Export Training Data For Deep Learning (Image Analyst)

ArcGIS Pro 3.1

In this topic

1. Summary
2. Usage
3. Parameters
4. Environments
5. Licensing information

Available with Spatial Analyst license.

Available with Image Analyst license.

Summary

Converts labeled vector or raster data into deep learning training datasets using a remote sensing image. The output will be a folder of image chips and a folder of metadata files in the specified format.

Usage

- This tool will create training datasets to support third-party deep learning applications, such as Google TensorFlow, Keras, PyTorch, Microsoft CNTK, and others.
- Deep learning class training samples are based on small subimages, called image chips, that contain the feature or class of interest.
- Use your existing classification training sample data or GIS feature class data, such as a building footprint layer, to generate image chips containing the class sample from the source image. Image chips are often 256 pixel rows by 256 pixel columns, unless the training sample size is larger. Each image chip can contain one or more objects. If the Labeled Tiles parameter metadata format is used, there can be only one object per image chip.
- By specifying the Reference System parameter value, training data can be exported in map space or pixel space (raw image space) to use for deep learning model training.
- This tool supports exporting training data from a collection of images. You can add an image folder as the Input Raster value. If the Input Raster value is a mosaic dataset or an image service, you can also specify that the Processing Mode parameter process the mosaic as either one input or each raster item separately.
- The cell size and extent can be adjusted using the geoprocessing environment settings.
- This tool honors the Parallel Processing Factor environment setting. By default, Parallel Processing Factor is not enabled; consequently, the tool will run on a single core. When large datasets are used, enable Parallel Processing Factor by specifying the number of cores the tool can use to distribute the workload.
- For information about requirements for running this tool and issues you may encounter, see Deep Learning frequently asked questions.

Parameters

DialogPython

Label	Explanation	Data Type
Input Raster	The input source imagery, typically multispectral imagery. Examples of the types of input source imagery include multispectral satellite, drone, aerial, and National Agriculture Imagery Program (NAIP). The input can be a folder of images.	Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Map Server; Map Server Layer; Internet Tiled Layer; Folder
Output Folder	The folder where the output image chips and metadata will be stored. The folder can also be a folder URL that uses a cloud storage connection file (*.acs).	Folder
Input Feature Class Or Classified Raster Or Table	The training sample data in either vector or raster form. Vector inputs should follow the training sample format generated using the Training Samples Manager pane. Raster inputs should follow a classified raster format generated by the Classify Raster tool. The raster input can also be from a folder of classified rasters. Classified raster inputs require a corresponding raster attribute table. Input tables should follow a training sample format generated by the Label Objects for Deep Learning within the Training Samples Manager pane. Following the proper training sample format will produce optimal results	Feature Class; Feature Layer; Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Table; Folder

