ESRI Fire Damage Assessment: Transcript

This demo shows how USAA used deep learning capabilities in the ArcGIS platform to perform damage assessment of homes after the devasting Woolsey fires. These fires were last year, in Nov [2018] and burned over 95,000 acres of land and destroyed over 1500 buildings.

So we're going to use ArcGIS Pro in order to detect buildings which were damaged by the fire. In order to do that, the first thing that Deep Learning needs is training data to look at and learn from. I'm going to click into the first bookmark which is West Malibu, [which is] the first drone imagery which came into the project. The GIS analyst brought in the drone imagery and we're going to start labeling the buildings. This is a manual labeling of buildings which we're going to export out to the neural network to do training on.

I'm zoomed into the West Malibu area, and I'm showing you, as I hover over the buildings -- the buildings which are labeled "damaged" and "undamaged". We did about 100 training samples over the drone imagery and then we bring up the geospatial analyst tools. You can type in "deep learning" -- there are a lot of different deep learning tools. In order to export these buildings out [as] training samples you can use this export training data button and then it will create image chips which will we use within the Jupyter notebook which ArcGIS has integrated into their software.

To open up the Jupyter notebook, you can click ArcGIS within the [Windows] start menu and then click "Jupyter notebook". We've already pre-loaded the notebook.

This is this Jupyter notebook that interacts with the ArcGIS Python API. It's simple boilerplate code that you can use to train models. We bring in the library. We prepare the data that we just exported out using this "export training samples". We indicate which fields contain the data that we're going to train on -- damaged is 1, undamaged is 2. Then we can actually visually see the image chips that were exported from ArcGIS Pro to the data scientist to use within their model. ArcGIS also has a way to visualize the learning rate which is a parameter that is used when you actually run the training.

You also specify the number of EPOCHs - in this case, we did 10 - and the amount of accuracy associated when it ran those EPOCHs.

At this point, we've run the training and it's done predictions based upon a sample of the data. We have ground-truth vs. predictions. Damaged was the ground-truth and what it predicted was damaged as well. There was about 99% accuracy after running the training multiple times.

Then you go further through the Jupyter notebook and you save out this model definition file. This is a very important aspect because this is what you can give to your GIS analyst to do inferencing within ArcGIS Pro. Now, you could go further and run inferencing within this Jupyter notebook, but typically since the GIS analyst gets the drone imagery first, they want to do it within a Windows interface GUI environment.

We can go to the next bookmark which is in Saratoga Hills, and that is the next batch of drone imagery data that came in from the field. In order to automatically detect damaged buildings with this drone imagery, we can use a tool called “detect objects using deep learning”. You can notice that there is a GPU symbol next to it which says that it can leverage the GPU instead of just doing it CPU-based.

You go ahead and click the tool. Your input raster is your new drone imagery -- Saratoga Hills. The next information you have to provide is the model definition file. And again, that model definition file is the one that you exported out in your Jupyter notebook. You click “run” and then you’ll get a new feature class within your table of contents with the results of the inferencing that happened.

We can manually go in and see this area was severely damaged -- where this [other area] is undamaged.

Right now we are running this demo on a Data Science WorkStation (DSWS) with RTX -- RTX6000. But we also have a version of this demo which can run on vGPU that used T4 for inferencing and V100 for training.