Azure AI Vision is a branch of artificial intelligence (AI) in which software interprets visual input, often from images or video feeds.

The Azure AI Vision service is designed to help you extract information from images. It provides functionality that you can use for:

* Description and tag generation - determining an appropriate caption for an image, and identifying relevant "tags" that can be used as keywords to indicate its subject.
* Object detection - detecting the presence and location of specific objects within the image.
* People detection - detecting the presence, location, and features of people in the image.
* Image metadata, color, and type analysis - determining the format and size of an image, its dominant color palette, and whether it contains clip art.
* Category identification - identifying an appropriate categorization for the image, and if it contains any known landmarks.
* Background removal - detecting the background in an image and output the image with the background transparent or a greyscale alpha matte image.
* Moderation rating - determine if the image includes any adult or violent content.
* Optical character recognition - reading text in the image.
* Smart thumbnail generation - identifying the main region of interest in the image to create a smaller "thumbnail" version.

**Analyzing an Image with Azure AI Services**

**Overview**

* **Method**: Use the Analyze Image REST method or equivalent SDK method.
* **Output**: Returns a JSON document with requested visual features.

**Python Example**

* **SDK Usage**:

python

Copy code

from azure.ai.vision.imageanalysis import ImageAnalysisClient

from azure.ai.vision.imageanalysis.models import VisualFeatures

from azure.core.credentials import AzureKeyCredential

client = ImageAnalysisClient(

endpoint=os.environ["ENDPOINT"],

credential=AzureKeyCredential(os.environ["KEY"])

)

result = client.analyze(

image\_url="<url>",

visual\_features=[VisualFeatures.CAPTION, VisualFeatures.READ],

gender\_neutral\_caption=True,

language="en",

)

**Available Visual Features**

* **VisualFeatures.TAGS**: Identifies tags about the image, including objects, scenery, setting, and actions.
* **VisualFeatures.OBJECTS**: Returns the bounding box for each detected object.
* **VisualFeatures.CAPTION**: Generates a caption of the image in natural language.
* **VisualFeatures.DENSE\_CAPTIONS**: Generates more detailed captions for the objects detected.
* **VisualFeatures.PEOPLE**: Returns the bounding box for detected people.
* **VisualFeatures.SMART\_CROPS**: Returns the bounding box of the specified aspect ratio for the area of interest.
* **VisualFeatures.READ**: Extracts readable text.

**JSON Response Example**

* **Example JSON Response**:

json

Copy code

{

"apim-request-id": "abcde-1234-5678-9012-f1g2h3i4j5k6",

"modelVersion": "<version>",

"denseCaptionsResult": {

"values": [

{

"text": "a house in the woods",

"confidence": 0.7055229544639587,

"boundingBox": {

"x": 0,

"y": 0,

"w": 640,

"h": 640

}

},

{

"text": "a trailer with a door and windows",

"confidence": 0.6675070524215698,

"boundingBox": {

"x": 214,

"y": 434,

"w": 154,

"h": 108

}

}

]

},

"metadata": {

"width": 640,

"height": 640

}

}

**Important Notes**

* **Bounding Box and Confidence Score**: Responses contain bounding boxes (for locations) or confidence scores (for tags or captions).
* **Client Configuration**: Must specify the visual features you want analyzed to determine the information contained in the response.

**Generating Thumbnails and Removing Backgrounds with Azure AI Vision**

**Generating Smart-Cropped Thumbnails**

* **Purpose**: Thumbnails provide smaller versions of images, useful for previews on websites and applications.
* **Use Case**: Displaying tourist attractions with representative thumbnail images.

**Features of Smart-Cropped Thumbnails**

* **Region of Interest**: Uses image analysis to focus on the main subject of the image.
* **Aspect Ratio**: Specify the aspect ratio of the cropped image, ranging from 0.75 to 1.80.
* **Example**: A large building cropped to show the region of interest.

**Removing Image Background**

* **Purpose**: Separate the foreground subject from the background.
* **Use Case**: Useful for further image processing, such as adding the subject to different backgrounds.

**Features of Background Removal**

* **Alpha Matte Creation**: Splits the image into foreground and background by creating an alpha matte.
* **Alpha Matte Definition**: Foreground in all white, background in black.
* **Examples**:
  + Original Image: A skateboarder performing a trick.
  + Background Removed: Just the skateboarder on a transparent background.
  + Alpha Matte: A silhouette of the skateboarder in white with a black background.