Label	Explanation	Data Type
	with the statistical information; however, the input can also be a point feature class without a class value field or an integer raster without any class information.	
Image Format	Specifies the raster format that will be used for the image chip outputs. The PNG and JPEG formats support up to three bands. - TIFF format—TIFF format will be used. - PNG format—PNG format will be used. - JPEG format—JPEG format will be used. - MRF (Meta Raster Format)—Meta Raster Format (MRF) will be used.	String
Tile Size X (Optional)	The size of the image chips for the x dimension.	Long
Tile Size Y (Optional)	The size of the image chips for the y dimension.	Long
Stride X (Optional)	The distance to move in the x direction when creating the next image chips. When stride is equal to tile size, there will be no overlap. When stride is equal to half the tile size, there will be 50 percent overlap.	Long
Stride Y (Optional)	The distance to move in the y direction when creating the next image chips. When stride is equal to tile size, there will be no overlap. When stride is equal to half the tile size, there will be 50 percent overlap.	Long
Output No Feature Tiles (Optional)	Specifies whether image chips that do not capture training samples will be exported. - Checked—All image chips, including those that do not capture training samples, will be exported. - Unchecked—Only image chips that capture training samples will be exported. This is the default. If checked, image chips that do not capture labeled data will also be exported; if not checked, they will not be exported.	Boolean
Metadata Format (Optional)	Specifies the format that will be used for the output metadata labels. If the input training sample data is a feature class layer, such as a building layer or a standard classification training sample file, use the KITTI Labels or PASCAL Visual Object Classes option (KITTI_rectangles or PASCAL_VOC_rectangles in Python). The output metadata is a .txt file or an .xml file containing the training sample data contained in the minimum bounding rectangle. The name of the metadata file matches the input source image name. If the input training sample data is a class map, use the Classified Tiles option (Classified_Tiles in Python) as the output metadata format. For the KITTI metadata format, 15 columns are created, but only 5 of them are used in the tool. The first column is the class value. The next 3 columns are skipped. Columns 5 through 8 define the minimum bounding rectangle, which is composed of four image coordinate locations: left, top, right, and bottom pixels. The minimum bounding rectangle encompasses the training chip used in the deep learning classifier. The remaining columns are not used. - KITTI Labels—The metadata will follow the same format as the Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) Object Detection Evaluation dataset. The KITTI dataset is a vision benchmark suite. The label files are plain text files. All values, both numerical and strings, are separated by spaces, and each row corresponds to one object.This format is used for object detection. - PASCAL Visual Object Classes—The metadata will follow the same format as the Pattern Analysis, Statistical Modeling and Computational Learning, Visual Object Classes (PASCAL_VOC) dataset. The PASCAL VOC dataset is a standardized image dataset for object class recognition. The label files are in XML format and contain information about image name, class value, and bounding boxes. This format is used for object detection. This is the default. - Classified Tiles—The output will be one classified image chip per input image chip. No other metadata for each image chip is used. Only the statistics output has more information about the classes, such as class names, class values, and output statistics. This format is primarily used for pixel classification. This format is also used for change detection when the output is one classified image chip from two image chips. - RCNN Masks—The output will be image chips that have a mask on the areas where the sample exists. The model generates bounding boxes and segmentation masks for each instance of an object in the image. This format is based on Feature Pyramid Network (FPN) and a ResNet101 backbone in the deep learning framework model. This format is used for object detection, however, can also be used for object tracking when the Siam Mask model type is used during training, as well as time series pixel classification when the PSETAE architecture is used. - Labeled Tiles—Each output tile will be labeled with a specific class. This format is used for object classification. - Multi-labeled Tiles—Each output tile will be labeled with one or more classes. For example, a tile may be labeled agriculture and also cloudy. This format is used for object classification. - Export Tiles—The output will be image chips with no label. This format is used for image translation techniques, such as Pix2Pix and Super Resolution. - CycleGAN—The output will be image chips with no label. This format is used for image translation technique CycleGAN, which is used to train images that do not overlap.	String

