Universidad Nacional de San Luis
Facultad de Ciencias Físico, Matemáticas y Naturales
Departamento de electrónica.

Universidad Nacional de San Luis

Proyecto final de carrera:

"Detector de obstáculos en veredas para asistencia de personas con ceguera."

ALUMNO: CARRANZA, LUCAS FEDERICO

DIRECTOR: TRABES, EMANUEL.

FECHA: NOVIEMBRE 2022

DEPARTAMENTO DE ELECTRÓNICA

Introducción:

- Según la Organización Mundial de la Salud, en el mundo existen 1300 millones de personas con discapacidad visual.

Objetivos:

Generales:

- Del proyecto: Realizar los primeros pasos hacia un dispositivo que pueda facilitar la movilidad en espacios exteriores a personas con discapacidad visual.
- Del alumno: Extracción de información de alta abstracción de la escena.

Principal:

- Desarrollo de un sistema que detecten los obstáculos que se encuentren en el camino, particularmente pozos y obstrucciones.

Especifico:

- Realizar una evaluación del desempeño del detector de obstáculos. Se persigue obtener un "mAP" de 70 %.

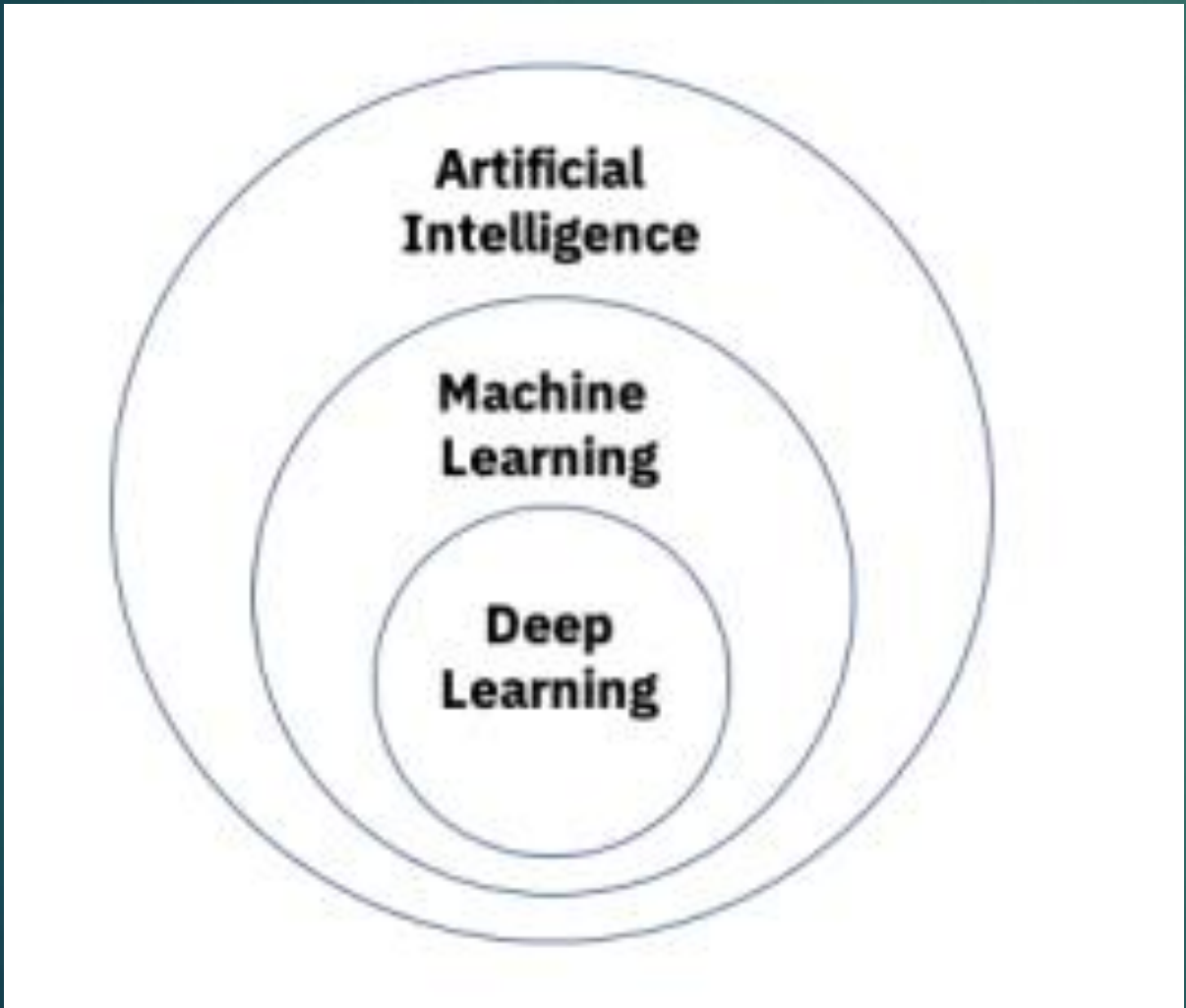
Estado del arte

- Tecnologías en el mercado:

Métodos de detección:

- Método de correlación.
- Bag of visual words.
- Comparación de histogramas.
- Inteligencia Artificial: Deep Learning.

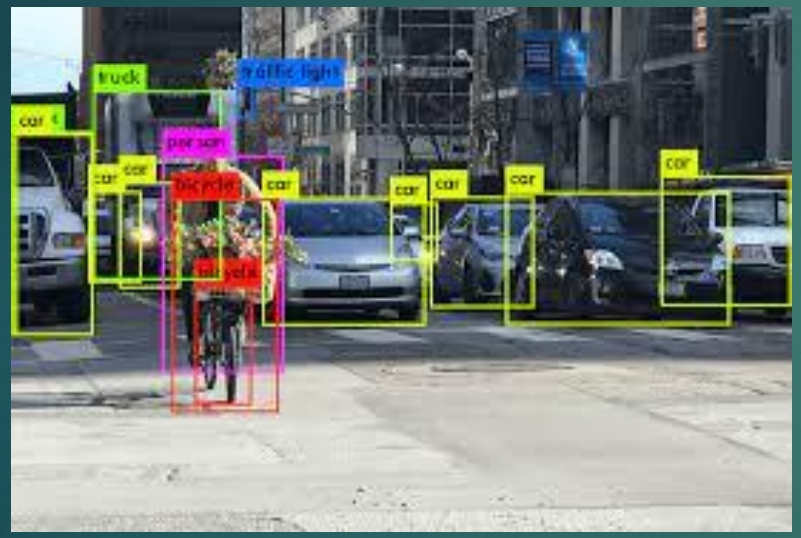
Inteligencia Artificial.