Custom models in Azure AI Vision allow you to train an AI model to classify images or detect objects in images.

**Custom Azure AI Vision Model Types**

**1. Image Classification**

* **Functionality**: Predicts a label for an entire image based on its contents.
* **Use Case**: Identifies the main subject of the image.
* **Types of Classification**:
  + **Multi-class Classification**: Each image belongs to only one class.
  + **Multi-label Classification**: An image can have multiple labels.
* **Example**:
  + **Images**: Apple, Banana, Orange.
  + **Labels**: The type of fruit in each image.

**2. Object Detection**

* **Functionality**: Detects the presence and location of objects within an image.
* **Use Case**: Identifies types and locations of items.
* **Components**:
  + **Class Label**: Identifies the type of each object.
  + **Location**: Coordinates of a bounding box around each object.
* **Example**:
  + **Use Case**: AI-enabled checkout system in a grocery store.
  + **Objects Detected**: Type and location of fruits (e.g., one apple and two oranges).

**3. Product Recognition**

* **Functionality**: Similar to object detection but with enhanced accuracy for product labels and brand names.
* **Use Case**: Identifies products in an image with their labels and locations.
* **Components**:
  + **Class Label**: Identifies the product type.
  + **Location**: Coordinates of a bounding box around each product.
* **Example**: Detecting specific brands and products in a retail setting.

**Create a Custom Azure AI Vision Model**

1. **Azure AI Services Resource**:
   * Ensure you have an Azure AI Services resource (or an Azure AI Vision resource) deployed in your subscription.
2. **Dataset Creation**:
   * **Blob Storage Container**:
     + Create a blob storage container and upload the training images.
   * **Define Dataset**:
     + Create the dataset and connect it to your blob storage container.
     + Define the project type: image classification, object detection, or product recognition.
   * **Labeling Data**:
     + Use Azure Machine Learning Data Labeling Project to label the images, generating the COCO file.
     + Connect the COCO file to your dataset.
3. **Model Training**:
   * Specify the model type, dataset, and training budget (time).
   * Train the custom model on the dataset and labels.
   * Verify and iterate the model's performance until satisfactory results are achieved.
4. **Using the Model**:
   * Once performance is satisfactory, the model can be used in Vision Studio or integrated into your application.

**Components of a Custom Vision Project**

* **Dataset**:
  + Collection of images used for training.
  + Stored in an Azure blob storage container.
  + Accompanied by a COCO file that defines label information.
* **COCO File**:
  + A JSON file that includes:
    - **Images**: Image location, name, dimensions, and ID.
    - **Annotations**: Classifications or objects, category, area, and bounding box (for object detection).
    - **Categories**: ID and name of each label class.

**Example of a COCO File**{

"images": [

{

"id": 1,

"width": 1024,

"height": 768,

"file\_name": "abc.jpg",

"coco\_url": "AmlDatastore://fruit/abc.jpg",

"absolute\_url": "https://myBlobStorage.blob.core.windows.net/fruit/abc.jpg",

"date\_captured": "<date>"

},

{

"id": 2,

"width": 1024,

"height": 768,

"file\_name": "xyz.jpg",

"coco\_url": "AmlDatastore://fruit/xyz.jpg",

"absolute\_url": "https://myBlobStorage.blob.core.windows.net/fruit/xyz.jpg",

"date\_captured": "<date>"

}

],

"annotations": [

{

"id": 1,

"category\_id": 1,

"image\_id": 1,

"area": 0.0

},

{

"id": 2,

"category\_id": 1,

"image\_id": 2,

"area": 0.0

}

],

"categories": [

{

"id": 1,

"name": "apple"

},

{

"id": 2,

"name": "orange"

},

{

"id": 3,

"name": "banana"

}

]

}

**Bounding Box for Object Detection**

"bbox": [

0.11803319477782331,

0.41586723392402375,

0.7765206955096307,

0.3483334397217212

]

**Labeling Your Training Images**

* **Upload Images**: Ensure your images are uploaded to Azure Blob Storage and your dataset is created.
* **Azure Machine Learning Studio**: Use the Data Labeling Project in Azure Machine Learning Studio for labeling.
* **Create/Connect Data Labeling Project**: In Vision Studio, create a new labeling project or connect to an existing one.
* **Add Categories**: Define categories for images or objects (e.g., apple, orange, banana).
* **Label Images**: Label 3-5 images per category accurately.
* **ML-Assisted Labeling**: Utilize ML-assisted labeling for efficiency, but review for accuracy.
* **COCO File**: The COCO file is generated after labeling and should be added to your dataset from the Azure Machine Learning workspace.

