Reconocimiento Visual con Deep Learning Detección de Objetos para el Reconocimiento de Montos

Prof. José M. Saavedra Prof. Auxiliar Cristóbal Loyola

noviembre 2021

1. Objetivo

El objetivo de esta tarea es entender, en forma práctica, el funcionamiento de detectores de objetos en imágenes, basados en modelos convolucionales, a través de una aplicación para el reconocimiento de montos manuscritos.

2. Descripción

El reconocimiento de montos manuscritos es un problema altamente desafiante debido a la alta variabilidad de escritura y al ruido presente en los documentos sobre los que se escribe. Para resolver este problema, los métodos tradicionales se enfocan en la segmentación de los diferentes símbolos presentes, para luego aplicar un clasificador sobre los segmentos reconocidos. Así, el problema de reconocimiento de una secuencia de dígitos se reduce a un problema de segmentación, que puede resultar mucho más complejo que el problema original.

De acuerdo con lo visto en clases, la detección de objetos consiste en localizar objetos de interés sobre una imagen. La localización consiste generalmente en predecir un rectángulo (a.k.a. *bounding box*) definido por un punto de referencia (x,y) y el ancho y alto del rectángulo. Así, no debería ser difícil darnos cuenta que el reconocimiento de montos puede reducirse a un problema de detección de objetos, en donde los objetos de interés son los dígitos.

Por lo anterior, en esta tarea se pide entrenar un modelo de detección de objetos y predecir el monto que aparece en una imagen de entrada. Algunos ejemplos de posibles imágenes de entrada se presentan en la Figura 1.

Para resolver este problema, deberán elegir alguno de los siguientes modelos:

- RetinaNet

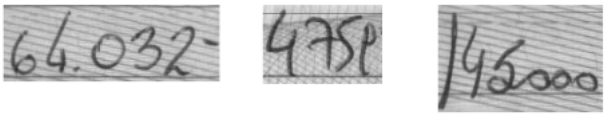

Figura 1: Ejemplos de imágenes de montos manuscritos

- TridentNet
- FCOS
- Yolo (versión ≥ 2)

Los estudiantes podrán hacer uso de alguna implementación ya existente de los modelos anteriores (en Pytoch o Tensorflow), sobre la que deberán realizar los ajustes necesarios y entrenar el modelo en el contexto de esta tarea. A continuación se mencionan algunos enlaces con implementaciones de los modelos indicados (pero pueden utilizar otras):

- `https://github.com/fizyr/keras-retinanet`
- `https://github.com/facebookresearch/detectron2/tree/master/projects/TridentNet/`
- `https://github.com/tianzhi0549/FCOS`

2.1. Dataset

Los datos de entrenamiento deben ser descargado de: `https://www.dropbox.com/s/yf6uhsez5f2rau0/orand-car-with-bbs.zip?dl=0` (passwd=cc7221). Dentro de la carpeta *training* se encontrarán 6370 anotaciones junto a sus correspondientes imágenes. Cada archivo de anotación corresponde a una imagen de la carpeta *imágenes*. El archivo de anotaciones consta de tantas filas como dígitos hay en la imagen, cada fila se compone de la clase del dígito junto con el *bounding box* correspondiente. Cada archivo de anotación es especificado bajo la siguiente sintaxis:

`<clase>: <x>, <y>, <w>, <h>`

donde:

- **clase**: es la clase del dígito con valores que van desde 0 hasta 9.
- **x**: es la coordenada x de la esquina superior izquierda del respectivo *bounding box*.
- **y**: es la coordenada y de la esquina superior izquierda del respectivo *bounding box*.
- **w**: es el ancho del respectivo *bounding box*.
- **h**: es alto del respectivo *bounding box*.

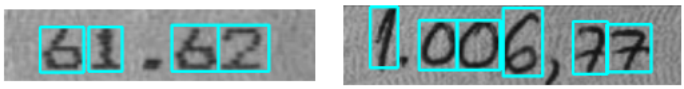

Figura 2: Ejemplos de anotaciones de las imágenes de montos.

En la Figura 2 se presentan dos ejemplos de imágenes de montos con sus respectivos *bounding boxes*, según lo indicado en sus correspondientes archivos de anotaciones.

Durante la predicción de rectángulos, se observará que el modelo detecta múltiples rectángulos sobre un mismo dígito, lo que producirá errores en la formación del monto reconocido. Para reducir este error, deberá usar la estrategia de *non-maximum suppression* de modo que se tenga una sola detección por dígito.

Como el objetivo final es predecir el monto presente en la imagen, la evaluación del método consiste en determinar el porcentaje de reconocimiento de montos correctos (accuracy). Para este fin, deberán utilizar las imágenes de test (carpeta *test*) que se adjuntan al dataset de entrenamiento, descrito anteriormente.

Algunos ejemplos de la detección de dígitos se muestra en la Figura 3.

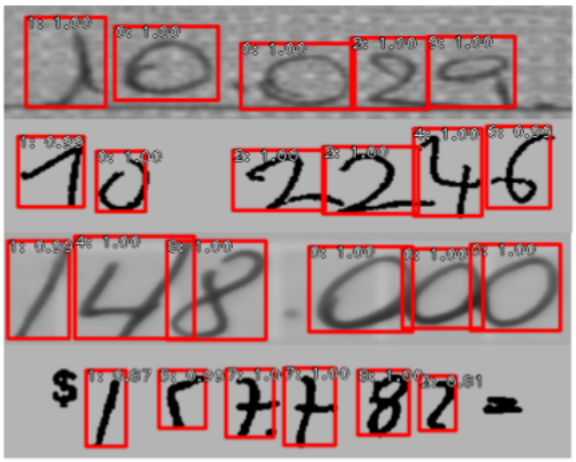

Figura 3: Ejemplos de detección de dígitos en imágenes de montos.

2.2. Restricciones

1. Grupos de a lo más 2 estudiantes.
2. El número de iteraciones y tamaños del batch lo deben configurar ustedes, de modo que puedan alcanzar el máximo desempeño.
3. Es obligatorio presentar un informe sobre el desarrollo del trabajo. Tareas sin informes no serán evaluadas.

3. Informe

Se deberá presentar un informe tipo paper, conteniendo las siguientes secciones:

1. **Abstract o Resumen**: es el resumen del trabajo.
2. **Introducción**: se describe el problema y el contexto de aplicación. (10 %)
3. **Desarrollo**: se describe el diseño e implementación del programa. (40 %)
4. **Resultados Experimentales y Discusión**: se debe presentar los resultados y hacer un análisis de los mismos. Presentar casos fáciles y difíciles. (40 %)
5. **Conclusiones** (10 %)

4. Entrega

La entrega se realizará por u-cursos hasta el 28 de noviembre de 2021 a las 23:59. Se debe adjuntar informe más código fuente.