- Machine learning:
  - Aprendizaje
    - Supervisado
    - No supervisado
    - Por refuerzo
- Deep Learning
  - Redes neuronales.
- Machine Learning vs Deep Learning.

Deep Learning

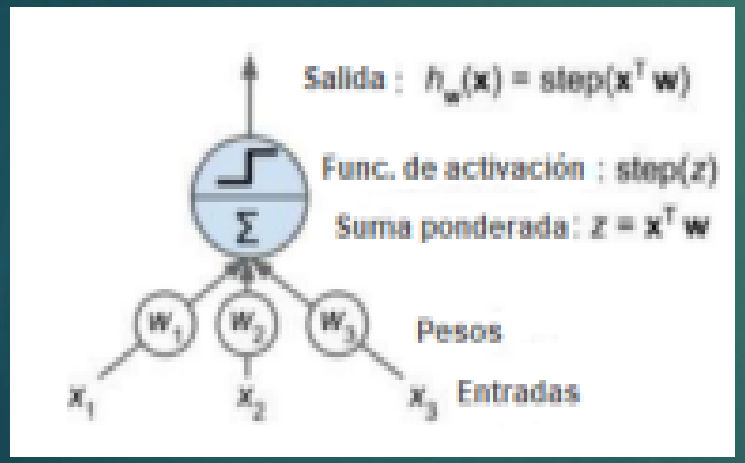
- Aplicaciones



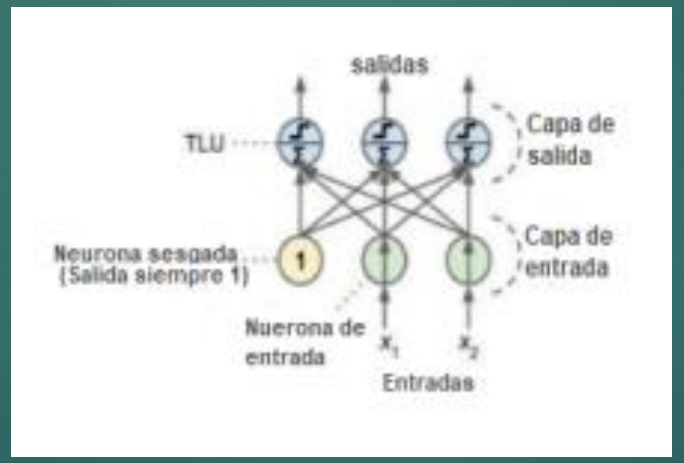
YouTube

Redes neuronales

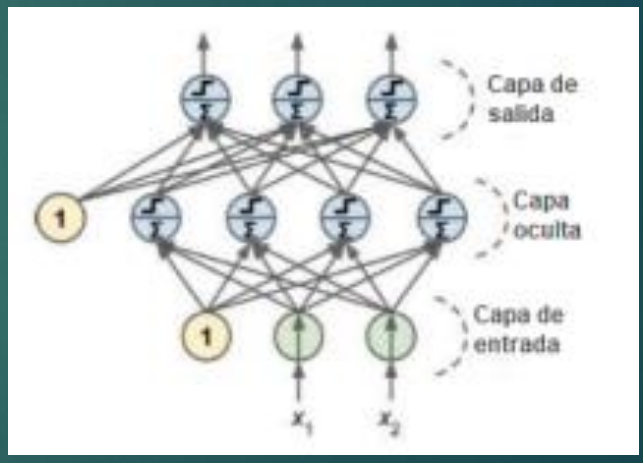
- **Definición:** Sistema computacional diseñado con el objetivo de simular la capacidad de procesamiento de información que realiza el cerebro humano.



Unidad lógica de umbral (Neurona artificial)



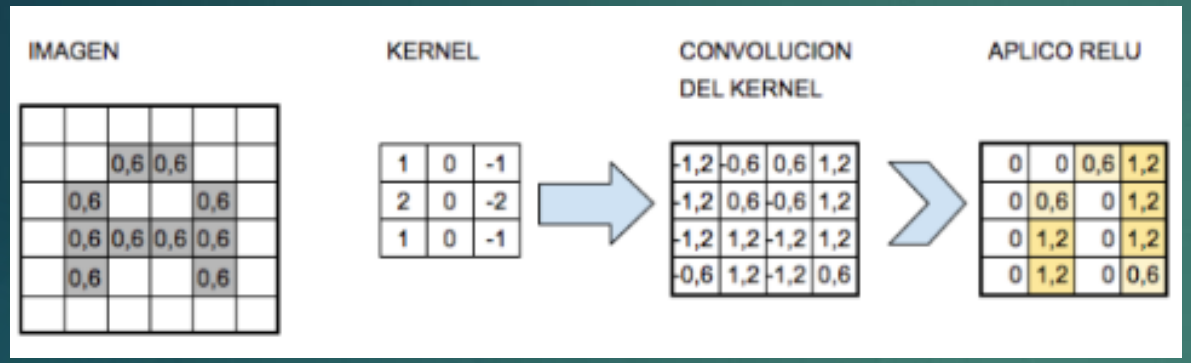
Perceptrón (Capa)



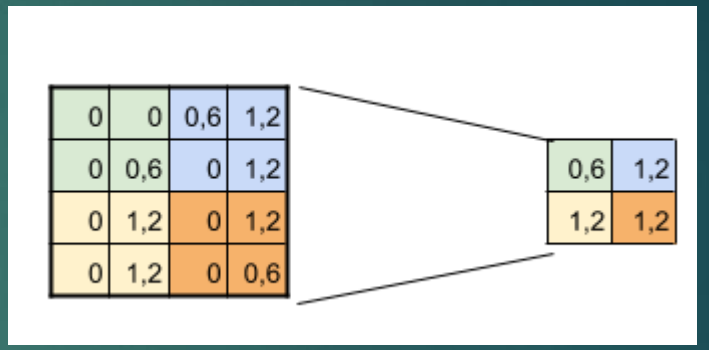
Perceptrones multicapa (capas interconectadas)

