Visión por computadora
Reconocimiento de objetos

Mecatrónica Aplicada
Sistemas Mecatrónicos

Ing. Nelson Ariel Fortunatti

Preparando un nuevo entorno en Anaconda

Install TensorFlow

Consola --> pip install tensorflow

Creamos un nuevo environment e instalamos tensorflow.

Anaconda --> conda create -n tf tensorflow==1.15.0

Anaconda --> conda activate tf

LINK REF --> https://docs.anaconda.com/anaconda/user-guide/tasks/tensorflow/

Preparando un nuevo entorno en Anaconda

Install OpenCV

Consola --> pip install opencv-python

Anaconda --> conda install -c conda-forge opencv

LINK REF -->
https://anaconda.org/conda-forge/opencv

Preparando un nuevo entorno en Anaconda

Install Keras

Consola --> pip install keras

Anaconda --> conda install -c conda-forge keras

LINK REF -->
https://anaconda.org/conda-forge/keras

Preparando un nuevo entorno en Anaconda

Install ImageAI

Consola --> pip install imageAI

Anaconda --> conda install -c powerai imageai

LINK REF --> https://anaconda.org/powerai/imageai

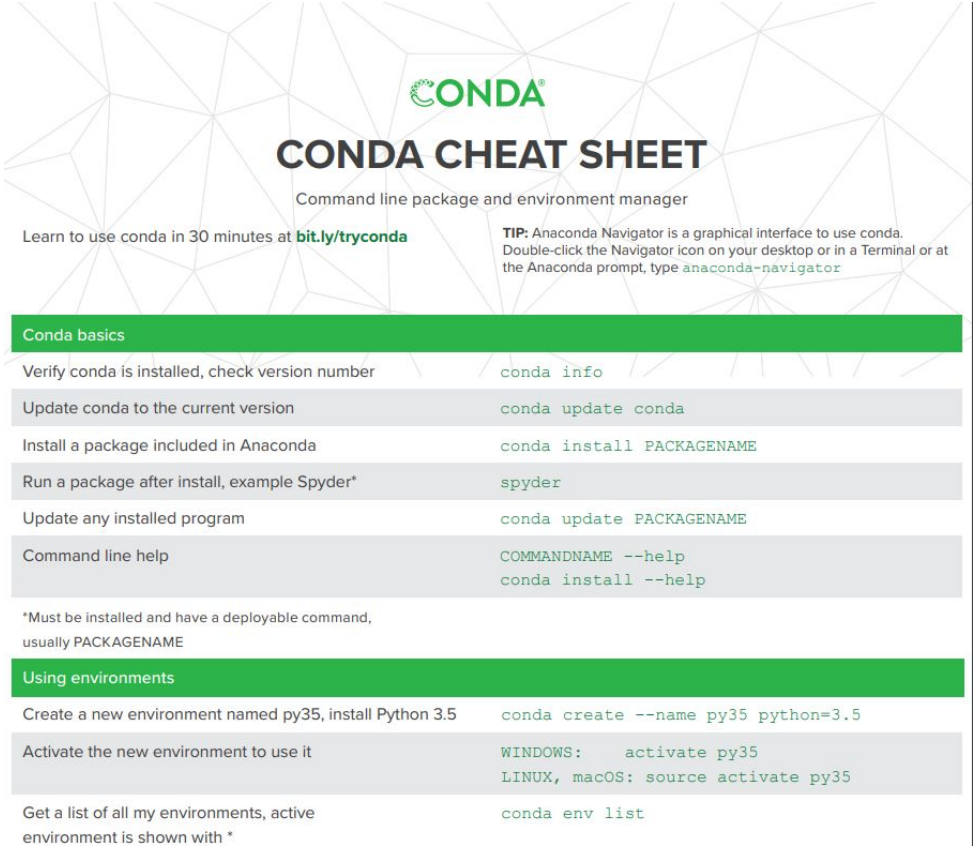
Preparando un nuevo entorno en Anaconda

Install LXML

Anaconda --> conda install -c anaconda lxml

LINK REF --> https://anaconda.org/anaconda/lxml

Conda Cheatsheet



CONDA

CONDA CHEAT SHEET

Command line package and environment manager

Learn to use conda in 30 minutes at **bit.ly/tryconda**

TIP: Anaconda Navigator is a graphical interface to use conda. Double-click the Navigator icon on your desktop or in a Terminal or at the Anaconda prompt, type `anaconda-navigator`

