Cloud AutoML Vision Object Detection

Making batch predictions

After you have created (https://cloud.google.com/vision/automl/object-detection/docs/train) (trained) a model you can make an asynchronous prediction request for a batch of images using the **`batchPredict`** (https://cloud.google.com/automl/docs/reference/rest/v1/projects.locations.models/batchPredict) method. The `batchPredict` method applies annotations to your image based on the objects identified by your model.

Batch prediction often offers a lower cost per inference and higher throughput than synchronous (online) prediction. However, batch prediction produces a long-running operation (LRO), meaning that results are only available once the LRO has completed.

The maximum lifespan for a custom model is 18 months. You must create and train a new model to continue annotating content after that time.

Batch prediction

You can request annotations (predictions) for images by using the `batchPredict` command. The `batchPredict` command takes, as input, a CSV file stored in your Google Cloud Storage location that contains the paths to the images to annotate. Each line specifies a separate path to an image in Google Cloud Storage. For example:

`batch_prediction.csv`:

```
gs://my-cloud-storage-bucket/prediction_files/image1.jpg
gs://my-cloud-storage-bucket/prediction_files/image2.jpg
gs://my-cloud-storage-bucket/prediction_files/image3.jpg
gs://my-cloud-storage-bucket/prediction_files/image4.jpg
gs://my-cloud-storage-bucket/prediction_files/image5.jpg
gs://my-cloud-storage-bucket/prediction_files/image6.png
```

If your filenames have spaces in them use quotation marks around the Cloud Storage location. For example:

- "gs://my-cloud-storage-bucket/prediction_files/image filename with spaces.jpg"

Depending on the number of images that you specified in your CSV file, the batch predict task can take some time to complete. Even on a small number of images batch prediction will take *at minimum* 30 minutes to complete.

REST & CMD LINE C# JAVA MORE

Before using any of the request data below, make the following replacements:

- ***project-id***: your GCP project ID.
- ***location-id***: A valid location identifier. Currently you must use the following value:
  - `us-central1`
- ***model-id***: the ID of your model, from the response when you created the model. The ID is the last element of the name of your model. For example:
  - model name: `projects/`***`project-id`***`/locations/`***`location-id`***`/models/IOD4412217016962778756`
  - model id: `IOD4412217016962778756`
- ***input-storage-path***: the path to a CSV file stored on Google Cloud Storage. The requesting user must have at least read permission to the bucket.
- ***output-storage-bucket***: a Google Cloud Storage bucket/directory to save output files to, expressed in the following form: `gs://bucket/directory/`. The requesting user must have write permission to the bucket.

Field-specific considerations:

- `params.score_threshold` - A value between 0.0 and 1.0. Only results with scores greater or equal to this value will be returned.

HTTP method and URL:

```
POST https://automl.googleapis.com/v1/projects/project-id/locations/location-id/mo
```

Request JSON body:

```
{
  "inputConfig": {
    "gcsSource": {
       "inputUris": [ "input-storage-path" ]
    }
  },
  "outputConfig": {
    "gcsDestination": {
      "outputUriPrefix": "output-storage-bucket"
    }
  },
  "params": {
    "score_threshold": "0.0"
  }
}
```

To send your request, choose one of these options:

CURL POWERSHELL

Note: Ensure you have set the GOOGLE_APPLICATION_CREDENTIALS (https://cloud.google.com/docs/authentication/production) environment variable to your service account private key file path.