Redes neuronales convolucionales

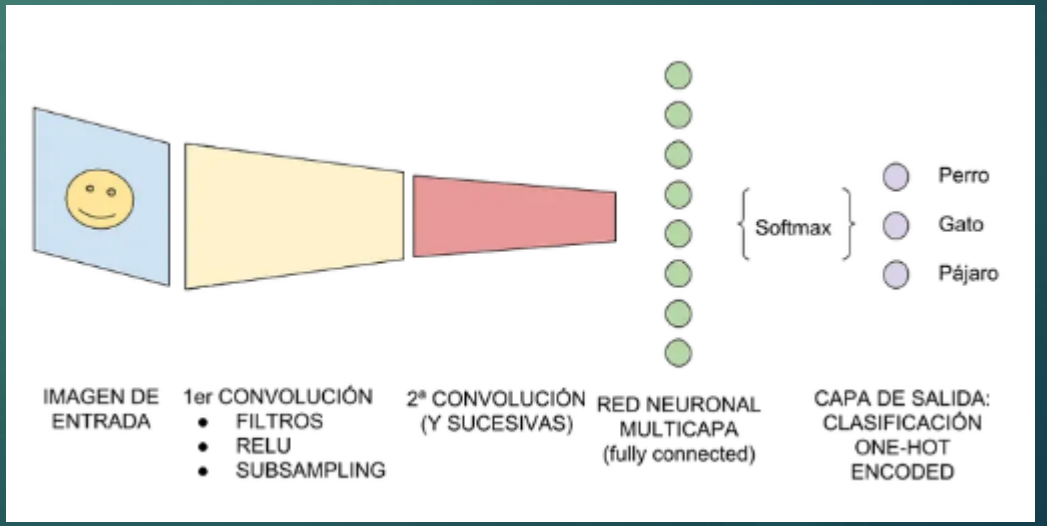
Convolución y función de activación



Sub muestreo

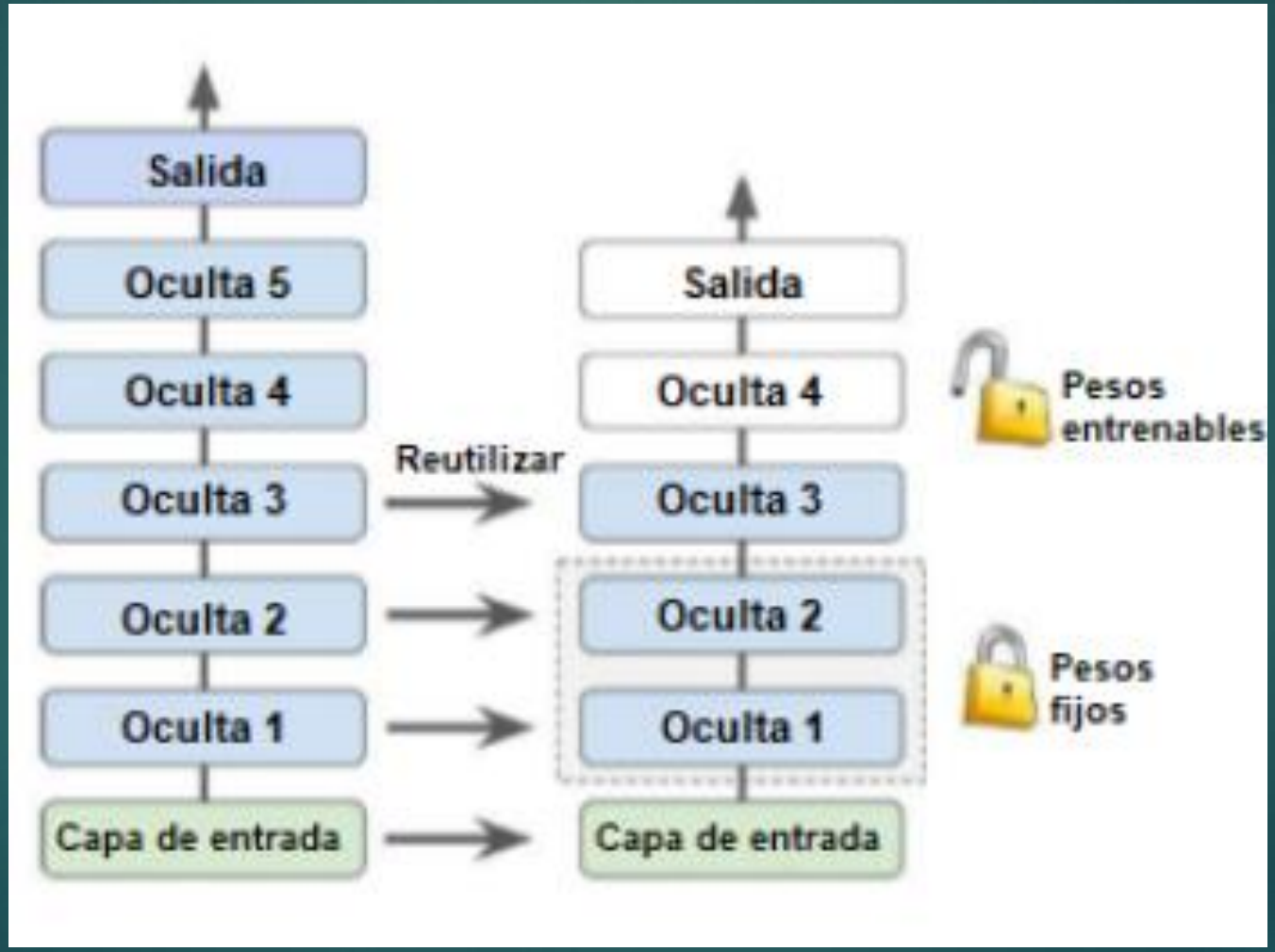


Aplanamiento y capa de salida

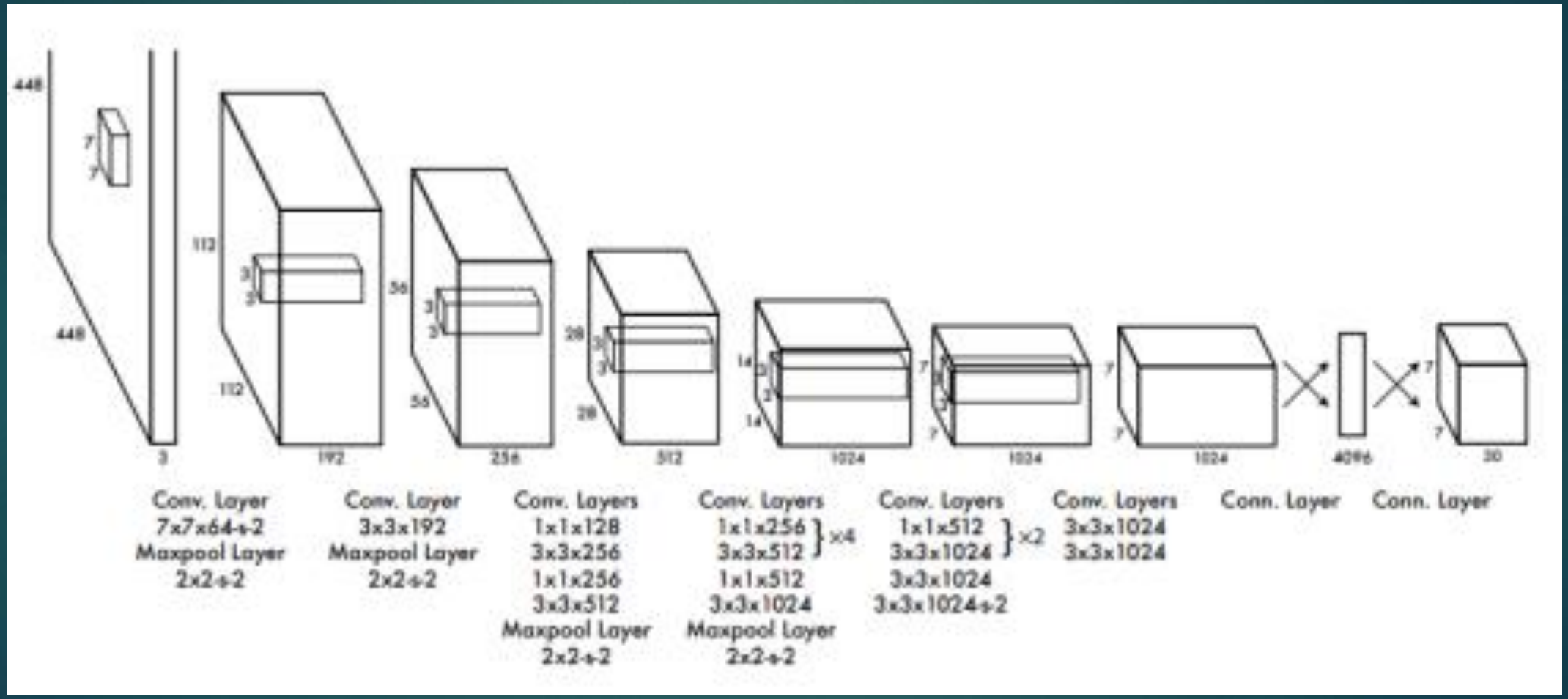


Proceso de entrenamiento

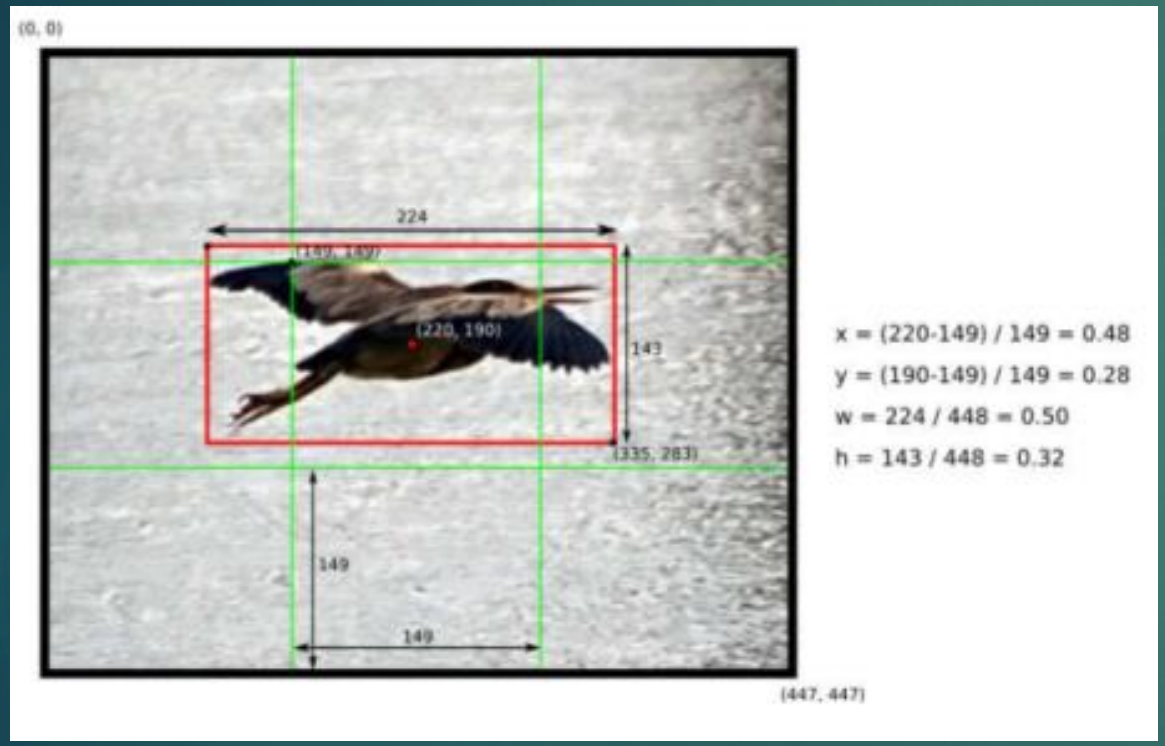
Transfer Learning

