# Data preparation

## Download: Dota V1.5

Train set: <https://drive.google.com/drive/folders/1gmeE3D7R62UAtuIFOB9j2M5cUPTwtsxK?usp=sharing>

Val set: <https://drive.google.com/drive/folders/1n5w45suVOyaqY84hltJhIZdtVFD9B224?usp=sharing>

Test set: <https://drive.google.com/drive/folders/1mYOf5USMGNcJRPcvRVJVV1uHEalG5RPl?usp=sharing>

For the label files, make sure you download **v1.5**. and **\_hbb** version – which stands for horizontal bounding boxes (As opposed to non-hbb which is oriented bounding boxes)

## Prepare folder

Default folder setup is

* <project root>/raw\_data/dota\_orig/train/images
* <project root>/raw\_data/dota\_orig/train/labelTxt
* <project root>/raw\_data/dota\_orig/val/images
* <project root>/raw\_data/dota\_orig/val/labelTxt
* <project root>/raw\_data/dota\_orig/test/images

Outputs will go to <project root>/raw\_data/dota\_hazed/.

Or you can modify accordingly in the ./dotah\_generator/dotah\_v2.py (At the bottom)

## Run

To run, within folder dotah\_generator, execute python dotah\_v2.py

It will run using multi-core processing. So install the dependencies if some error pops up.

# Substitute-in the vision transformer into KJRDNet pipeline

Open file ./models/KJRDNet\_wo\_detection.py

In there, import the necessary, initialize, and load the weights (and freeze the weights). There are some template codes we already prepared.

# Update the pre-trained weights full for FFAnet and Vision encoder

Go to trainers/trainer\_kjrd\_net.py, find init\_main\_block() and updates the paths to the pre-trained weights.

# Train the full KJRDNet

To start trainer, from project root, execute

python train.py --config\_file configs/config\_rcan.yaml  
--output\_dir ./output\_models/

If there are problems with the Dataset class, go to trainers/trainer\_base.py and look for the if-else block  
elif self.dataset.lower() == 'kjrd'

Ensure the paths within the code blocks for the hazy image, clear (original image), and labels (original labels) are correct.