**Proyectos EL 7008 - Semestre Primavera 2019**

1. **Detección y Reconocimiento de Objectos usando Tiny-YOLO en teléfonos (7x)**

**Tutor: Patricio Loncomilla**

El objetivo de este proyecto es generar una versión de Tiny-YOLO v3 que funcione en teléfonos (Android o IoS). En el caso de Android, se recomienda usar tensorflow en Java, ya que es la opción más sencilla. Dado que la red ya está entrenada, simplemente debe ser cargada desde la aplicación en Android (no es necesario entrenar nada). El sistema final debe mostrarse funcionando en un teléfono real, detectando los objetos capturados por la cámara trasera. Se usará como base un código que permite usar otro tipo de detector (SSD-MobileNet), y deberá ser modificado de modo que pueda usar Tiny-YOLO v3. Finalmente, se debe comparar la performance de las dos redes.

Referencias:

- YOLO: Real-Time Object Detection. <https://pjreddie.com/darknet/yolo/>

- Jonathan Huang et al. Speed/accuracy trade-offs for modern convolutional object detectors. https://arxiv.org/pdf/1611.10012.pdf

- https://github.com/tensorflow/tensorflow/tree/master/tensorflow/examples/android

1. **Rocky-YOLO: Detección rocas utilizando YOLO (7x)**

**Tutor: Patricio Loncomilla**

El objetivo de este trabajo es modificar YOLOv2/YOLOv3 de modo que pueda detectar objetos usando elipses en vez de rectángulos. Esto puede ser útil para, por ejemplo, detectar rocas en imágenes. Se debe elegir una implementación de YOLO existente que sea entrenable y pueda leer pesos pre-entrenados. Luego se debe modificar la última capa de modo que, en vez de considerar el ancho y alto de la ventana, considere el ancho y alto de la elipse, y su inclinación (coseno y seno del ángulo). También debe modificarse la función de pérdida y el lector de los datos de entrenamiento para que pueda leer el ángulo de la elipse como parámetro extra. Hay dos bases de datos disponibles: (1) Imágenes de frente de túnel a escala, y (2) imágenes de rocas lunares (opcional). En el caso (1), las imágenes ya están anotadas para elipses. En el caso (2), el alumno debe encontrar una estrategia para transformar las anotaciones existentes (segmentación semántica) a elipses.

Referencias:

- YOLO: Real-Time Object Detection. https://pjreddie.com/darknet/yolo/

- Kenzo Lobos, Patricio Loncomilla, Javier Ruiz-del-Solar. Rocky-YOLO: Detección de rocas usando deep learning para aplicaciones en minería. SIMIN 2019.

- https://www.kaggle.com/romainpessia/artificial-lunar-rocky-landscape-dataset

1. **Reconocimiento de instancias de objetos particulares (7x)**

**Tutor: Patricio Loncomilla**

El objetivo de este trabajo es probar un reconocedor de instancias de objetos particulares programado en Pytorch, y poder hacerlo funcionar usando ROS (para el robot Pepper con computador externo). El detector de instancias de objetos se detalla en el siguiente artículo: "YoloSPoC: Recognition of multiple object instances by using yolo-based proposals and deep SPoC-based descriptors". El reconocedor de objetos básicamente consiste en una etapa de generación de object proposals (usando YOLOv3), seguida por un extractor de características (una red convolucional) y un clasificador nearest-neighbor, el cual asigna el vector de características a uno de los objetos de referencia existentes, sin que se necesite ningún tipo de entrenamiento.

Referencias:

- Patricio Loncomilla, Javier Ruiz del Solar. YoloSPoC: Recognition of multiple object instances by using yolo-based proposals and deep SPoC-based descriptors. RoboCup 2019.

- YOLO: Real-Time Object Detection. https://pjreddie.com/darknet/yolo/

- CNN Image Retrieval in PyTorch: Training and evaluating CNNs for Image Retrieval in PyTorch. <https://github.com/filipradenovic/cnnimageretrieval-pytorch>

1. **Re-entrenamiento Red neuronal SSD para detección de objetos**

**Tutor: Hans Starke**

**Alumno: Mauricio Antonelli**

El proyecto a desarrollar consiste en re-entrenar una red pequeña SSD basada en la arquitectura MobileNetV2 para poder detectar un set de 5 objetos, esto con el objetivo de implementar un método simple de detección de objetos que funcione en tiempo real y que podría ser utilizado en la RoboCup.

Paper: https://arxiv.org/abs/1512.02325

Github: https://github.com/qfgaohao/pytorch-ssd

1. **Deep face recognition para robots móviles**

**Tutor: Hans Starke**

**Alumno: Juan Pablo Cáceres**

Implementar sistema ArcFace (Deep Face Recognition) de tal manera que funcione en robots Pepper y Bender. Posteriormente, se verá la posibilidad de modificar la arquitectura por una más liviana.

Referencia: J. Deng, J. Guo, N. Xue, S. Zafeiriou, “ArcFace: Additive Angular Margin Loss for Deep Face Recognition", arXiv, February 2019.

1. **Segmentación de muebles con YOLACT**

**Tutor: Hans Starke**

**Alumno: Javier Urrutia**

El objetivo de este proyecto es utilizar YOLACT para segmentar muebles de una imagen, lo que, por ejemplo, se puede aplicar como una fuente de información para robots de servicio, pues éstos deben trabajar en espacios interiores. Al final del proyecto se espera tener la red implementada en un nodo utilizable en ROS.

Paper: <https://arxiv.org/abs/1904.02689>

Github: <https://github.com/dbolya/yolact>

1. **Modificación Backbone de CenterNet**

**Tutor: Hans Starke**

**Alumno: Nicolás Marticorena**

El objetivo de este proyecto consiste en la mejora de la eficiencia de una CenterNet modificando el Backbone de esta misma, la idea principal consiste en probar distintas redes que sean mejores en términos de tamaño, tiempo de inferencia y mAP.

Papers: http://arxiv.org/abs/1904.07850

https://arxiv.org/abs/1909.00700

Github: https://github.com/xingyizhou/CenterNet

1. **Detector de lugar donde apunta una persona**

**Tutor: Hans Starke**

**Alumno: Lukas Pavez**

El objetivo del proyecto es implementar un detector de keypoints de la mano de una persona en una imagen. Esto se realiza en el marco de la competencia Robocup, donde se tienen pruebas donde se desea saber hacia donde apunta una persona, no se especifica el número de personas en la imagen, así que se tendrá el supuesto de que solo hay una.

Papers: <https://arxiv.org/pdf/1801.07698.pdf>

Github: https://github.com/deepinsight/insightface

https://github.com/TreB1eN/InsightFace\_Pytorch