Cloud Vision API Product Search

Formatting a bulk import CSV

You can use the `import` (https://cloud.google.com/vision/product-search/docs/reference/rest/v1/projects.locations.productSets/import) method to create a product set and products with reference images all at the same time using a CSV file. This page describes how to format the CSV file.

Creating your reference images

Reference images are images containing various views of your products. The following recommendations apply:

- Make sure the size of the file doesn't exceed the maximum size (20MB).
- Consider viewpoints that logically highlight the product and contain relevant visual information.
- Create reference images that supplement any missing viewpoints. For example, if you only have images of the right shoe in a pair, provide mirrored versions of those files as the left shoe.
- Upload the highest resolution image available.
- Show the product against a white background.
- Convert PNGs with transparent backgrounds to a solid background.

Images must be stored in a Google Cloud Storage (https://cloud.google.com/storage/) bucket. If you're authenticating your image create call with an API key, the bucket must be public. If you're authenticating with a service account, that service account must have read access on the bucket.

CSV formatting guidelines

To use the `import` method, both the CSV file *and* the images it points to must be in a Google Cloud Storage bucket. CSV files are limited to a maximum of 20000 lines. To import more images, split them into multiple CSV files.

The CSV file must contain one image per line and contain the following columns:

1. `image-uri`: The Google Cloud Storage URI of the reference image.
2. `image-id`: ***Optional***. A unique value if you supply it. Otherwise, the system will assign a unique value.
3. `product-set-id`: A unique identifier for the product set to import the images into.
4. `product-id`: A user-defined ID for the product identified by the reference image. A `product-id` can be associated with multiple reference images.

★ **Note:** A single product may also belong to several product sets. If a `product-id` already exists on bulk import then `product-category`, `product-display`, and `labels` are ignored for that line entry.

5. `product-category`: Allowed values are `homegoods-v2`, `apparel-v2`, `toys-v2`, `packagedgoods-v1`, and `general-v1` *; the category for the product identified by the reference image. Inferred by the system if not specified in the create request. Allowed values are also listed in the `productCategory` (https://cloud.google.com/vision/product-search/docs/reference/rest/v1/projects.locations.products#Product.FIELDS.product_category) reference documentation.

*** Legacy `productCategory` codes:** Legacy categories (`homegoods`, `apparel`, and `toys`) are still supported, but the updated `-v2` categories should be used for new products.

6. `product-display-name`: ***Optional***. If you don't provide a name for the product `displayName` (https://cloud.google.com/vision/product-search/docs/reference/rest/v1/projects.locations.products#Product.FIELDS.display_name) will be set to " ". You can update (https://cloud.google.com/vision/product-search/docs/reference/rest/v1/projects.locations.products/patch) this value later.
7. `labels`: ***Optional***. A string (with quotation marks) of key-value pairs that describe the products in the reference image. For example:
    - `"category=shoes"`
    - `"color=black,style=formal"`

    Vision Product Search also allows you to provide multiple values for a single key. For example:

- `"category=shoes,category=heels"`
- `"color=black,style=formal,style=mens"`

8. `bounding-poly`: ***Optional***. Specifies the area of interest in the reference image. If a bounding box is **not** specified:
   a. Bounding boxes for the image are inferred by the Vision API; multiple regions in a single image may be indexed if multiple products are detected by the API.
   b. The line must end with a comma.

   See the example below for a product without a bounding poly specified.

   If you include a bounding box, the `boundingPoly` column should contain an even number of comma-separated numbers, with the format `p1_x,p1_y,p2_x,p2_y,...,pn_x,pn_y`. An example line looks like this: `0.1,0.1,0.9,0.1,0.9,0.9,0.1,0.9`.

   To define a bounding box with the actual pixel values of your image use non-negative integers. Thus, you could express bounding boxes in 1000 pixel by 1000 pixel images in the following way:

```
gs://example-reference-images/10001-001/10001-001_A.jpg,img001,sample-set-summe
gs://example-reference-images/10001-001/10001-001_A.jpg,img001,sample-set-summe
gs://example-reference-images/10002-002/10002-002_B.jpg,img002,sample-set-summe
```

★ **Note:** You can use Vision API's Object Localizer (https://cloud.google.com/vision/docs/detecting-objects) feature to get bounding box coordinates for generalized objects in images.

Vision Product Search also allows you to use normalized values for bounding boxes. Define a bounding box using normalized values with float values in [0, 1].

Using normalized values, the above reference image rows could also be expressed as:

```
gs://example-reference-images/10001-001/10001-001_A.jpg,img001,sample-set-summe
gs://example-reference-images/10001-001/10001-001_A.jpg,img001,sample-set-summe
gs://example-reference-images/10002-002/10002-002_B.jpg,img002,sample-set-summe
```

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/)*, and code samples are licensed under the Apache 2.0 License* (https://www.apache.org/licenses/LICENSE-2.0). *For details, see our* *Site Policies* (https://developers.google.com/terms/site-policies). *Java is a registered trademark of Oracle and/or its affiliates.*

Last updated January 2, 2020.