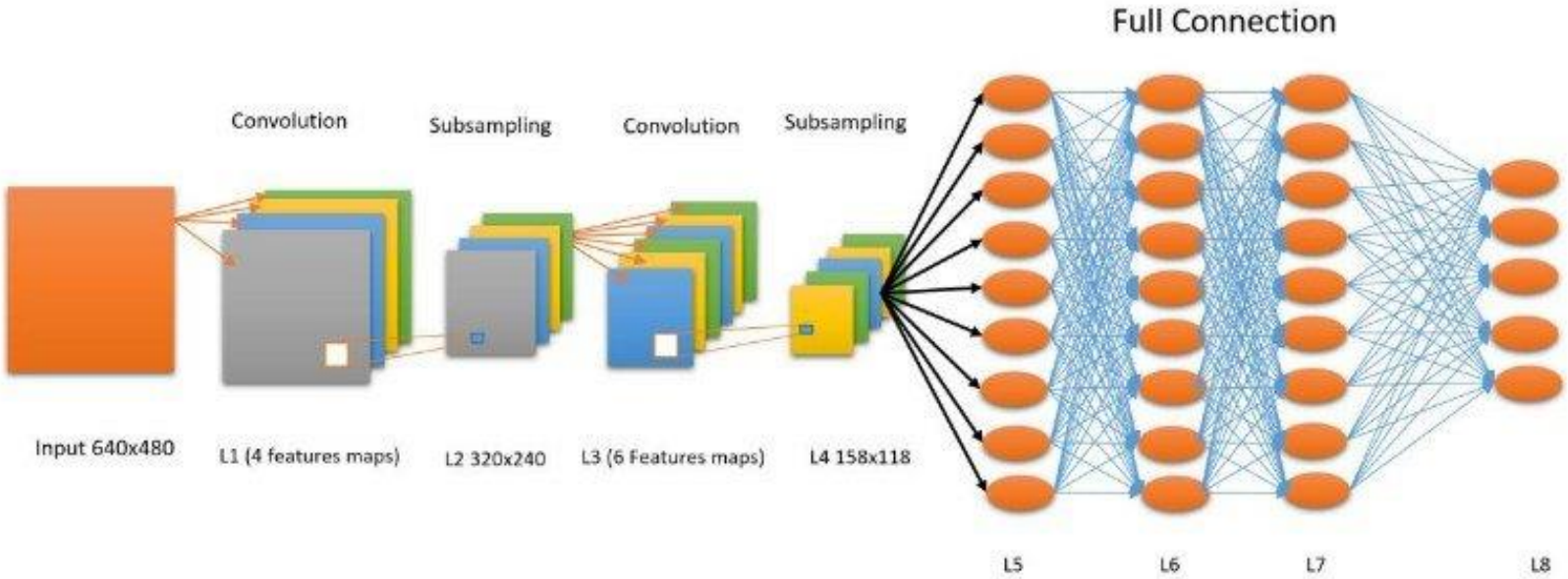
CNN – Activación Softmax

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

$$L_{\text{cross-entropy}}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_i y_i \log(\hat{y}_i)$$