**Training Your Model**

* **Select Model Type**: Choose the appropriate model type (image classification, object detection, or product recognition).
* **Specify Dataset**: Indicate the dataset to be used for training.
* **Training Budget**: Set the maximum time for training (training might take less time than specified).
* **Train Model**: Start the training process.

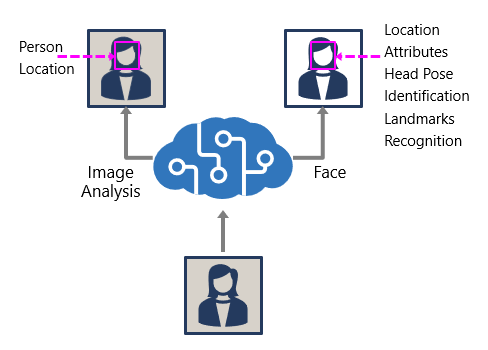
**Evaluating Your Model**

* **View Performance**: Post-training, view the model's performance on the evaluation run.
* **Default Evaluation Run**: If no separate evaluation dataset is provided, a default evaluation run is used.
* **Trigger New Evaluation Runs**: You can initiate new evaluation runs on different image sets.
* **Test in Vision Studio**: Use Vision Studio to try out the trained model on various images.

The ability to detect when a person is present, identify a person's facial location, or recognize an individual based on their facial features is a key way in which AI systems can exhibit human-like behavior and build empathy with users.

Options for face detection analysis and identification

There are two Azure AI services that you can use to build solutions that detect faces or people in images.



The Azure AI Vision and Face services offer facial analysis capabilities

**The Azure AI Vision service:**

The Azure AI Vision service enables you to detect people in an image, as well as returning a bounding box for its location.

**The Face service:**

The Face service offers more comprehensive facial analysis capabilities than the Azure AI Vision service, including:

* Face detection (with bounding box).
* Comprehensive facial feature analysis (including head pose, presence of spectacles, blur, facial landmarks, occlusion and others).
* Face comparison and verification.
* Facial recognition.

While all applications of artificial intelligence require considerations for responsible and ethical use, system that rely on facial data can be particularly problematic.

When building a solution that uses facial data, considerations include (but aren't limited to):

* Data privacy and security. Facial data is personally identifiable, and should be considered sensitive and private. You should ensure that you have implemented adequate protection for facial data used for model training and inferencing.
* Transparency. Ensure that users are informed about how their facial data is used, and who will have access to it.
* Fairness and inclusiveness. Ensure that your face-based system can't be used in a manner that is prejudicial to individuals based on their appearance, or to unfairly target individuals.

**Detect faces with the Azure AI Vision service**

To detect and analyze faces with the Azure AI Vision service, call the Analyze Image function (SDK or equivalent REST method), specifying People as one of the visual features to be returned.

In images that contain one or more people, the response includes details of their location in the image and the attributes of the detected person, like this:

{

"modelVersion": "2023-10-01",

"metadata": {

"width": 400,

"height": 600

},

"peopleResult": {

"values": [

{

"boundingBox": {

"x": 0,

"y": 56,

"w": 101,

"h": 189

},

"confidence": 0.9474349617958069

},

{

"boundingBox": {

"x": 402,

"y": 96,

"w": 124,

"h": 156

},

"confidence": 0.9310565276194865

},

...

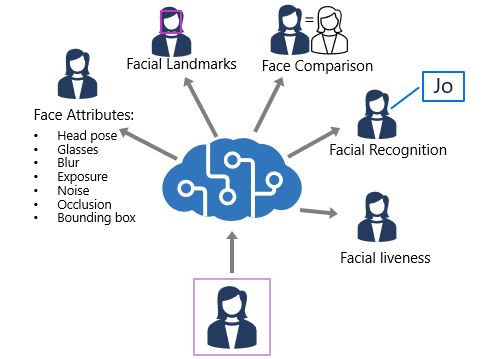
]

}

}

**Understand capabilities of the face service**

The **Face** service provides comprehensive facial detection, analysis, and recognition capabilities.



