**Documentación**

**-Utilización de las librerías:**

keras, numpy, pandas, ultralytics, tensorflow, random, os, matplotlib, sklearn, glob, cv2, shutil, IPython.display, PIL, tqdm, copy.

Para hacer uso de estas librerías se realizaron las correspondientes importaciones y descargas. Las descargas se hicieron bajo el comando: pip install <librería>

**-Carga de archivos y división de datos por carpetas**

Se crearon dos carpetas principales: images y labels. En images se encuentran los archivos .jpg, y en labels los archivos .txt, que corresponden a las coordenas de las etiquetas definidas a través de YoloLabel.

Se les asignaron porcentajes de cubrimiento de los archivos a cada carpeta (test, train, val). En este caso se asignó un 70% de los datos a la carpeta de train, 10% a la carpeta de val y 20% a la carpeta de test. Luego se aleatorizó el orden de los archivos, pero de una forma conjunta entre images y labels, esto debido a que los documentos que queden en cada carpeta deben ser coincidenciales con su contraparte, es decir, cada archivo de images debe tener su correspondiente en labels.

**-Creación de un archivo config**

Se creó un archivo config.yaml, en el que se declararon las etiquetas utilizadas para la detección de objetos. Estas etiquetas fueron: ['Vehiculos', 'Construcciones', 'Vias', 'Rios', 'Mineria'].

**-Funciones para la visualización del etiquetado de imágenes**

*Estas funciones se encuentran comentadas, para poder usarlas no hace falta más que descomentarlas y correr el archivo.*

Se definieron ciertas funciones para visualizar el etiquetado y detección de elementos de una imagen de acuerdo con las coordenadas establecidas en labels. Para probar esta funcionalidad, se especificó el path al que se desea acceder y se establece aleatoriamente un número para visualizar la imagen, se comprueba que efectivamente este archivo exista y despliega el archivo con una comparación de la imagen plana y su versión con una etiqueta.

**-Entrenamiento del modelo de detección de imágenes**

Para el entrenamiento del modelo de detección de imágenes, se utilizó la librería se ultralytics. Con la función model.train, se especificó el path del archivo config.yaml, y se determinó un número de 10 epochs, es decir, el número de veces que se van a pasar cada ejemplo de entrenamiento por la red.

Los resultados obtenidos, quedaron registrados en la carpeta runs/detect/train2. Con estos resultados, es posible evaluar el rendimiento del modelo.