CNN – Arquitectura completa

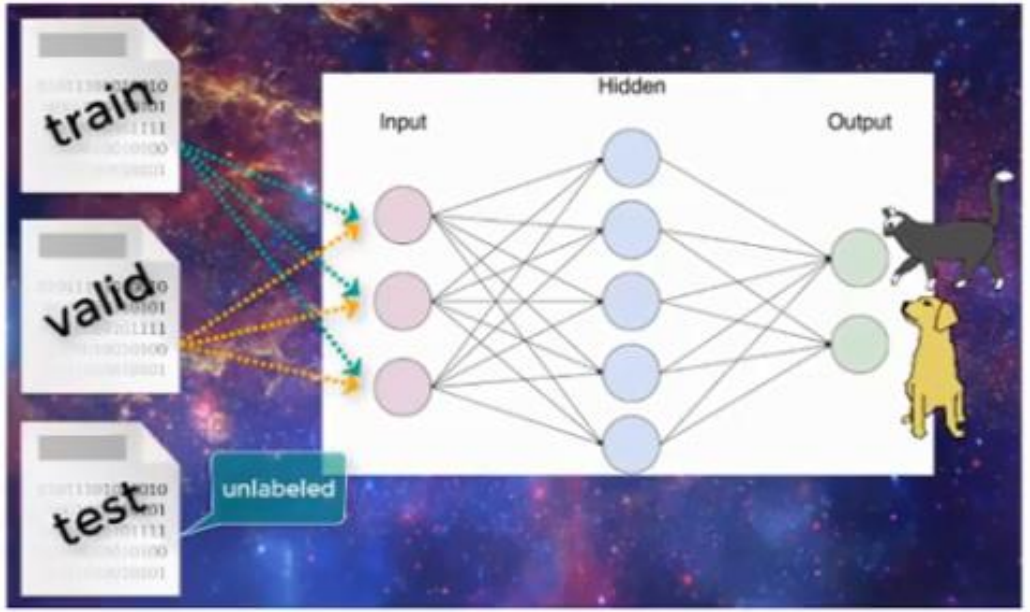


Preparación del DataSet

reddit

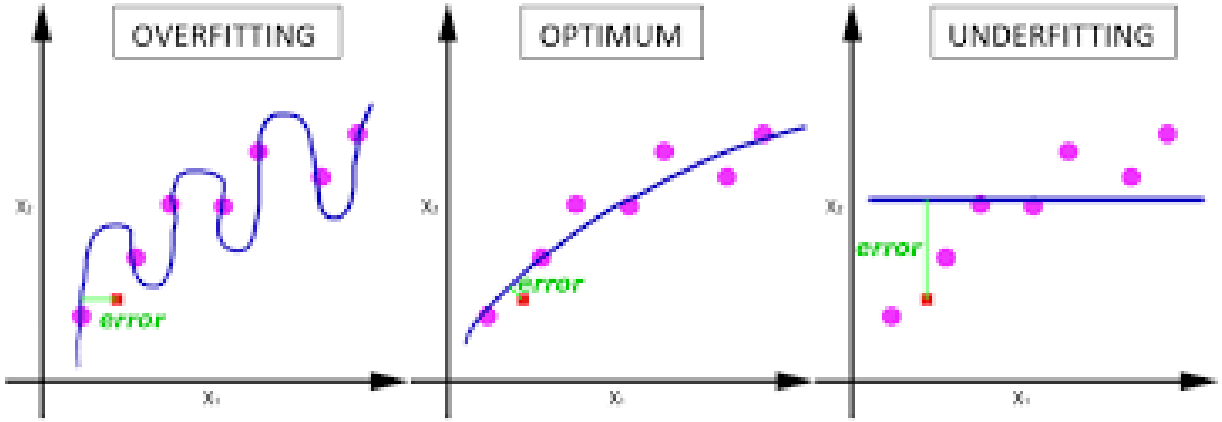
kaggle

GitHub



Problemas de Underfitting y Overfitting

- No converge
- Converge con Underfitting
- Converge con Overfitting
- Converge de manera óptima o apropiada



Underfitting (subajuste)

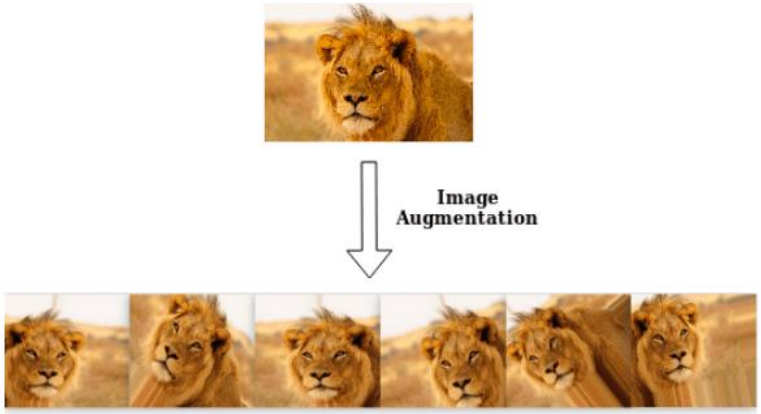
- Precisión del set de entrenamiento y validación baja.
- Modelo no capaz de generalizar.
- Posibles motivos:
  - Modelo demasiado simple.
  - Modelo con pocas iteraciones de entrenamiento.
  - Arquitectura del modelo no adecuada.
  - Poca cantidad y variedad en los datos de entrenamiento.