The Face service provides functionality that you can use for:

* *Face detection* - for each detected face, the results include an ID that identifies the face and the bounding box coordinates indicating its location in the image.
* *Face attribute analysis* - you can return a wide range of facial attributes, including:
  + Head pose (*pitch*, *roll*, and *yaw* orientation in 3D space)
  + Glasses (*NoGlasses*, *ReadingGlasses*, *Sunglasses*, or *Swimming Goggles*)
  + Blur (*low*, *medium*, or *high*)
  + Exposure (*underExposure*, *goodExposure*, or *overExposure*)
  + Noise (visual noise in the image)
  + Occlusion (objects obscuring the face)
  + Accessories (glasses, headwear, mask)
  + QualityForRecognition (*low*, *medium*, or *high*)
* *Facial landmark location* - coordinates for key landmarks in relation to facial features (for example, eye corners, pupils, tip of nose, and so on)
* *Face comparison* - you can compare faces across multiple images for similarity (to find individuals with similar facial features) and verification (to determine that a face in one image is the same person as a face in another image)
* *Facial recognition* - you can train a model with a collection of faces belonging to specific individuals, and use the model to identify those people in new images.
* *Facial liveness* - liveness can be used to determine if the input video is a real stream or a fake to prevent bad intentioned individuals from spoofing the recognition system.

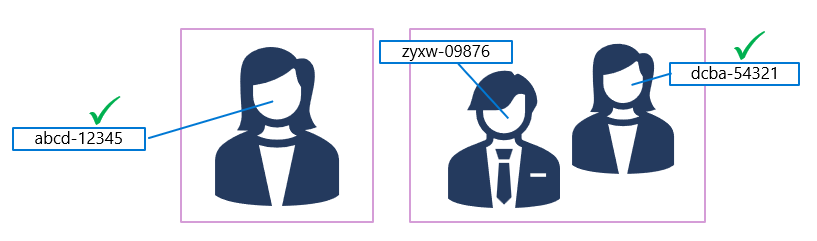
You can provision **Face** as a single-service resource, or you can use the Face API in a multi-service **Azure AI Services** resource.

If you want to use the identification, recognition, and verification features of **Face**, you'll need to apply for the [Limited Access policy](https://aka.ms/cog-services-limited-access) and get approval before these features are available.

**Compare and match detected faces**

When a face is detected by the Face service, a unique ID is assigned to it and retained in the service resource for 24 hours. The ID is a GUID, with no indication of the individual's identity other than their facial features..

While the detected face ID is cached, subsequent images can be used to compare the new faces to the cached identity and determine if they are *similar* (in other words, they share similar facial features) or to *verify* that the same person appears in two images.



This ability to compare faces anonymously can be useful in systems where it's important to confirm that the same person is present on two occasions, without the need to know the actual identity of the person.

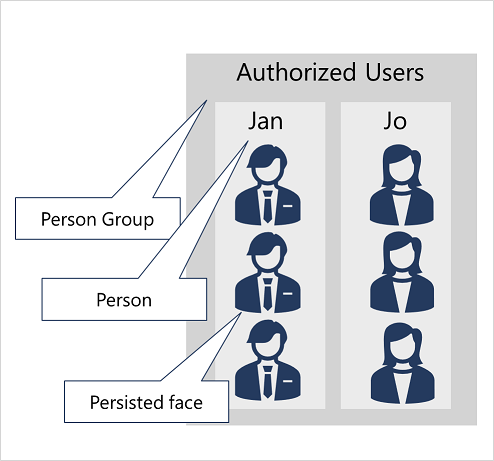
For example, by taking images of people as they enter and leave a secured space to verify that everyone who entered leaves.

**Implement facial recognition**

For scenarios where you need to positively identify individuals, you can train a facial recognition model using face images.

To train a facial recognition model with the Face service:

1. Create a **Person Group** that defines the set of individuals you want to identify (for example, *employees*).
2. Add a **Person** to the **Person Group** for each individual you want to identify.
3. Add detected faces from multiple images to each **person**, preferably in various poses. The IDs of these faces will no longer expire after 24 hours (so they're now referred to as *persisted* faces).
4. Train the model.