Save the request body in a file called `request.json`, and execute the following command:

```
curl -X POST \
-H "Authorization: Bearer "$(gcloud auth application-default print-access-token)
-H "Content-Type: application/json; charset=utf-8" \
-d @request.json \
https://automl.googleapis.com/v1/projects/project-id/locations/location-id/model
```

Response:

You should see output similar to the following:

```
{
  "name": "projects/project-id/locations/location-id/operations/IOD9266156233314795
  "metadata": {
    "@type": "type.googleapis.com/google.cloud.automl.v1.OperationMetadata",
    "createTime": "2019-06-19T21:28:35.302067Z",
    "updateTime": "2019-06-19T21:28:35.302067Z",
    "batchPredictDetails": {
      "inputConfig": {
        "gcsSource": {
          "inputUris": [
            "input-storage-path"
          ]
        }
      }
    }
  }
}
```

You can use the operation ID (`IOD926615623331479552`, in this case) to get the status of the task. For an example, see Getting the status of an operation (#get-operation).

Depending on the number of images that you specified in your CSV file, the batch predict task can take some time to complete. Even on a small number of images batch prediction will take *at minimum* 30 minutes to complete.

Once the operation has completed, the `state` shows as `DONE` and your results are written to the Google Cloud Storage file you specified:

```
{
  "name": "projects/project-id/locations/location-id/operations/IOD9266156233314795
  "metadata": {
    "@type": "type.googleapis.com/google.cloud.automl.v1.OperationMetadata",
    "createTime": "2019-06-19T21:28:35.302067Z",
    "updateTime": "2019-06-19T21:57:18.310033Z",
    "batchPredictDetails": {
      "inputConfig": {
        "gcsSource": {
          "inputUris": [
            "input-storage-path"
          ]
        }
      },
      "outputInfo": {
        "gcsOutputDirectory": "gs://storage-bucket-vcm/subdirectory/prediction-8370
      }
    }
  },
  "done": true,
  "response": {
    "@type": "type.googleapis.com/google.cloud.automl.v1.BatchPredictResult"
  }
}
```

See the Output JSONL files (#output-jsonl) section below for a sample output file.

Note: You can use your model to make batch predictions *even if* your model is not deployed.

Output JSONL files

When the batch predict task is complete, the output of the prediction is stored in the Google Cloud Storage location that you specified in your command.

In your output storage location (with your chosen object prefix) the files `image_object_detection_1.jsonl`, `image_object_detection_2.jsonl`,..., `image_object_detection_N.jsonl` will be created, where N may be 1, and depends on the total number of the successfully predicted images and annotations.

A single image will be listed only once with all its annotations, and its annotations will never be split across files.

You can specify a minimum score in the request to limit the results returned.

Each JSONL file will contain, per line, a JSON representation of a proto that wraps image's "ID" : "<id_value>" followed by a list of zero or more `AnnotationPayload` protos (called annotations), which have `imageObjectDetection` detail populated.

If prediction for any image failed (partially or completely), then an additional `errors_1.jsonl`, `errors_2.jsonl`,..., `errors_N.jsonl` files will be created (N depends on total number of failed predictions). These files will have a JSON representation of a proto that wraps the same "ID" : "<id_value>" but here followed by exactly one `google.rpc.Status` containing only `code` and `message` fields.

Example JSONL file (a single .jsonl file with 2 lines/file annotations):

`image_object_detection_0.jsonl`

Line 1 (`salad4.jpg` annotations JSON)

```
{
  "ID": "gs://my-storage-bucket/salad4.jpg",
  "annotations": [
    {
      "annotation_spec_id": "6133348538618216448",
      "display_name": "Tomato",
      "image_object_detection": {
        "bounding_box": {
          "normalized_vertices": [
            {
```

```
            "x": 0.51460195,
            "y": 0.60645401
          },
          {
            "x": 0.64264798,
            "y": 0.71467251
          }
        ],
        "vertices": []
      },
      "score": 0.87153709
    }
  },
  {
    "annotation_spec_id": "3827505529404522496",
    "display_name": "Salad",
    "image_object_detection": {
      "bounding_box": {
        "normalized_vertices": [
          {
            "y": 0.17702378
          },
          {
            "x": 1,
            "y": 0.88745028
          }
        ],
        "vertices": []
      },
      "score": 0.8579272
    }
  },
  ...
  {
    "annotation_spec_id": "2674584024797675520",
    "display_name": "Baked goods",
    "image_object_detection": {
      "bounding_box": {
        "normalized_vertices": [
          {
            "x": 0.69695503,
            "y": 0.17798239
          },
          {
            "x": 0.82379168,
            "y": 0.37175834
          }
        ],
        "vertices": []
      },
      "score": 0.0031145806
    }
  }
  ]
}
```