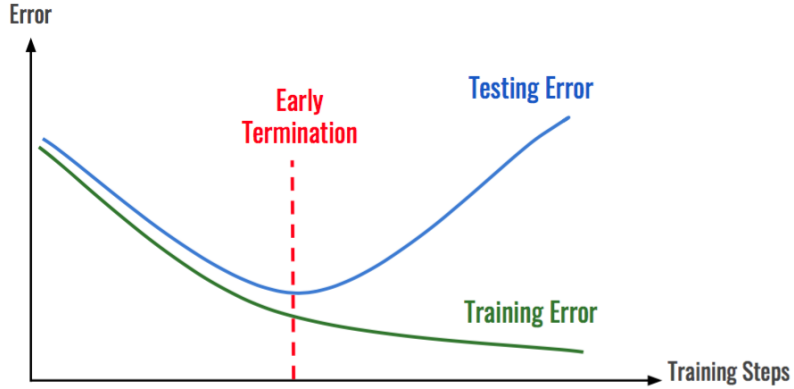
Overfitting (sobreajuste)

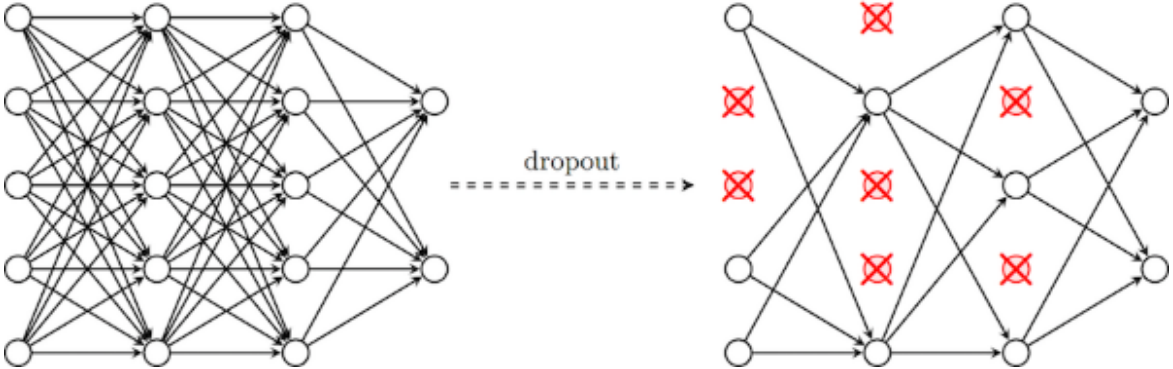
SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

- Precisión del set de entrenamiento alta y validación baja.
- Modelo memoriza el set de entrenamiento.
- Posibles motivos:
  - Pocos datos de entrenamiento.
  - Modelo demasiado complejo.
- Maneras de prevenir:
  - Image augmentation
  - Dropout
  - Early Stopping