Label	Explanation	Data Type
	- Imagenet—Each output tile will be labeled with a specific class. This format is used for object classification; however, it can also be used for object tracking when the Deep Sort model type is used during training. - Panoptic Segmentation—The output will be one classified image chip and one instance per input image chip. The output will also have image chips that mask the areas where the sample exists; these image chips will be stored in a different folder.This format is used for both pixel classification and instance segmentation, so two output labels folders will be produced.	
Start Index (Optional)	**Legacy:** This parameter has been deprecated.	Long
Class Value Field (Optional)	The field that contains the class values. If no field is specified, the system searches for a value or classvalue field. If the feature does not contain a class field, the system determines that all records belong to one class.	Field
Buffer Radius (Optional)	The radius of a buffer around each training sample that will be used to delineate a training sample area. This allows you to create circular polygon training samples from points. The linear unit of the Input Feature Class Or Classified Raster Or Table parameter value's spatial reference is used.	Double
Input Mask Polygons (Optional)	A polygon feature class that delineates the area where image chips will be created. Only image chips that fall completely within the polygons will be created.	Feature Layer
Rotation Angle (Optional)	The rotation angle that will be used to generate additional image chips. An image chip will be generated with a rotation angle of 0, which means no rotation. It will then be rotated at the specified angle to create an additional image chip. The same training samples will be captured at multiple angles in multiple image chips for data augmentation. The default rotation angle is 0.	Double
Reference System (Optional)	Specifies the type of reference system that will be used to interpret the input image. The reference system specified must match the reference system used to train the deep learning model. - Map space—A map-based coordinate system will be used. This is the default. - Pixel space—Image space will be used, with no rotation and no distortion.	String
Processing Mode (Optional)	Specifies how all raster items in a mosaic dataset or an image service will be processed. This parameter is applied when the input raster is a mosaic dataset or an image service. - Process as mosaicked image—All raster items in the mosaic dataset or image service will be mosaicked together and processed. This is the default. - Process all raster items separately—All raster items in the mosaic dataset or image service will be processed as separate images.	String
Blacken Around Feature (Optional)	Specifies whether the pixels around each object or feature in each image tile will be masked out. This parameter only applies when the Metadata Format parameter is set to Labeled Tiles and an input feature class or classified raster has been specified. - Unchecked—Pixels surrounding objects or features will not be masked out. This is the default. - Checked—Pixels surrounding objects or features will be masked out.	Boolean
Crop Mode (Optional)	Specifies whether the exported tiles will be cropped so that they are all the same size. This parameter only applies when the Metadata Format parameter is set to either Labeled Tiles or Imagenet, and an input feature class or classified raster has been specified. - Fixed size—Exported tiles will be cropped to the same size and will center on the feature. This is the default. - Bounding box—Exported tiles will be cropped so that the bounding geometry surrounds only the feature in the tile.	String
Additional Input Raster (Optional)	An additional input imagery source that will be used for image translation methods. This parameter is valid when the Metadata Format parameter is set to Classified Tiles, Export Tiles, or CycleGAN.	Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Map Server; Map Server Layer;

Label	Explanation	Data Type
		Internet Tiled Layer; Folder
Instance Feature Class (Optional)	The training sample data collected that contains classes for instance segmentation. The input can also be a point feature class without a class value field or an integer raster without any class information. This parameter is only valid when the Metadata Format parameter is set to Panoptic Segmentation.	Feature Class; Feature Layer; Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Table; Folder
Instance Class Value Field (Optional)	The field that contains the class values for instance segmentation. If no field is specified, the tool will use a value or class value field, if one is present. If the feature does not contain a class field, the tool will determine that all records belong to one class. This parameter is only valid when the Metadata Format parameter is set to Panoptic Segmentation.	Field
Minimum Polygon Overlap Ratio (Optional)	The minimum overlap percentage for a feature to be included in the training data. If the percentage overlap is less than the value specified, the feature will be excluded from the training chip, and will not be added to the label file. The percent value is expressed as a decimal. For example, to specify an overlap of 20 percent, use a value of 0.2. The default value is 0, which means that all features will be included. This parameter improves the performance of the tool and also improves inferencing. The speed is improved since less training chips are created. The inferencing is improved since the model is trained to only detect large patches of objects, and ignores small corners of features. This means less false positives will be detected, and less false positives will be removed by the Non Maximum Suppression tool. This parameter is active when the Input Feature Class Or Classified Raster Or Table parameter value is a feature class.	Double

```
ExportTrainingDataForDeepLearning(in_raster, out_folder, in_class_data, image_chip_format, {tile_size_x}, {tile_size_y}, {stride_x}, {str
```