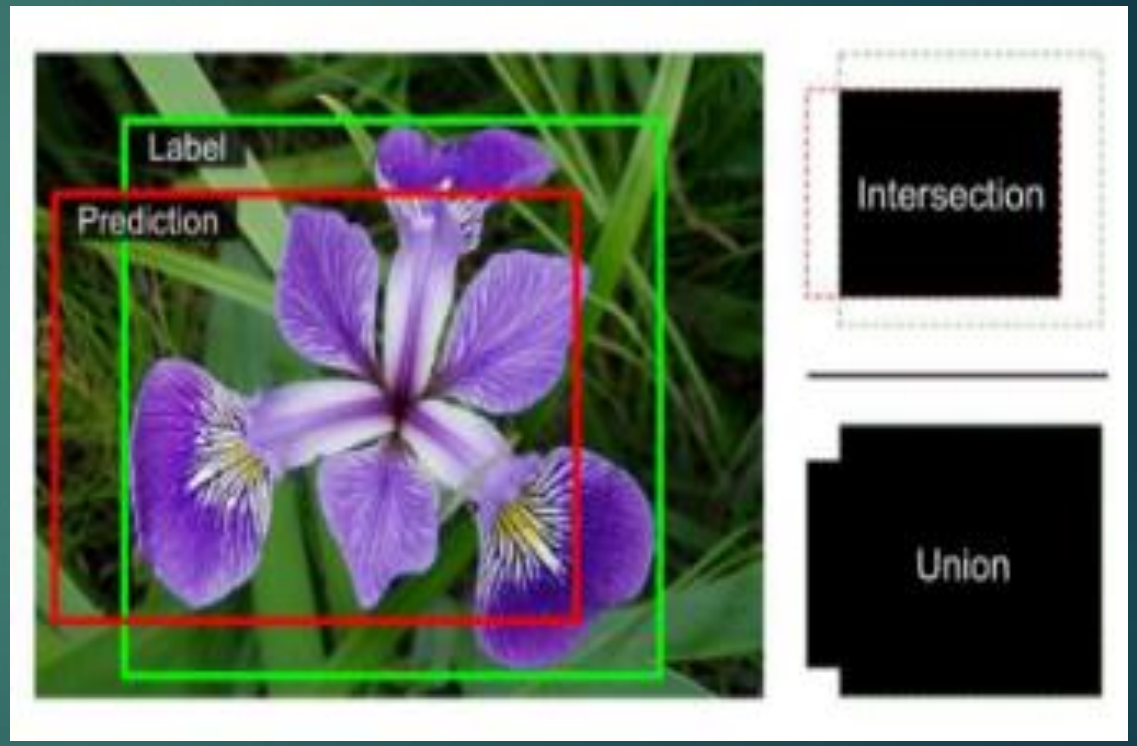


YOLO (You Only Look Once)



YOLO - Funcionamiento





YOLO - Beneficios

- Alta tasa de velocidad.
- Yolo es más general, supera a otros métodos cuando hay que generalizar partiendo de imágenes reales a otras áreas como el arte.
- Yolo accede a toda la imagen para realizar predicciones.
- Aprende representaciones generalizables de objetos.