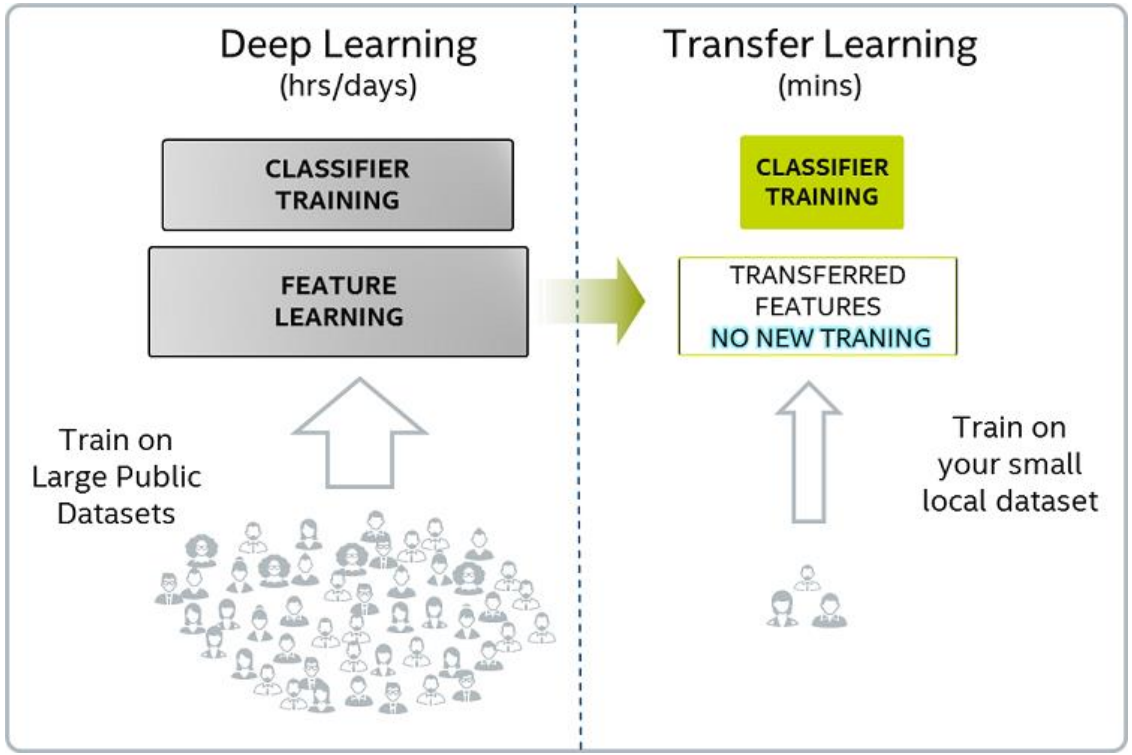
Aumento de imágenes – Dropout – Parada Temprana







Diferencia entre tipos de entrenamiento



Introducción a Transfer Learning

- Utiliza una red pre-entrenada .
- Se reentrena para hacer una nueva clasificación.
- Ahorra tiempo de entrenamiento.
- Requiere menor capacidad de cómputo.
- Se obtienen buenos resultados.
- Útil para modelos con dataset pequeños.

Motivación – Importancia de reciclar

Motivación – Importancia de reciclar

- Es sumamente importante el reciclado de desechos para el cuidado del medioambiente.
- Se trata de la reutilización de elementos u objetos ya utilizados, los que de otro modo serían desechados contribuyendo al aumento de la formación de basura y al daño ambiental permanente.
- En nuestro país, aunque se trata de fomentar el reciclado, solo el 24% de la población hace un esfuerzo por separar los residuos, esto es debido al esfuerzo que la tarea conlleva.

Modelo propuesto y análisis de resultados

- Dataset utilizados:
  - Dividido en 2 clases (Reciclables – No reciclables)

  Train: 6532 imágenes – Valid: 1708 imágenes – Test: 100 imágenes

  - Dividido en 6 clases (Metal - Papel/Carton – Vidrio – Plástico - Orgánico - No reciclable)

  Train: 11976 imágenes – Valid: 3039 imágenes – Test: 90 imágenes

- Tamaño entrada imágenes 200 x 200 pixeles

- Se probaron ambos dataset con un modelo entrenado desde el inicio y con aprendizaje por transferencia

Herramientas utilizadas

Matriz de confusión

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$

Matriz de confusión para una variable de salida dicotómica (clasificación)		Resultado de la predicción	
		Sí	No
Valor real de la clase	Sí	Verdadero positivo	Falso Negativo
	No	Falso positivo	Verdadero negativo

Ejemplos de imágenes utilizadas en el dataset

METAL

ORGANICOS

PAPEL/CARTON

NO-RECICLABLE

PLASTICO

VIDRIO

Modelos desarrollados (dos clases)

SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

Model: "sequential"

Layer (type)	Output	Shape Param #
conv2d (Conv2D)	(None, 200, 200, 32)	896
max_pooling2d (MaxPooling2D)	(None, 100, 100, 32)	0
conv2d_1 (Conv2D)	(None, 100, 100, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 50, 50, 64)	0
conv2d_2 (Conv2D)	(None, 50, 50, 128)	32896
conv2d_3 (Conv2D)	(None, 50, 50, 128)	65664
max_pooling2d_2 (MaxPooling2D)	(None, 25, 25, 128)	0
flatten (Flatten)	(None, 80000)	0
dense (Dense)	(None, 512)	40960512
dropout (Dropout 0,5)	(None, 512)	0
dense_1 (Dense)	(None, 256)	131328
dropout_1 (Dropout 0,3)	(None, 256)	0
dense_2 (Dense)	(None, 256)	65792
dropout_2 (Dropout 0,3)	(None, 256)	0
dense_3 (Dense)	(None, 2)	514