Name	Explanation	Data Type
in_raster	The input source imagery, typically multispectral imagery. Examples of the types of input source imagery include multispectral satellite, drone, aerial, and National Agriculture Imagery Program (NAIP). The input can be a folder of images.	Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Map Server; Map Server Layer; Internet Tiled Layer; Folder
out_folder	The folder where the output image chips and metadata will be stored. The folder can also be a folder URL that uses a cloud storage connection file (*.acs).	Folder
in_class_data	The training sample data in either vector or raster form. Vector inputs should follow the training sample format generated using the Training Samples Manager pane. Raster inputs should follow a classified raster format generated by the Classify Raster tool. The raster input can also be from a folder of classified rasters. Classified raster inputs require a corresponding raster attribute table. Input tables should follow a training sample format generated by the Label Objects for Deep Learning within the Training Samples Manager pane. Following the proper training sample format will produce optimal results with the statistical information; however, the input can also be a point feature class without a class value field or an integer raster without any class information.	Feature Class; Feature Layer; Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Table; Folder
image_chip_format	Specifies the raster format that will be used for the image chip outputs. The PNG and JPEG formats support up to three bands. • TIFF—TIFF format will be used. • PNG—PNG format will be used. • JPEG—JPEG format will be used. • MRF—Meta Raster Format (MRF) will be used.	String
tile_size_x (Optional)	The size of the image chips for the x dimension.	Long
tile_size_y (Optional)	The size of the image chips for the y dimension.	Long
stride_x (Optional)	The distance to move in the x direction when creating the next image chips. When stride is equal to tile size, there will be no overlap. When stride is equal to half the tile size, there will be 50 percent overlap.	Long

Name	Explanation	Data Type
stride_y (Optional)	The distance to move in the y direction when creating the next image chips. When stride is equal to tile size, there will be no overlap. When stride is equal to half the tile size, there will be 50 percent overlap.	Long
output_nofeature_tiles (Optional)	Specifies whether image chips that do not capture training samples will be exported. - ALL_TILES—All image chips, including those that do not capture training samples, will be exported. - ONLY_TILES_WITH_FEATURES—Only image chips that capture training samples will be exported. This is the default.	Boolean

metadata_format
(Optional)
	Specifies the format that will be used for the output metadata labels.
	If the input training sample data is a feature class layer, such as a building layer or a standard classification training sample file, use the KITTI Labels or PASCAL Visual Object Classes option (KITTI_rectangles or PASCAL_VOC_rectangles in Python). The output metadata is a .txt file or an .xml file containing the training sample data contained in the minimum bounding rectangle. The name of the metadata file matches the input source image name. If the input training sample data is a class map, use the Classified Tiles option (Classified_Tiles in Python) as the output metadata format.
	- KITTI_rectangles—The metadata will follow the same format as the Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) Object Detection Evaluation dataset. The KITTI dataset is a vision benchmark suite. The label files are plain text files. All values, both numerical and strings, are separated by spaces, and each row corresponds to one object.This format is used for object detection.
	- PASCAL_VOC_rectangles—The metadata will follow the same format as the Pattern Analysis, Statistical Modeling and Computational Learning, Visual Object Classes (PASCAL_VOC) dataset. The PASCAL VOC dataset is a standardized image dataset for object class recognition. The label files are in XML format and contain information about image name, class value, and bounding boxes. This format is used for object detection. This is the default.
	- Classified_Tiles—The output will be one classified image chip per input image chip. No other metadata for each image chip is used. Only the statistics output has more information about the classes, such as class names, class values, and output statistics. This format is primarily used for pixel classification. This format is also used for change detection when the output is one classified image chip from two image chips.
	- RCNN_Masks—The output will be image chips that have a mask on the areas where the sample exists. The model generates bounding boxes and segmentation masks for each instance of an object in the image. This format is based on Feature Pyramid Network (FPN) and a ResNet101 backbone in the deep learning framework model. This format is used for object detection, however, can also be used for object tracking when the Siam Mask model type is used during training, as well as time series pixel classification when the PSETAE architecture is used.
	- Labeled_Tiles—Each output tile will be labeled with a specific class. This format is used for object classification.
	- MultiLabeled_Tiles—Each output tile will be labeled with one or more classes. For example, a tile may be labeled agriculture and also cloudy. This format is used for object classification.
	- Export_Tiles—The output will be image chips with no label. This format is used for image translation techniques, such as Pix2Pix and Super Resolution.
	- CycleGAN—The output will be image chips with no label. This format is used for image translation technique CycleGAN, which is used to train images that do not overlap.
	- Imagenet—Each output tile will be labeled with a specific class. This format is used for object classification; however, it can also be used for object tracking when the Deep Sort model type is used during training.
	- Panoptic_Segmentation—The output will be one classified image chip and one instance per input image chip. The output will also have image chips that mask the areas where the sample exists; these image chips will be stored in a different folder.This format is used for both pixel classification and instance segmentation, so two output labels folders will be produced.
	For the KITTI metadata format, 15 columns are created, but only 5 of them are used in the tool. The first column is the class value. The next 3 columns are skipped. Columns 5 through 8 define the minimum bounding rectangle, which is composed of four image coordinate locations: left, top, right, and bottom pixels. The minimum bounding rectangle encompasses the training chip used in the deep learning classifier. The remaining columns are not used.
	The following is an example of the PASCAL_VOC_rectangles option:
	`  <?xml version=”1.0”?>`
	`- <layout>`
	`      <image>000000000</image>`
	`      <object>1</object>`
	`    - <part>`
	`        <class>1</class>`
	`      - <bndbox>`
	`          <xmin>31.85</xmin>`
	`          <ymin>101.52</ymin>`
	`          <xmax>256.00</xmax>`
		String