Conda basics	
Verify conda is installed, check version number	`conda info`
Update conda to the current version	`conda update conda`
Install a package included in Anaconda	`conda install PACKAGENAME`
Run a package after install, example Spyder*	`spyder`
Update any installed program	`conda update PACKAGENAME`
Command line help	`COMMANDNAME --help` `conda install --help`

*Must be installed and have a deployable command, usually PACKAGENAME

Using environments	
Create a new environment named py35, install Python 3.5	`conda create --name py35 python=3.5`
Activate the new environment to use it	`WINDOWS: activate py35` `LINUX, macOS: source activate py35`
Get a list of all my environments, active environment is shown with *	`conda env list`

https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c49671267e40c689e0bc00ca/conda-cheatsheet.pdf

Tensor Flow

TensorFlow es una plataforma de código abierto de extremo a extremo para el aprendizaje automático.

Cuenta con un ecosistema integral y flexible de herramientas, bibliotecas y recursos de la comunidad que les permite:

- A los investigadores impulsar un aprendizaje automático innovador.
- A los desarrolladores, compilar e implementar con facilidad aplicaciones con tecnología de AA.

https://www.tensorflow.org/

Keras

- Keras es un framework de deep learning de Código Abierto escrita en Python.
- Corre sobre TensorFlow.
- Keras hace más simple correr nuevos experimentos.
- Está especialmente diseñada para posibilitar la experimentación en más o menos poco tiempo con redes de Aprendizaje Profundo.
- Sus fuertes se centran en ser amigable para el usuario, modular y extensible.

https://keras.io/

OpenCV

OpenCV (Open Source Computer Vision Library) es una biblioteca de código abierto y machine learning.

Está creada para proveer una infraestructura común entre las aplicaciones de visión por computadora y los productos comerciales.

Al poseer una licencia BSD, se puede utilizar tanto para desarrollos abiertos como cerrados.

https://opencv.org/

ImageAI

ImageAI es una biblioteca de visión por computadora escrita en Python.

Utiliza TensorFlow, Keras, OpenCV como base y brinda una API sencilla para operar en el reconocimiento de objetos.

Es utilizado por miles de desarrolladores, es el primer paso para la inserción en el área de visión por computadora.

Como desventaja aún no posee todas las características que se podrían aprovechar haciendo uso de TensorFlow y Keras directamente.

http://imageai.org/

Computer vision tasks (1)

- **Image Classification**: Predict the type or class of an object in an image.
    - *Input*: An image with a single object, such as a photograph.
    - *Output*: A class label (e.g. one or more integers that are mapped to class labels).

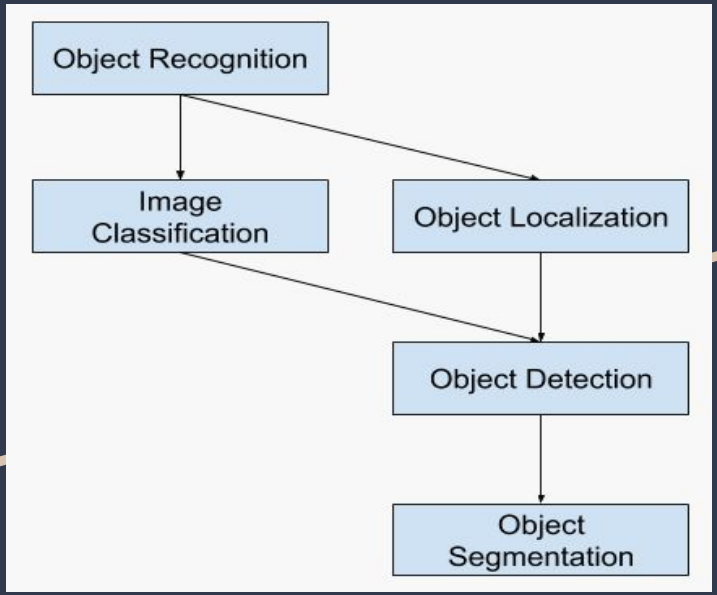
Computer vision tasks (2)

- **Object Localization**: Locate the presence of objects in an image and indicate their location with a bounding box.
    - *Input*: An image with one or more objects, such as a photograph.
    - *Output*: One or more bounding boxes (e.g. defined by a point, width, and height).