Total params: 41,276,098
Trainable params: 41,276,098
Non-trainable params: 0

Modelos desarrollados (seis clases)

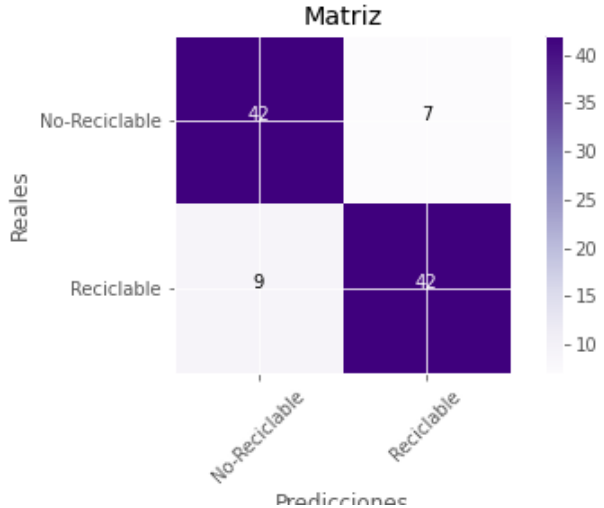
SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

Model: "sequential"

Layer (type)	Output	Shape Param #
conv2d (Conv2D)	(None, 200, 200, 32)	896
conv2d_1 (Conv2D)	(None, 200, 200, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 100, 100, 32)	0
conv2d_2 (Conv2D)	(None, 100, 100, 64)	18496
conv2d_3 (Conv2D)	(None, 100, 100, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 50, 50, 64)	0
conv2d_4 (Conv2D)	(None, 50, 50, 128)	32896
conv2d_5 (Conv2D)	(None, 50, 50, 128)	65664
conv2d_6 (Conv2D)	(None, 50, 50, 128)	65664
max_pooling2d_2 (MaxPooling2D)	(None, 25, 25, 128)	0
conv2d_7 (Conv2D)	(None, 25, 25, 256)	131328
conv2d_8 (Conv2D)	(None, 25, 25, 256)	262400
conv2d_9 (Conv2D)	(None, 25, 25, 256)	262400
max_pooling2d_3 (MaxPooling2D)	(None, 12, 12, 256)	0
flatten (Flatten)	(None, 36864)	0
dense (Dense)	(None, 1024)	37749760
dropout (Dropout 0,5)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524800
dropout_1 (Dropout 0,3)	(None, 512)	0
dense_2 (Dense)	(None, 512)	262656
dropout_2 (Dropout 0,3)	(None, 512)	0
dense_3 (Dense)	(None, 6)	3078

Total params: 39,426,214
Trainable params: 39,426,214
Non-trainable params: 0

Resultados dataset 2 clase

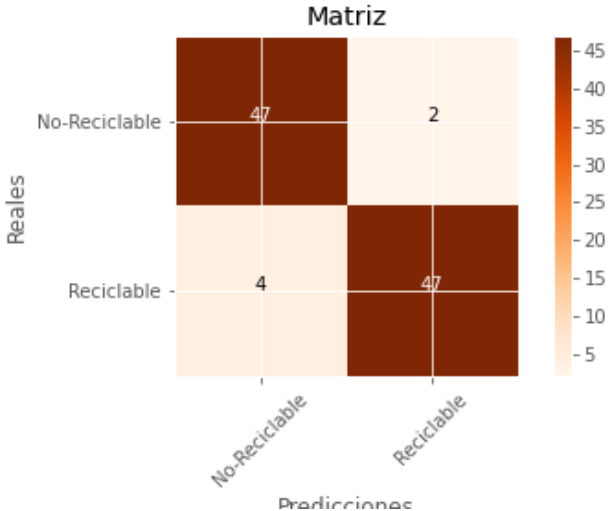


$$Precision = \frac{TP}{TP + FP} = \frac{42}{42 + 9} \approx 0{,}82$$

$$Recall = \frac{TP}{TP + FN} = \frac{42}{42 + 7} \approx 0{,}85$$

$$\text{F1} = 2 * \frac{precision * recall}{precision + recall} = 2 * \frac{0{,}82 * 0{,}85}{0{,}82 + 0{,}85} \approx 0{,}83$$