Name	Explanation	Data Type

	`        <ymax>256.00</ymax>`
	`      </bndbox>`
	`    </part>`
	`</layout>`
	For more information, see PASCAL Visual Object Classes.	

start_index (Optional)	**Legacy:** This parameter has been deprecated. Use a value of 0 or # in Python.	Long
class_value_field (Optional)	The field that contains the class values. If no field is specified, the system searches for a value or classvalue field. If the feature does not contain a class field, the system determines that all records belong to one class.	Field
buffer_radius (Optional)	The radius of a buffer around each training sample that will be used to delineate a training sample area. This allows you to create circular polygon training samples from points. The linear unit of the in_class_data parameter value's spatial reference is used.	Double
in_mask_polygons (Optional)	A polygon feature class that delineates the area where image chips will be created. Only image chips that fall completely within the polygons will be created.	Feature Layer
rotation_angle (Optional)	The rotation angle that will be used to generate additional image chips. An image chip will be generated with a rotation angle of 0, which means no rotation. It will then be rotated at the specified angle to create an additional image chip. The same training samples will be captured at multiple angles in multiple image chips for data augmentation. The default rotation angle is 0.	Double
reference_system (Optional)	Specifies the type of reference system that will be used to interpret the input image. The reference system specified must match the reference system used to train the deep learning model. • MAP_SPACE—A map-based coordinate system will be used. This is the default. • PIXEL_SPACE—Image space will be used, with no rotation and no distortion.	String
processing_mode (Optional)	Specifies how all raster items in a mosaic dataset or an image service will be processed. This parameter is applied when the input raster is a mosaic dataset or an image service. • PROCESS_AS_MOSAICKED_IMAGE—All raster items in the mosaic dataset or image service will be mosaicked together and processed. This is the default. • PROCESS_ITEMS_SEPARATELY—All raster items in the mosaic dataset or image service will be processed as separate images.	String
blacken_around_feature (Optional)	Specifies whether the pixels around each object or feature in each image tile will be masked out. This parameter only applies when the metadata_format parameter is set to Labeled_Tiles and an input feature class or classified raster has been specified. • NO_BLACKEN—Pixels surrounding objects or features will not be masked out. This is the default. • BLACKEN_AROUND_FEATURE—Pixels surrounding objects or features will be masked out.	Boolean
crop_mode (Optional)	Specifies whether the exported tiles will be cropped so that they are all the same size. This parameter only applies when the metadata_format parameter is set to either Labeled_Tiles or Imagenet, and an input feature class or classified raster has been specified. • FIXED_SIZE—Exported tiles will be cropped to the same size and will center on the feature. This is the default. • BOUNDING_BOX—Exported tiles will be cropped so that the bounding geometry surrounds only the feature in the tile.	String
in_raster2 (Optional)	An additional input imagery source that will be used for image translation methods. This parameter is valid when the metadata_format parameter is set to Classified_Tiles, Export_Tiles, or CycleGAN.	Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Map Server; Map Server

