DS Dev assessment exercise

The Problem

We need to prepare data to be processed by a machine learning algorithm. We will need to merge images and label metadata in order to evaluate the dataset. Create a program in python that:

1. Processes a metadata file with image metadata

- Parse data from 3 different sources. These sources are the 3 different files supplied, which data format is described bellow.
    - Assume that this process could be extended to accept other sources.
- All labels should be of type *Raccoon.*
- Operations on the metadata following parsing:
    - Calculate percentage of the total area of the image occupied by labels.
    - Calculate label width-to-height ratio of all the labels and plot the distribution of the output.
- Save the output to a file called `data_output.json` where the script located. This file should be cleared when the script is re-ran.
- Output should be in normalized coordinates.

2. Processes images

- Crop image around label with size defined on config file.
    - Draw labels on image.
    - Padding to ensure image size is consistently max size.
- Optional: Additionally of cropping around the label, make a sliding window with:
    - Overlap as defined in config file in `overlap` field.
    - Window size has `max_width`/2 by `max_height`/2 from config file.
    - Labels drawn on the images.
- Save processed images as jpeg where the script is located, in a sub-folder defined in the config file's field `output_file`. This sub-folder should be cleared when the script is re-ran.

3. Logging

- Logger output should be `Timestamp Class FunctionName AnythingExtraRelevant: Log message`
- Output should be:
    - Shown on screen.
    - Saved to a file called `processing.log` where the script located. This file should not be cleared when the script is re-ran.

General Guidelines

- Due to the foreseeable amount of data, ensure code is efficient and runs in the least amount of time possible.
- Using TensorFlow 2.x would be a plus.
- Code development should be done on a Git repository with atomic commits.
- Make sure to output the time it takes to run the entire script.
- Optional: Implement unit testing on the metadata processing.

Source 1 format

```
{
  "data":
    [
      {
        "folder": "/Path/to/Images",
        "filename": "image_name.jpg",
        "xmin": 0.5,
        "ymin": 0.5,
        "xmax": 0.55,
        "ymax": 0.55,
        "image_height": 640,
        "image_width": 480
      },
      ...
    ]
}
```

Source 2 format

```
{
  "data":
    [
      {
        "path": "/Path/to/Images/image_name.png",
        "xmin": 200,
        "ymin": 200,
        "xmax": 300,
        "ymax": 300,
        "image_height": "640",
        "image_width": "480"
      },
      ...
    ]
}
```

Source 3 format

```
{
  "data":
    [
      {
        "filename": "image_name",
        "folder": "/Path/to/Images",
        "label_horz_center": 200,
        "label_vert_center": 200,
        "label_width": 300,
        "label_height": 300
      },
      ...
    ]
}
```

Config format

```
{
  "max_height": 300,
  "max_width": 300,
  "output_folder": "images",
  "output_file": "data_output.json",
  "overlap": 0.15
}
```

Metadata processing output format

```json
{
  "data": [
    {
      "path": "/Path/to/Image/",
      "filename": "image_name.jpg",
      "image_height": 640,
      "image_width": 480,
      "label_area_perc": 0.8,
      "labels": [
        {
          "xmin": 0.5,
          "ymin": 0.5,
          "xmax": 0.55,
          "ymax": 0.55,
          "label": "Raccoon",
          "width_to_height": 0.115
        },
        ...
      ]
    },
    ...
  ]
}
```

This output refers to the output of the metadata processing ***before*** the images are processed, hence it refers to the complete image and labels.