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{42 + 42}{42 + 42 + 7 + 9} \approx 0{,}84$$



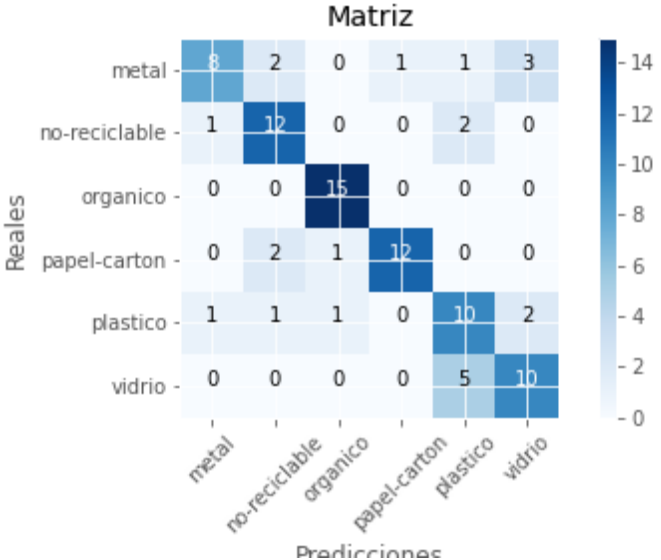
$$Precision = \frac{TP}{TP + FP} = \frac{47}{47 + 4} \approx 0,\mathbf{9}2$$

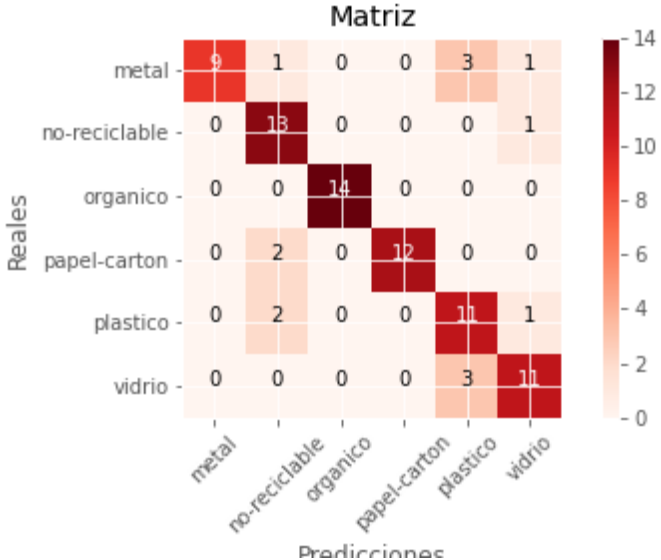
$$Recall = \frac{TP}{TP + FN} = \frac{47}{47 + 2} \approx 0{,}95$$

$$\text{F1} = 2 * \frac{precision * recall}{precision + recall} = 2 * \frac{0{,}92 * 0{,}95}{0{,}92 + 0{,}92} \approx 0{,}93$$

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{47 + 47}{47 + 47 + 4 + 2} \approx 0{,}94$$

Resultados dataset 6 clases





$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{67}{90} \approx 0{,}744$$

$$\text{F1}_{\text{vidrio}} = 2 * \frac{0{,}66*0{,}66}{0{,}66+0{,}66} \approx 0{,}66$$

$$\text{F1}_{\text{organico}} = 2 * \frac{0{,}882*1}{0{,}882+1} \approx 0{,}937$$

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{70}{84} \approx 0{,}833$$

$$\text{F1}_{\text{metal}} = 2 * \frac{1*0{,}642}{1+0{,}642} \approx 0{,}781$$

$$\text{F1}_{\text{organico}} = 2 * \frac{1*1}{1+1} = 1$$

Motivación

SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

- El inconveniente de fabricar productos con defectos de fábrica se puede solucionar mediante técnicas de inteligencia artificial.