Computer vision tasks (3)

- **Object Detection**: Locate the presence of objects with a bounding box and types or classes of the located objects in an image.
    - *Input*: An image with one or more objects, such as a photograph.
    - *Output*: One or more bounding boxes (e.g. defined by a point, width, and height), and a class label for each bounding box.

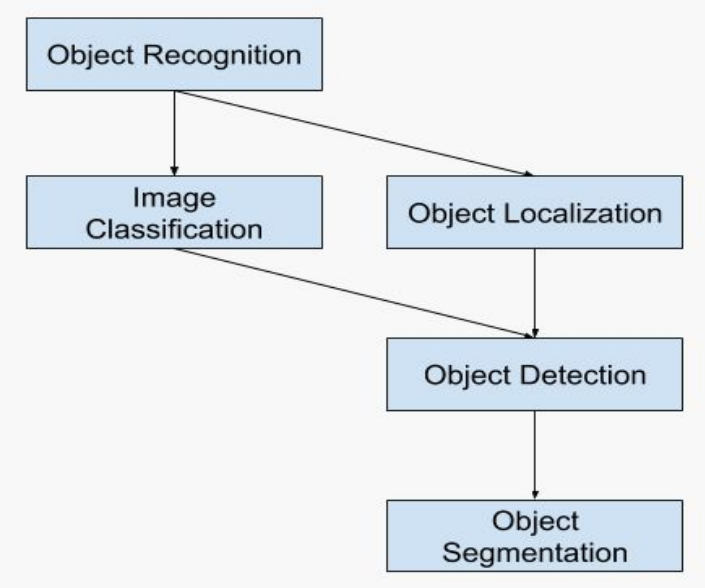
Object recognition is a group of computer vision tasks.



- **Image classification**: Algorithms produce a list of object categories present in the image.

- **Single-object localization**: Algorithms produce a list of object categories present in the image, along with an axis-aligned bounding box indicating the position and scale of one instance of each object category.

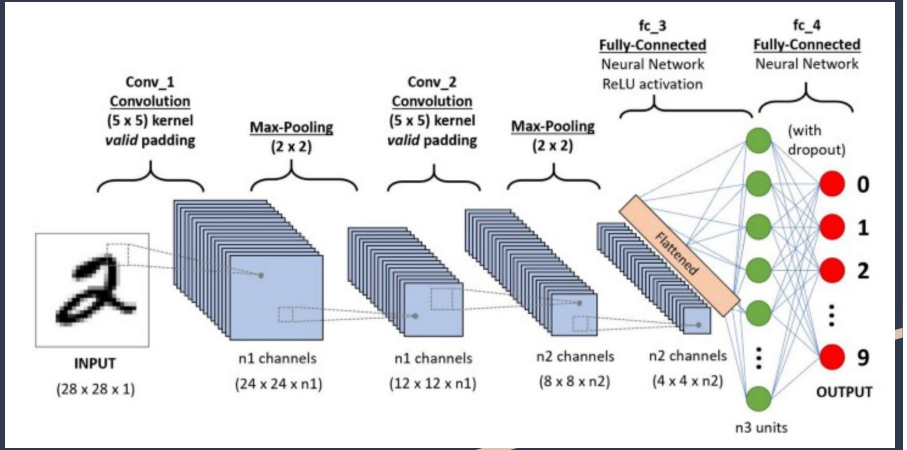
Object recognition is a group of computer vision tasks.



- **Object detection**: Algorithms produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

- **Object segmentation:** “object instance segmentation” or “semantic segmentation,” where instances of recognized objects are indicated by highlighting the specific pixels of the object instead of a coarse bounding box.

CNN



A **Convolutional Neural Network** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

The preprocessing required in a CNN is much lower as compared to other classification algorithms.

While in primitive methods filters are hand-engineered, with enough training, CNNs have the ability to learn these filters/characteristics.

R-CNN Model

"*Region-Based Convolutional Neural Network*," developed by Ross Girshick.

