The parameter settings for the model training process

```
    NAME = None # Override in subclasses

    # NUMBER OF GPUs to use. For CPU training, use 1
    GPU_COUNT = 1

    # Number of images to train with on each GPU. A 12 GB GPU can typically
    # handle 2 images of 1024×1024 px.
    # Adjust based on your GPU memory and image sizes. Use the highest
    # number that your GPU can handle to achieve best performance.
    IMAGES_PER_GPU = 2

    # Number of training steps per epoch
    # This value does not need to match the size of the training set. Tensorboard
    # updates are saved at the end of each epoch, so setting this to a
    # smaller number involves more frequent TensorBoard updates.
    # Validation stats are calculated at each epoch end, and they
    # can be time-intensive; therefore, to avoid lengthy times when calculating validation statistics, do not set this value to a too-small number
    #.
    STEPS_PER_EPOCH = 1,000

    # Number of validation steps to run at the end of every training epoch.
    # A larger number improves the accuracy of the validation statistics, but slows
    # down training.
    VALIDATION_STPES = 50

    # The strides of each layer of the FPN Pyramid. These values
    # are based on a Resnet101 backbone.
    BACKBONE_STRIDES = [4, 8, 16, 32, 64]

    # Number of classification classes (including background)
    NUM_CLASSES = 1 # Override in subclasses

    ###### RPN ######
    # Length of square anchor side in pixels
    RPN_ANCHOR_SCALES = (32, 64, 128, 256, 512)

    # Ratios of anchors at each cell (width/height)
    # A value of 1 represents a square anchor, while 0.5 indicates a wide anchor
    RPN_ANCHOR_RATIOS = [0.5, 1, 2]

    # Anchor stride
    # If 1, then anchors are created for each cell in the backbone feature map.
```

```
# If 2, then anchors are created for every other cell, and so on.
RPN_ANCHOR_STRIDE = 2

# How many anchors per image to use for RPN training
RPN_TRAIN_ANCHORS_PER_IMAGE = 256

# ROIs kept after nonmaximum suppression (training and inference)
POST_NMS_ROIS_TRAINING = 2,000
POST_NMS_ROIS_INFERENCE = 1,000

# If enabled, resizes instance masks to a smaller size to reduce
# memory load. Recommended when using high-resolution images.
USE_MINI_MASK = True
MINI_MASK_SHAPE = (56, 56) # (height, width) of the mini-mask

# Input image resizing
# Images are resized such that the smallest side is >= IMAGE_MIN_DIM and
# the longest side is <= IMAGE_MAX_DIM. When both conditions cannot
# be satisfied simultaneously the IMAGE_MAX_DIM value takes precedence.
IMAGE_MIN_DIM = 800
IMAGE_MAX_DIM = 1,024
# If True, pad images with zeros such that they become (MAX_DIM × MAX_DIM)
IMAGE_PADDING = True    # currently, the False option is not supported

# Image mean (RGB)
MEAN_PIXEL = np.array([123.7, 116.8, 103.9])

# Number of ROIs per image to feed to classifier/mask heads
TRAIN_ROIS_PER_IMAGE = 128 # TODO: this study uses 512

# Percent of positive ROIs used to train classifier/mask heads
ROI_POSITIVE_RATIO = 0.33

# Pooled ROIs
POOL_SIZE = 7
MASK_POOL_SIZE = 14
MASK_SHAPE = [28, 28]

# Maximum number of ground truth instances to use in one image
MAX_GT_INSTANCES = 100

# Bounding box refinement standard deviation for RPN and final detections.
RPN_BBOX_STD_DEV = np.array([0.1, 0.1, 0.2, 0.2])
BBOX_STD_DEV = np.array([0.1, 0.1, 0.2, 0.2])
```

```
# Max number of final detections
DETECTION_MAX_INSTANCES = 100

# Minimum probability value to accept a detected instance
# ROIs below this threshold are skipped
DETECTION_MIN_CONFIDENCE = 0.7

# Nonmaximum suppression threshold for detection
DETECTION_NMS_THRESHOLD = 0.3

# Learning rate and momentum
# The paper uses a learning rate of 0.02, but we found that value to cause weights to explode often
LEARNING_RATE = 0.002
LEARNING_MOMENTUM = 0.9

# Weight decay regularization
WEIGHT_DECAY = 0.0001

# Use RPN ROIs or externally generated ROIs for training
# This value should be True in most situations. Set to False to train
# the head branches on ROIs generated by code rather than the ROIs from
# the RPN. For example, to debug the classifier head without having to
# train the RPN.
USE_RPN_ROIS = True
```