Función de perdida.

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

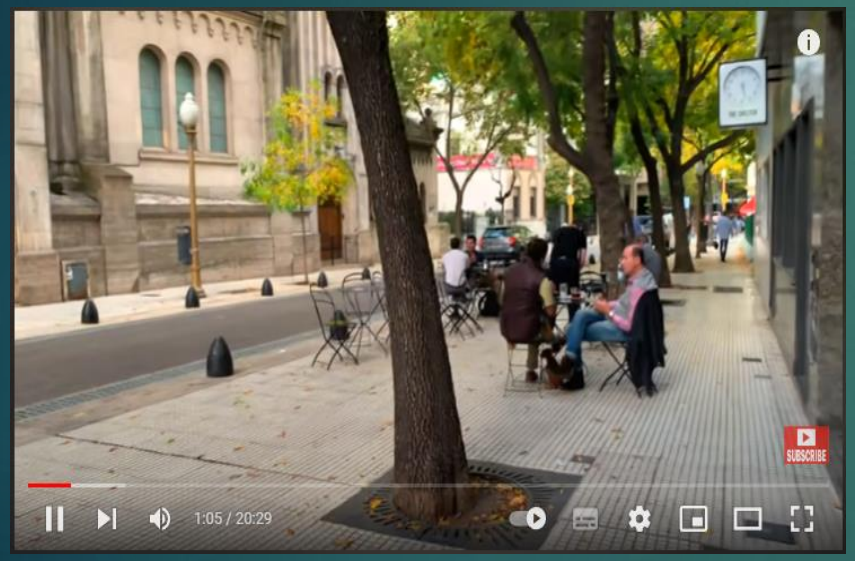
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} L_i^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

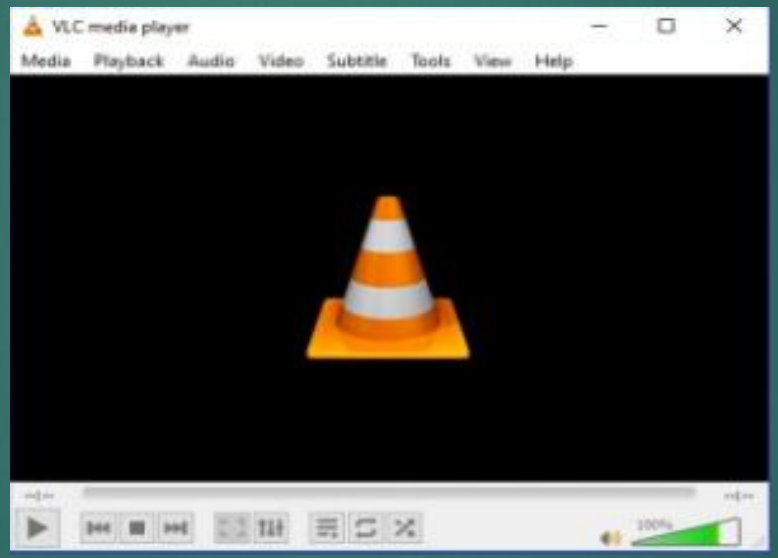
Desarrollo del sistema de detección - Dataset