Name	Explanation	Data Type
		Layer; Internet Tiled Layer; Folder
in_instance_data (Optional)	The training sample data collected that contains classes for instance segmentation. The input can also be a point feature class without a class value field or an integer raster without any class information. This parameter is only valid when the metadata_format parameter is set to Panoptic_Segmentation.	Feature Class; Feature Layer; Raster Dataset; Raster Layer; Mosaic Layer; Image Service; Table; Folder
instance_class_value_field (Optional)	The field that contains the class values for instance segmentation. If no field is specified, the tool will use a value or class value field, if one is present. If the feature does not contain a class field, the tool will determine that all records belong to one class. This parameter is only valid when the metadata_format parameter is set to Panoptic_Segmentation.	Field
min_polygon_overlap_ratio (Optional)	The minimum overlap percentage for a feature to be included in the training data. If the percentage overlap is less than the value specified, the feature will be excluded from the training chip, and will not be added to the label file. The percent value is expressed as a decimal. For example, to specify an overlap of 20 percent, use a value of 0.2. The default value is 0, which means that all features will be included. This parameter improves the performance of the tool and also improves inferencing. The speed is improved since less training chips are created. The inferencing is improved since the model is trained to only detect large patches of objects, and ignores small corners of features. This means less false positives will be detected, and less false positives will be removed by the Non Maximum Suppression tool. This parameter is enabled when the in_class_data parameter value is a feature class.	Double

Code sample

ExportTrainingDataForDeepLearning example 1 (Python window)
This example creates training samples for deep learning.

```
# Import system modules
import arcpy
from arcpy.ia import *

# Check out the ArcGIS Image Analyst extension license
arcpy.CheckOutExtension("ImageAnalyst")

ExportTrainingDataForDeepLearning("c:/test/image.tif", "c:/test/outfolder",
    "c:/test/training.shp", "TIFF", "256", "256", "128", "128",
        "ONLY_TILES_WITH_FEATURES", "Labeled_Tiles", 0, "Classvalue",
        0, None, 0, "MAP_SPACE", "PROCESS_AS_MOSAICKED_IMAGE", "NO_BLACKEN", "FIXED_SIZE")
```

ExportTrainingDataForDeepLearning example 2 (stand-alone script)
This example creates training samples for deep learning.

```
# Import system modules and check out ArcGIS Image Analyst extension license
import arcpy
arcpy.CheckOutExtension("ImageAnalyst")
from arcpy.ia import *

# Set local variables
inRaster = "c:/test/InputRaster.tif"
out_folder = "c:/test/OutputFolder"
in_training = "c:/test/TrainingData.shp"
image_chip_format = "TIFF"
tile_size_x = "256"
tile_size_y = "256"
stride_x= "128"
stride_y= "128"
output_nofeature_tiles= "ONLY_TILES_WITH_FEATURES"
metadata_format= "Labeled_Tiles"
start_index = 0
classvalue_field = "Classvalue"
buffer_radius = 0
in_mask_polygons = "MaskPolygon"
rotation_angle = 0
reference_system = "PIXEL_SPACE"
```

```
processing_mode = "PROCESS_AS_MOSAICKED_IMAGE"
blacken_around_feature = "NO_BLACKEN"
crop_mode = “FIXED_SIZE”

# Execute
ExportTrainingDataForDeepLearning(inRaster, out_folder, in_training,
    image_chip_format,tile_size_x, tile_size_y, stride_x,
    stride_y,output_nofeature_tiles, metadata_format, start_index,
    classvalue_field, buffer_radius, in_mask_polygons, rotation_angle,
    reference_system, processing_mode, blacken_around_feature, crop_mode)
```

Environments

Cell Size, Current Workspace, Extent, Parallel Processing Factor, Scratch Workspace

Licensing information

- Basic: Requires Image Analyst or Spatial Analyst
- Standard: Requires Image Analyst or Spatial Analyst
- Advanced: Requires Image Analyst or Spatial Analyst

Related topics

- An overview of the Classification and Pattern Recognition toolset
- Find a geoprocessing tool
- An overview of the Deep Learning toolset
- Install deep learning frameworks for ArcGIS
- Train Deep Learning Model
- Deep learning models
- Classify Objects Using Deep Learning
- Classify Pixels Using Deep Learning
- Detect Objects Using Deep Learning