Redes para detección de objetos

- Para realizar una detección de objetos usando redes neuronales es conveniente usar arquitecturas basadas *en propuestas de región de interés*.

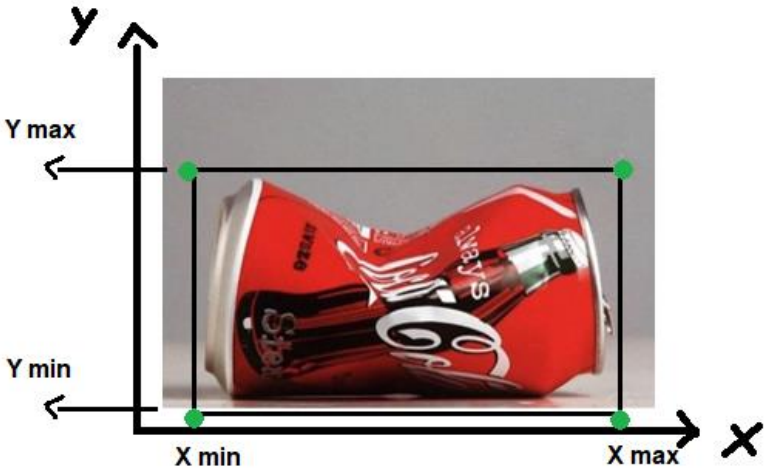
La arquitectura utilizada en este trabajo fue la denominada Faster R-CNN.

Faster R-CNN

Está compuesto por imágenes y por cada imagen un archivo con extensión XML.

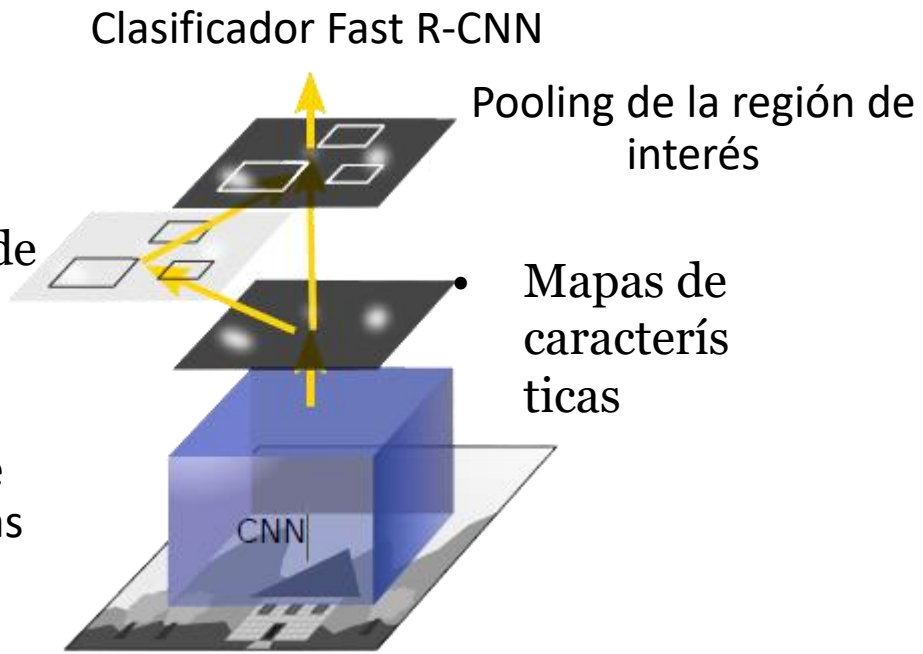
```
<annotation>
        <folder>images_resize</folder>
        <filename>1366_2000_cocuchasdsd.jpg</filename>
        <path>C:\Users\1366_2000_cocuchasdsd.jpg</path>
        <source>
                <database>Unknown</database>
        </source>
        <size>
                <width>448</width>
                <height>448</height>
                <depth>3</depth>
        </size>
        <segmented>0</segmented>
        <object>
                <name>Lata_defectuosa</name>
                <pose>Unspecified</pose>
                <truncated>0</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>26</xmin>
                        <ymin>163</ymin>
                        <xmax>413</xmax>
                        <ymax>369</ymax>
                </bndbox>
        </object>
</annotation>
```