Dataset – Herramientas utilizadas

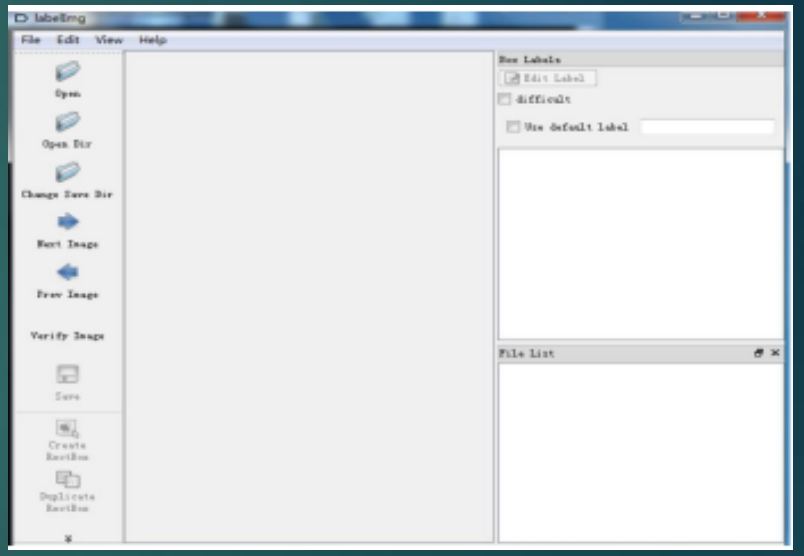
Walking Tour



VLC

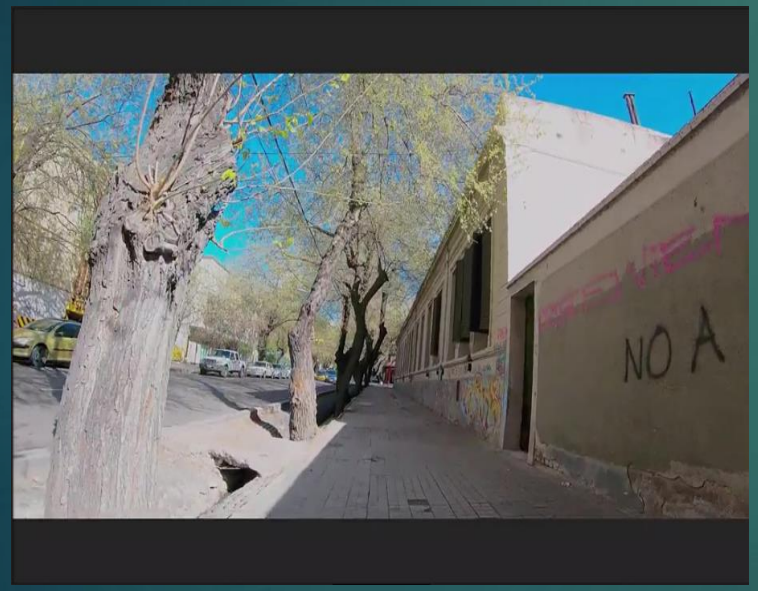


LabelImg



Dataset.

Imagen



Anotación

```
<annotation>
        <folder>Fotos_posos</folder>
        <filename>095.jpg</filename>
        <path>/home/luquitas/Escritorio/Veredas_Rotas/Fotos_posos/095.jpg</path>
        <source>
                <database>Unknown</database>
        </source>
        <size>
                <width>1920</width>
                <height>1080</height>
                <depth>3</depth>
        </size>
        <segmented>0</segmented>
        <object>
                <name>pozo</name>
                <pose>Unspecified</pose>
                <truncated>0</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>433</xmin>
                        <ymin>806</ymin>
                        <xmax>715</xmax>
                        <ymax>941</ymax>
                </bndbox>
        </object>
</annotation>
```

Bounding Box

Google Colab

Entorno

+ Código + Texto

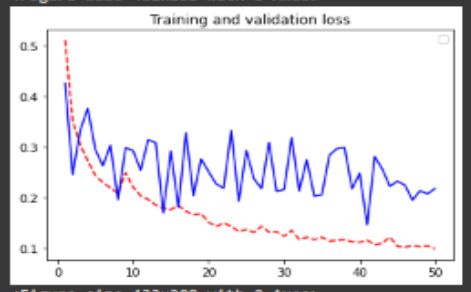
Graficas entrenamiento

```
[ ] history_dict = history.history
    print(history_dict.keys())
```

```
dict_keys(['val_loss', 'loss'])
```

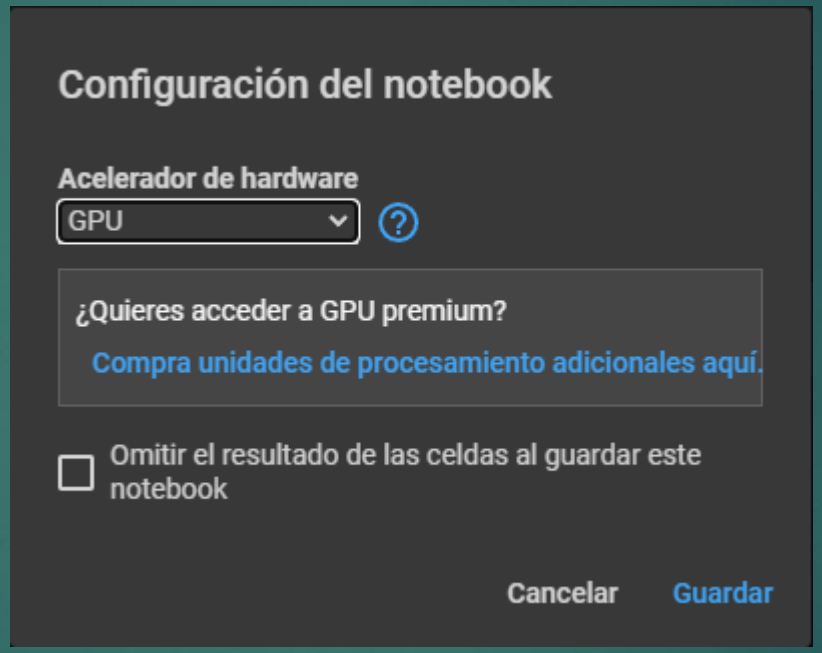
```
[ ] loss     = history.history['loss']
    val_loss = history.history['val_loss']

    epochss    = range(1,len(loss)+1,1)

    plt.plot ( epochss,     loss, 'r--' )
    plt.plot ( epochss, val_loss ,  'b' )
    plt.title ('Training and validation loss' )

    plt.legend()
    plt.figure()
```

```
WARNING:matplotlib.legend:No handles with labels found to put in legend.
<Figure size 432x288 with 0 Axes>
```