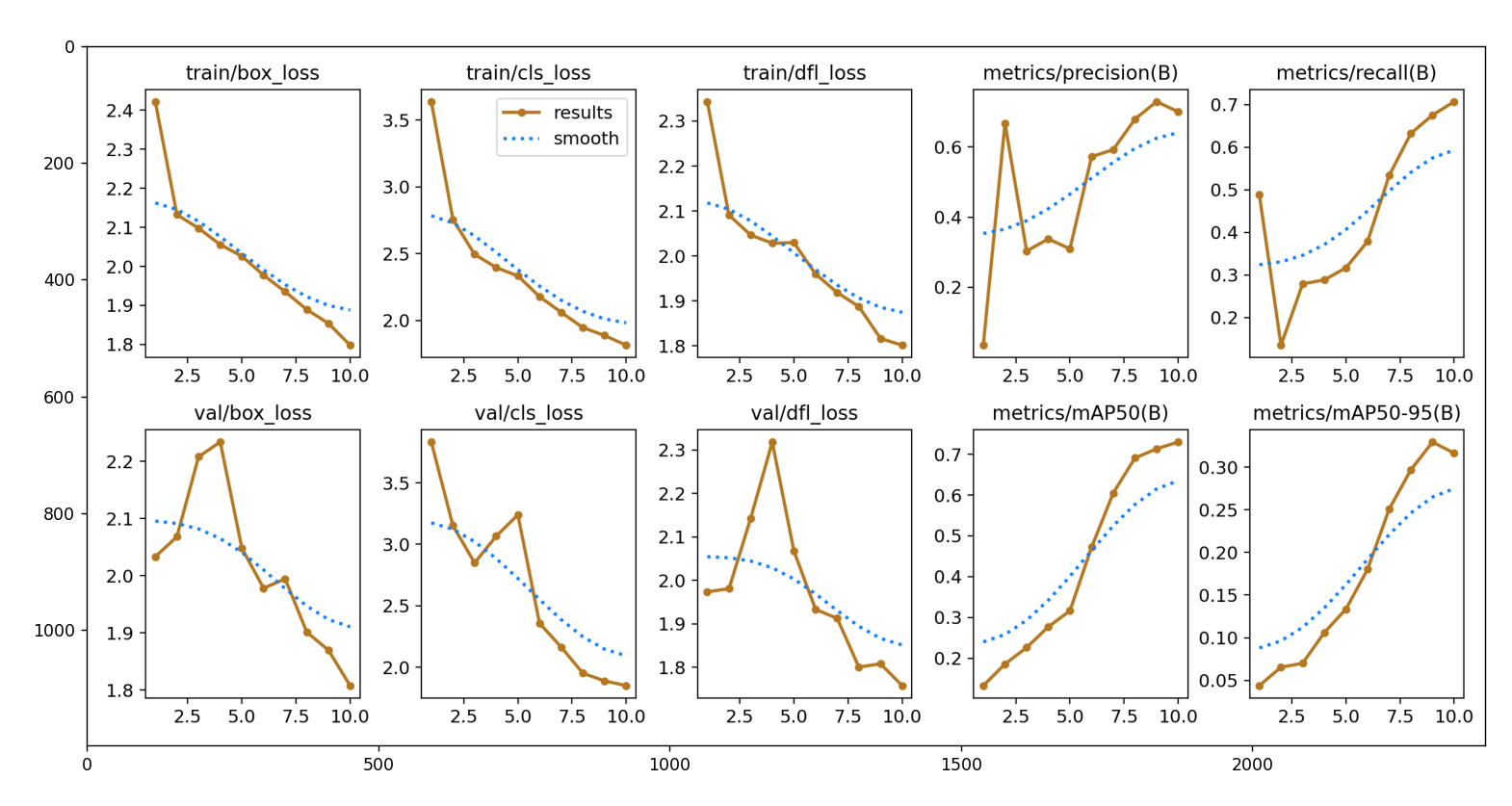
**-Evaluación de resultados**

Para evaluar y analizar los resultados del entrenamiento del modelo, se definieron las funciones de evaluación de las métricas de entrenamiento, validación y prueba. Los resultados obtenidos de estas validaciones quedaron guardados en: runs/detect/val, runs/detect/val2, runs/detect/val3, respectivamente.

**-Visualización de resultados**

Los resultados obtenidos pueden ser visualizados gráficamente con las funciones definidas al final del proyecto.

**Resultados:**

****

**Pérdida de Caja (Box Loss) de Entrenamiento y Validación**: Ambas disminuyen, lo cual es positivo y muestra que el modelo está aprendiendo a predecir la ubicación de las cajas delimitadoras de manera más precisa.

**Pérdida de Clasificación (Cls Loss) de Entrenamiento y Validación:** Esta métrica también está disminuyendo, indicando que el modelo mejora en la clasificación correcta de los objetos dentro de las cajas delimitadoras.

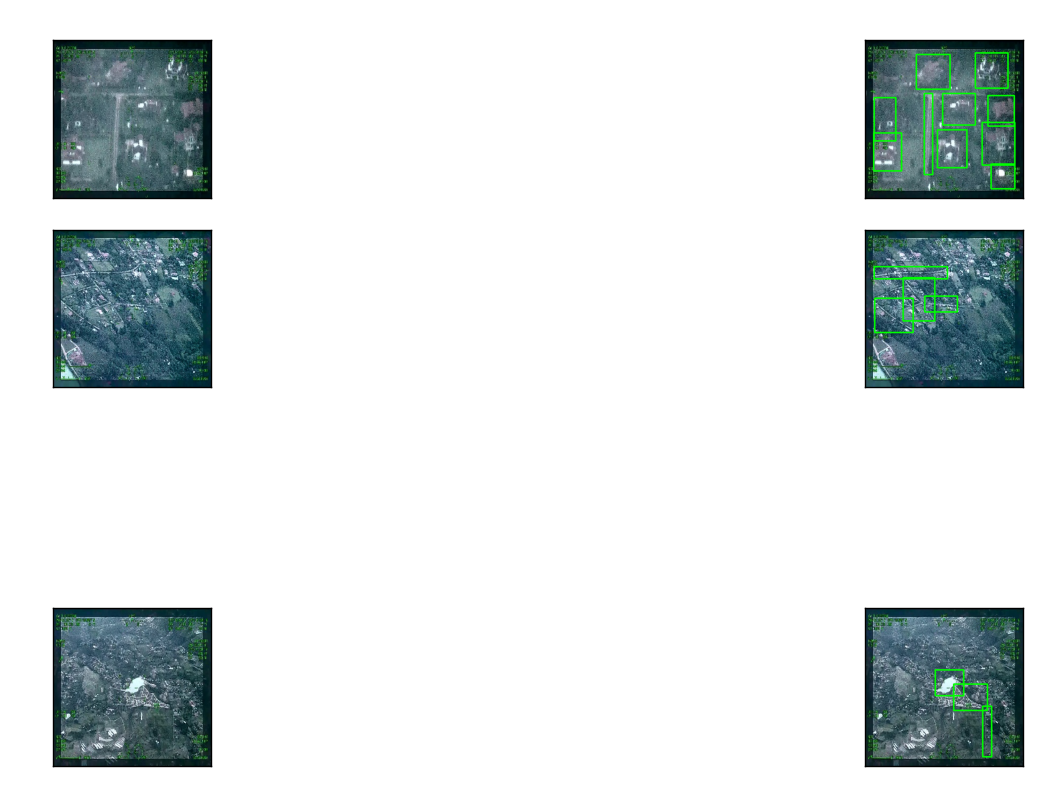
**Pérdida de DFL de Entrenamiento y Validación:** La pérdida de DFL (Loss Distribution Focal Loss) disminuye en general, señalando que el modelo está aprendiendo adecuadamente tanto la clasificación como la localización.

**Precisión (Precision):** La precisión presenta oscilaciones pero tiende a aumentar, lo cual sugiere que el modelo está mejorando en su capacidad de no etiquetar ejemplos negativos como positivos.

**Exhaustividad (Recall):** La exhaustividad aumenta, indicando que el modelo se vuelve más efectivo en identificar todas las muestras positivas.

**mAP@50:** Esta métrica de precisión promedio aumenta, lo que significa que hay más coincidencias entre las cajas delimitadoras predichas por el modelo y las cajas verdaderas.

**mAP@50-95:** También en aumento, lo que sugiere mejoras en la detección precisa de objetos a varios niveles de exigencia de solapamiento con las cajas verdaderas.

****