R-CNN

- **Module 1: Region Proposal**. Generate and extract category independent region proposals, e.g. candidate bounding boxes.
- **Module 2: Feature Extractor**. Extract feature from each candidate region, e.g. using a deep convolutional neural network.
- **Module 3: Classifier**. Classify features as one of the known class, e.g. linear SVM classifier model.

R-CNN Model

"*Region-Based Convolutional Neural Network*," developed by Ross Girshick.

Fast R-CNN

- **Training is a multi-stage pipeline**. Involves the preparation and operation of three separate models.
- **Training is expensive in space and time**. Training a deep CNN on so many region proposals per image is very slow.
- **Object detection is slow**. Make predictions using a deep CNN on so many region proposals is very slow.

R-CNN Model

"*Region-Based Convolutional Neural Network*," developed by Ross Girshick.

Faster R-CNN

- **Module 1: Region Proposal Network**. Convolutional neural network for proposing regions and the type of object to consider in the region.
- **Module 2: Fast R-CNN**. Convolutional neural network for extracting features from the proposed regions and outputting the bounding box and class labels.

YOLO Model Family

"*You Only Look Once*," developed by Joseph Redmon.

The approach involves a single neural network trained end to end that takes a photograph as input and predicts bounding boxes and class labels for each bounding box directly.

The technique offers lower predictive accuracy.

The model works by first splitting the input image into a grid of cells, where each cell is responsible for predicting a bounding box if the center of a bounding box falls within it.

Reconocimiento de objetos en imágenes

```python
from imageai.Detection import ObjectDetection

detector = ObjectDetection()

#model_path = "./models/yolo-tiny.h5"
model_path = "./models/yolo.h5"
input_path = "./input/vehiculos.jpg"
output_path = "./output/resul-output.jpg"
print("Initializing: setModelTypeAsYOLOv3")
detector.setModelTypeAsYOLOv3()
#detector.setModelTypeAsTinyYOLOv3()
print("Initializing: setModelPath(model_path)")
detector.setModelPath(model_path)
print("Initializing: loadModel()")
detector.loadModel()
print("Initializing: detectObjectsFromImage(...)")
detection = detector.detectObjectsFromImage(input_image=input_path,
                                            output_image_path=output_path)
print("Results:")
for eachItem in detection:
    print(eachItem["name"] , " : ", eachItem["percentage_probability"])
print(detection)
```

Resultados

```
Results:
traffic light  :  84.35483574867249
truck  :  56.47655725479126
truck  :  57.12488293647766
truck  :  68.44587922096252
motorcycle  :  63.3245944976806 64
car  :  90.2835488319397
car  :  91.33853316307068
car  :  92.5764262676239
car  :  92.58168935775757
car  :  93.20821762084961
car  :  93.5617208480835
car  :  94.91747617721558
car  :  95.1159119606018
car  :  95.29452323913574
car  :  95.35254240036011
car  :  95.56770920753479
car  :  95.85888385772705
car  :  96.26147150993347
car  :  96.47133350372314
car  :  96.82047963142395
car  :  96.85346484184265
car  :  97.06475138664246
car  :  98.26093912124634
car  :  98.84251356124878
```

Estructura de archivos

- **Object detection**: root folder
- **Models**: stores pre-trained model
- **Input**: stores image file on which we want to perform object detection
- **Output**: stores image file with detected objects

Modelos pre-entrenados

Descargas:

RetinaNet Model - resnet50_coco_best_v2.0.1.h5

YOLOv3 Model - yolo.h5

TinyYOLOv3 Model - yolo-tiny.h5

Bibliografía

https://machinelearningmastery.com/object-recognition-with-deep-learning/

https://stackabuse.com/object-detection-with-imageai-in-python/

https://imageai.readthedocs.io/en/latest/detection/index.html

Reconocimiento de nuevos objetos

Entrenamiento Red Neuronal

Bibliografía

https://medium.com/deepquestai/train-object-detection-ai-with-6-lines-of-code-6d087063f6ff

https://medium.com/deepquestai/object-detection-training-preparing-your-custom-dataset-6248679f0d1d

https://medium.com/@sanghuynh_73086/how-to-install-labelimg-in-windows-with-anaconda-c659b27f0f

LabelIMG Program

https://github.com/tzutalin/labelImg