Line 2 (`salad5.jpg` annotations JSON)

```
{
  "ID": "gs://my-storage-bucket/salad5.jpg",
  "annotations": [
    {
      "annotation_spec_id": "3827505529404522496",
      "display_name": "Salad",
      "image_object_detection": {
        "bounding_box": {
          "normalized_vertices": [
            {
              "x": 0.0044940538,
              "y": 0.070641488
            },
            {
              "x": 0.99745369,
              "y": 0.87991059
            }
          ],
          "vertices": []
        },
        "score": 0.91780394
      }
    },
    {
      "annotation_spec_id": "6133348538618216448",
      "display_name": "Tomato",
      "image_object_detection": {
        "bounding_box": {
          "normalized_vertices": [
            {
              "x": 0.71612686,
              "y": 0.42916849
            },
            {
              "x": 0.90833008,
              "y": 0.55670345
            }
          ],
          "vertices": []
        },
        "score": 0.31472182
      }
    },
    ...
    {
      "annotation_spec_id": "7286270043225063424",
      "display_name": "Seafood",
      "image_object_detection": {
        "bounding_box": {
          "normalized_vertices": [
            {
              "x": 0.72394878,
              "y": 0.42934245
            },
            {
              "x": 0.91764188,
              "y": 0.56444579
            }
          ],
          "vertices": []
        },
        "score": 0.00031806546
      }
    }
  ]
}
```

Getting the status of an operation

REST & CMD LINE   C#   GO   MORE

Before using any of the request data below, make the following replacements:

- ***project-id***: your GCP project ID.
- ***operation-id***: the ID of your operation. The ID is the last element of the name of your operation. For example:
    - operation name: `projects/project-id/locations/location-id/operations/IOD5281059901324392598`
    - operation id: `IOD5281059901324392598`

HTTP method and URL:

```
GET https://automl.googleapis.com/v1/projects/project-id/locations/us-central1/ope
```

To send your request, choose one of these options:

CURL POWERSHELL

Note: Ensure you have set the GOOGLE_APPLICATION_CREDENTIALS (https://cloud.google.com/docs/authentication/production) environment variable to your service account private key file path.

Execute the following command:

```
curl -X GET \
-H "Authorization: Bearer "$(gcloud auth application-default print-access-token)
https://automl.googleapis.com/v1/projects/project-id/locations/us-central1/opera
```

You should see output similar to the following for a completed **import operation**:

```
{
  "name": "projects/project-id/locations/us-central1/operations/operation-id",
  "metadata": {
    "@type": "type.googleapis.com/google.cloud.automl.v1.OperationMetadata",
    "createTime": "2018-10-29T15:56:29.176485Z",
    "updateTime": "2018-10-29T16:10:41.326614Z",
    "importDataDetails": {}
  },
  "done": true,
  "response": {
    "@type": "type.googleapis.com/google.protobuf.Empty"
  }
}
```

You should see output similar to the following for a completed **create model operation**:

```
{
  "name": "projects/project-id/locations/us-central1/operations/operation-id",
  "metadata": {
    "@type": "type.googleapis.com/google.cloud.automl.v1.OperationMetadata",
    "createTime": "2019-07-22T18:35:06.881193Z",
    "updateTime": "2019-07-22T19:58:44.972235Z",
    "createModelDetails": {}
  },
  "done": true,
  "response": {
    "@type": "type.googleapis.com/google.cloud.automl.v1.Model",
    "name": "projects/project-id/locations/us-central1/models/model-id"
  }
}
```