1. Create dataset.yaml

COCO128 is a small tutorial dataset composed of the first 128 images in COCO train2017. These same 128 images are used for both training and validation to verify our training pipeline is capable of overfitting. data/coco128.yaml, shown below, is the dataset configuration file that defines 1) an optional download command/URL for auto-downloading, 2) a path to a directory of training images (or path to a *.txt file with a list of training images), 3) the same for our validation images, 4) the number of classes, 5) a list of class names:

```
# download command/URL (optional)
download: https://github.com/ultralytics/yolov5/releases/download/v1.0/coco128.zip

# train and val data as 1) directory: path/images/, 2) file: path/images.txt, or 3) list: [path1/images/, pa
train: ../coco128/images/train2017/
val: ../coco128/images/train2017/

# number of classes
nc: 80

# class names
names: ['person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic ligh
        'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow'
        'elephant', 'bear', 'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie', 'suitcase', 'frisb
        'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboa
        'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple',
        'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'couch',
        'potted plant', 'bed', 'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote', 'keyboard',
        'cell phone', 'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scis
        'teddy bear', 'hair drier', 'toothbrush']
```

2. Create Labels

After using a tool like CVAT, makesense.ai or Labelbox to label your images, export your labels to **YOLO format**, with one `*.txt` file per image (if no objects in image, no `*.txt` file is required). The `*.txt` file specifications are:

- One row per object
- Each row is `class x_center y_center width height` format.
- Box coordinates must be in **normalized xywh** format (from 0 - 1). If your boxes are in pixels, divide `x_center` and `width` by image width, and `y_center` and `height` by image height.
- Class numbers are zero-indexed (start from 0).

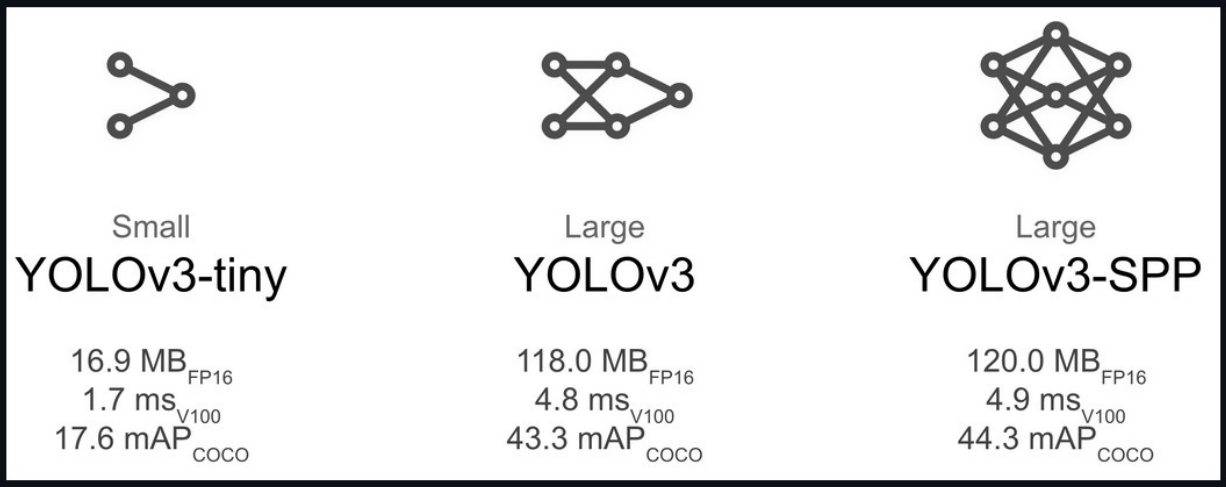
3. Organize Directories

Organize your train and val images and labels according to the example below. In this example we assume `/coco128` is **next to** the `/yolov3` directory. **YOLOv3 locates labels automatically for each image** by replacing the last instance of `/images/` in each image path with `/labels/`. For example:

```
dataset/images/im0.jpg  # image
dataset/labels/im0.txt  # label
```

4. Select a Model

Select a pretrained model to start training from. Here we select YOLOv3, the smallest and fastest model available. See our README table for a full comparison of all models.



5. Train

Train a YOLOv3 model on COCO128 by specifying dataset, batch-size, image size and either pretrained `--weights yolov3.pt` (recommended), or randomly initialized `--weights '' --cfg yolov3.yaml` (not recommended). Pretrained weights are auto-downloaded from the latest YOLOv3 release.

```
# Train YOLOv3 on COCO128 for 5 epochs
$ python train.py --img 640 --batch 16 --epochs 5 --data coco128.yaml --weights yolov3.pt
```

All training results are saved to `runs/train/` with incrementing run directories, i.e. `runs/train/exp2`, `runs/train/exp3` etc. For more details see the Training section of our Google Colab Notebook. Open in Colab Open in Kaggle