The trained model is stored in your Face (or Azure AI Services) resource, and can be used by client applications to:

* *Identify* individuals in images.
* *Verify* the identity of a detected face.
* Analyze new images to find faces that are *similar* to a known, persisted face.

Optical Character Recognition (OCR). OCR allows you to extract text from images, such as photos of street signs and products, as well as from documents — such as handwritten or unstructured documents.

Azure AI provides two different features that read text from documents and images, one in the Azure AI Vision Service, the other in Azure AI Document Intelligence. There is overlap in what each service provides, however each is optimized for results depending on what the input is.

* **Image Analysis** Optical character recognition (OCR):
  + Use this feature for general, unstructured documents with smaller amount of text, or images that contain text.
  + Results are returned immediately (synchronous) from a single API call.
  + Has functionality for analyzing images past extracting text, including object detection, describing or categorizing an image, generating smart-cropped thumbnails and more.
  + Examples include: street signs, handwritten notes, and store signs.
* **Document Intelligence**:
  + Use this service to read small to large volumes of text from images and PDF documents.
  + This service uses context and structure of the document to improve accuracy.
  + The initial function call returns an asynchronous operation ID, which must be used in a subsequent call to retrieve the results.
  + Examples include: receipts, articles, and invoices.

You can access both technologies via the REST API or a client library.

To use the Read OCR feature, call the ImageAnalysis function (REST API or equivalent SDK method), passing the image URL or binary data, and optionally specifying a gender neutral caption or the language the text is written in (with a default value of en for English).

To make an OCR request to ImageAnalysis, specify the visual feature as READ.

result = client.analyze(

image\_url=<image\_to\_analyze>,

visual\_features=[VisualFeatures.READ]

)

If using the REST API, specify the feature as read.

https://<endpoint>/computervision/imageanalysis:analyze?features=read&...

The results of the Read OCR function are returned synchronously, either as JSON or the language specific object of a similar structure

{

"metadata":

{

"width": 500,

"height": 430

},

"readResult":

{

"blocks":

[

{

"lines":

[

{

"text": "Hello World!",

"boundingPolygon":

[

{"x":251,"y":265},

{"x":673,"y":260},

{"x":674,"y":308},

{"x":252,"y":318}

],

"words":

[

{

"text":"Hello",

"boundingPolygon":

[

{"x":252,"y":267},

{"x":307,"y":265},

{"x":307,"y":318},

{"x":253,"y":318}

],

"confidence":0.996

},

{

"text":"World!",

"boundingPolygon":

[

{"x":318,"y":264},

{"x":386,"y":263},

{"x":387,"y":316},

{"x":319,"y":318}

],

"confidence":0.99

}

]

},

]

}

]

}

}

The Azure Video Indexer service is designed to help you extract information from videos. It provides functionality that you can use for:

* Facial recognition - detecting the presence of individual people in the image. This requires Limited Access approval.
* Optical character recognition - reading text in the video.
* Speech transcription - creating a text transcript of spoken dialog in the video.
* Topics - identification of key topics discussed in the video.
* Sentiment - analysis of how positive or negative segments within the video are.
* Labels - label tags that identify key objects or themes throughout the video.
* Content moderation - detection of adult or violent themes in the video.
* Scene segmentation - a breakdown of the video into its constituent scenes.

The Video Analyzer service provides a portal website that you can use to upload, view, and analyze videos interactively

**Integrating Azure Video Indexer into Custom Applications**

**Integration Methods**

1. **Azure Video Indexer Widgets**:
   * **Embed Widgets**: Use Azure Video Indexer widgets to embed video play, analysis, and editing functionalities into custom HTML interfaces.
   * **Controlled Access**: Share insights from specific videos without granting full access to the Azure Video Indexer portal.
2. **Azure Video Indexer API**:
   * **Access Token**: Obtain an access token using the Azure Video Indexer REST API.
   * **Automation**: Automate tasks such as video indexing, project creation, retrieving insights, and managing custom models via API calls.

**Deployment**

* **ARM Templates**:
  + **Azure Resource Manager**: Use ARM templates to deploy the Azure AI Video Indexer resource in your subscription.
  + **Parameters**: Define the necessary parameters in the template file to automate resource creation.