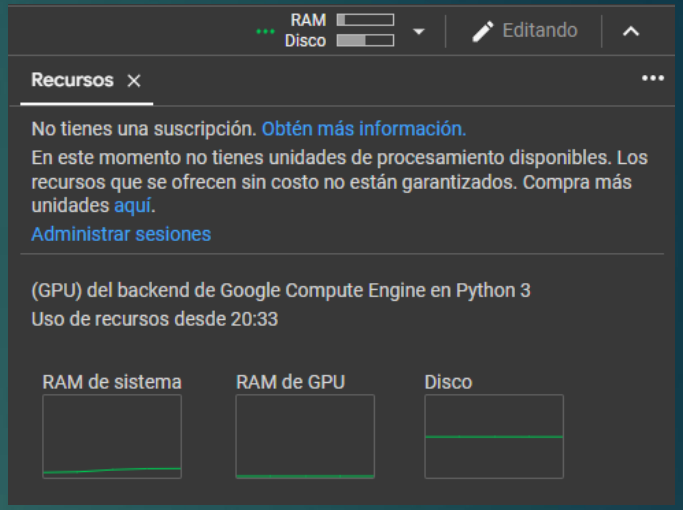



```
<Figure size 432x288 with 0 Axes>
```

Configuración



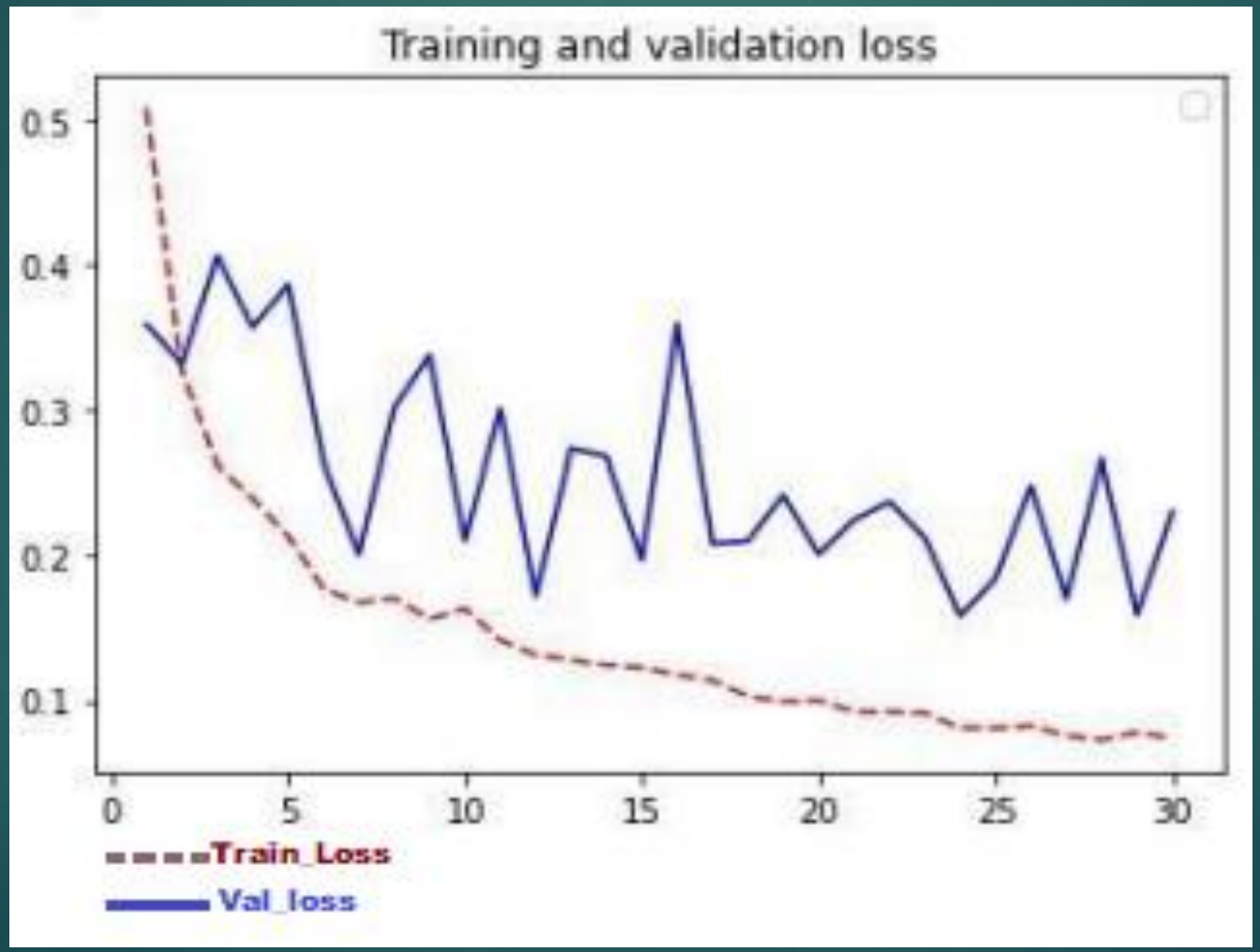
Recursos



Explicación del algoritmo

TensorFlow y Keras	• Keras: APPI de alto nivel. • TensorFlow: Biblioteca de código abierto.
Dataset	• Google drive.
Pesos pre entrenados	• Git hub.
Configuración del entrenamiento	• Call-backs. • Optimizador.