Faster R-CNN

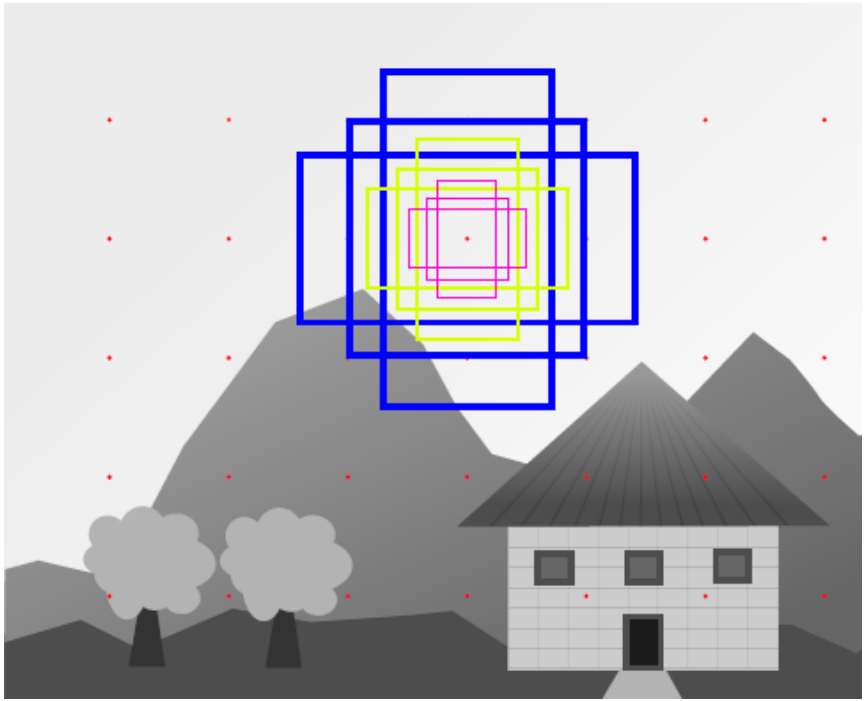



```
<object>
        <name>Lata_defectuosa</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
                <xmin>26</xmin>
                <ymin>163</ymin>
                <xmax>413</xmax>
                <ymax>369</ymax>
        </bndbox>
</object>
```

Faster R-CNN


Clasificador Fast R-CNN
Pooling de la región de interés
Propuesta de región
Mapas de caracterís ticas
Extractor de características
CNN

Redes de propuesta de región de interés



- ✓ 9 anclajes
- ✓ 3 relaciones de aspecto
- ✓ 3 escalas

Entrenamiento de la Red de propuesta de región

- Se asigna clase binaria
- Para activar los anclajes se utiliza IoU (intersección sobre unión) o Índice de Jaccard

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Dataset

- ✓ Datos de entrenamiento: 590 archivos
- ✓ Datos de validación: 294 archivos

Las clases usadas para clasificar cada objeto: ***Lata****,* ***Lata_defectuosa****,* ***Botella****,* ***Botella_defectuosa****.*

Evaluación de rendimiento

Resnet50

```
average precision:0.7613592959499047
Evaluated performance's results


------------------
(program exited with code: 0)
```

AP=76%

MobileNet

```
average precision:0.7210792959499045
Evaluated performance's results


------------------
(program exited with code: 0)
```

AP=72%

Pruebas de modelos

Raspberry PI 3

- ✓ Buenas detecciones, mayoría de clasificaciones correctas, minoría de falsos positivos.
- ✓ Performance de detección en tiempo real no fluido.

Pruebas del modelo

Líneas de investigación futuras

SASE 2022
SIMPOSIO ARGENTINO DE SISTEMAS EMBEBIDOS

- Mejora de los dataset.
- Rebalanceo de las clases.
- Prevención de ruido en las imágenes que afecten el modelo.
- Reentrenamiento y mejora de los modelos utilizando los datos mejorados.

- Profundizar en aprendizaje por transferencia.

¡¡¡GRACIAS!!!

¿Preguntas?