Testeo del sistema – Grafica de entrenamiento.



Metrica de rendimiento – Promedio medio de precisión (mAP)

- Una vez entrenada la red se calcula el mAP llegando a un resultado de 71,1% .

```
    if(len(true_bbs) <=  len(detected_bbs)):
      print("entro")
      if(len(detected_bbs) < 1):
          print("salgo")
          continue;
      #saca el iou de todos los bbs_true con el primer bb de la detectada
      for true_bb in true_bbs:
        best_iou = bbox_iou(true_bb, detected_bbs[0])
        best_bb = detected_bbs[0]
        print("best_iou=", best_iou)
        #compara el bbs_true actual con todos los detectados y saca el iou
        for detected_bb in detected_bbs:
          iou = bbox_iou(true_bb, detected_bb)
          #Si el iou es mejor al calculado anteriormente lo reemplaza
          if(iou > best_iou):
            best_iou = iou
            best_bb = detected_bb
        print("best_iou=", best_iou)
        #si es mayor al umbral suma a un true positive, sino es un false positive
        if (best_iou > 0.2):
          true_positive+=1
        else:
          false_positive+=1

          print("true=", true_positive)
          print("false=", false_positive)



print("true positive=", true_positive)
print("false positive=", false_positive)
print("~~~~~~~~~~~~~~~~~~mAP~~~~~~~~~~~~~~~~~~")
print("mAP=",(true_positive/(true_positive+false_positive)))
```

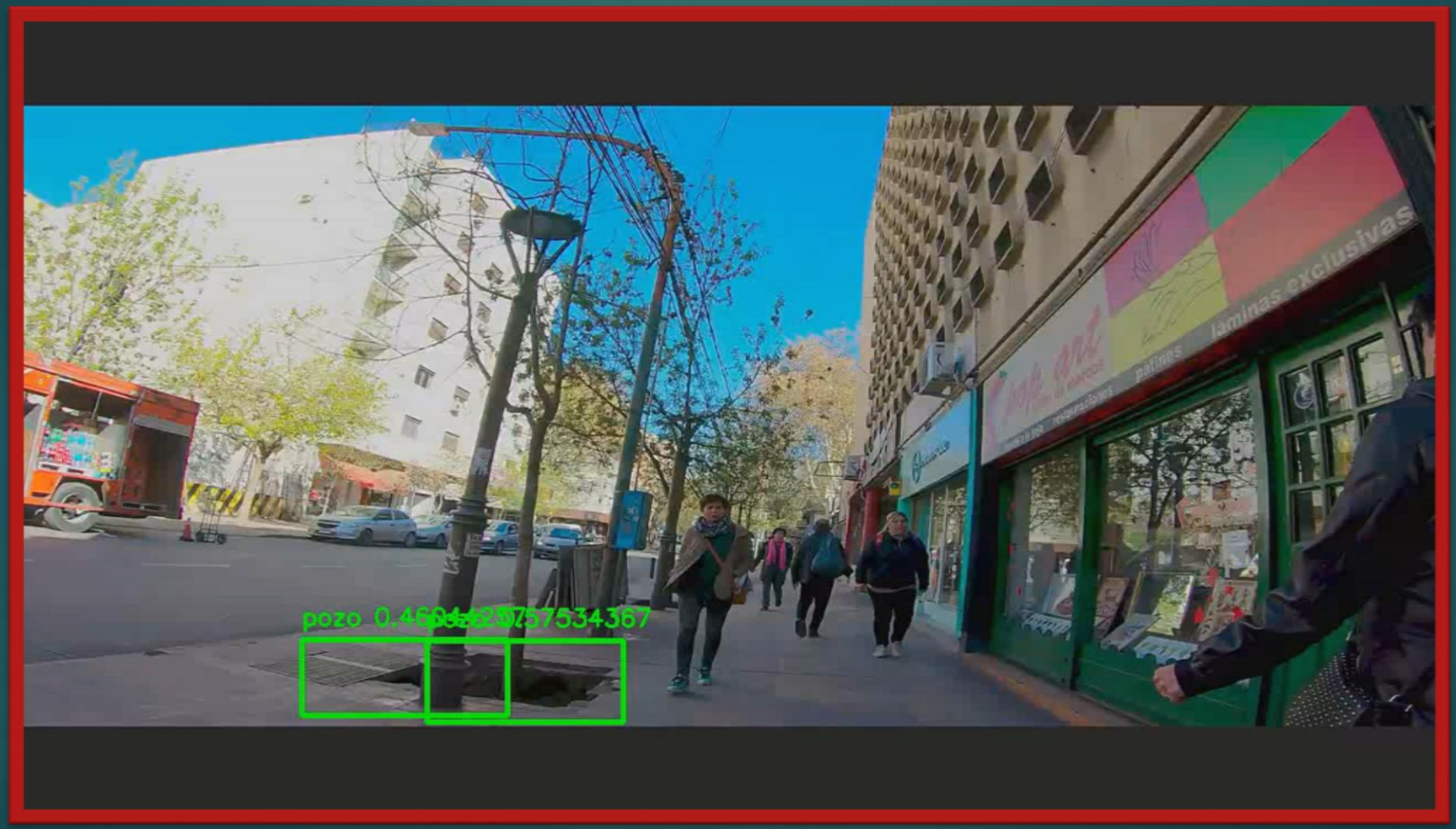
Formula

$$\mathbf{mAP} = \sum_{i=1}^{n} \frac{TP}{TP + FP}.$$

Resultado

```
true positive= 32
false positive= 13
mAP = 0.7111111111111111
```

Implementación en un video real



Conclusiones.

Mejoras futuras:

- Implementación en dispositivos móviles.
- TensorFlow Lite.
- Android Studio.
- Aumentar el volumen y calidad del dataset.
- Web Scraping.
- Sitema de almacenamiento masivo.

Agradecimientos.

¿Preguntas?