Virtual Assistant for Image Classification (VAfIC)

Welcome to the Virtual Assistant for Image Classification (VAfIC) project! This documentation provides an overview of the project structure and details on how to use it.

How to use?

**Before getting started, please review our API documentation on GitHub:**

<https://github.com/VitaliiYorza/WAdKO/tree/main/wadko-api>

<https://github.com/VitaliiYorza/WAdKO/tree/main/wadko-frontend>

<https://github.com/VitaliiYorza/WAdKO/tree/main/wadko-ml-models>

Ensure you have the necessary dependencies installed as mentioned in the respective repositories.

In our project there are 3 main files.

1. image\_scan\_model.py
2. predict\_image.py
3. app.py

These three files collectively form the backbone of our image processing project, offering a comprehensive solution for anyone seeking efficient image scanning, prediction, and classification.

Their cohesion and functionality are pivotal to the success of the Virtual Assistant for Image Classification (VAfIC) project. Below you can find a short description about them

1. image\_scan\_model.py
2. Description:
   1. Definition of the ImageScanModel class for image scanning and prediction.
3. Class: ImageScanModel
   1. Represents a model for image scanning and prediction.
4. Attributes:
   1. - imageBase64URL (str): Base64-encoded image data.
   2. - imageItemType (str): Type of the image item ('Number', 'Cat', etc.).
   3. - result (str): Result of the image scan or prediction.
5. Methods:
   1. - \_\_init\_\_(self, imageBase64URL, imageItemType, result): Initializes an instance of the ImageScanModel class.
   2. - \_\_str\_\_(self): Returns a string representation of the ImageScanModel instance.
   3. - to\_dict(self): Converts the instance to a dictionary.
   4. - from\_dict(cls, data): Creates an instance of ImageScanModel from a dictionary.
   5. - to\_json(self): Converts the instance to a JSON object.
   6. - from\_json(cls, json\_data): Creates an instance of ImageScanModel from a JSON object.

This file, image\_scan\_model.py, serves as the core module for image scanning and prediction within the Virtual Assistant for Image Classification (VAfIC) project. It defines the ImageScanModel class, acting as a model for image-related tasks .This file plays a crucial role in handling image-related information and acts as the backbone for image scanning and prediction tasks in the VAfIC project.

1. predict\_image.py
2. Description:
   1. Prediction script using a pre-trained fastai vision model.
3. Functions:
   1. - predict\_image(image\_path, model\_path='export.pkl'): Predicts the class of an image using a pre-trained fastai vision model.
4. Example Usage:
   1. >>> image\_path = 'kot.jpg'
   2. >>> prediction = predict\_image(image\_path)
   3. >>> print(f'Prediction: {prediction}')

The predict\_image.py file is a script designed for making predictions using a pre-trained fastai vision model. This script is a valuable component within the Virtual Assistant for Image Classification (VAfIC) project. The primary purpose is to predict the class of an image given its path, utilizing a pre-trained model specified by the model\_path parameter, which defaults to „export.pkl” This file enables seamless integration of image prediction capabilities into the VAfIC project, offering a straightforward way to obtain predictions for various image classes.

1. app.py
2. Description:
   1. Flask application for image scanning and prediction.
3. Dependencies:
   1. - Flask
   2. - flask\_cors
   3. - image\_scan\_model
   4. - PIL
   5. - fastai.vision.all
   6. - os
   7. - cv2
   8. - keras.models
   9. - tensorflow
   10. - numpy
4. Functions and Methods:
   1. - process\_data(soup, image): Process image data based on the specified category.
   2. - predict\_cat(data): Predict whether the image contains a cat or a dog.
   3. - predict\_digit(data):Predict the digit from the image.
   4. - decode\_base64\_image(data\_uri): Decode a Base64-encoded image.
   5. - getPath(path):Get the absolute path of the specified file.
   6. - scan\_image(): Endpoint to receive image data, process it, and return the prediction result.
5. Usage:
   1. - Run the Flask application on host "0.0.0.0" and port 8081.

The app.py file acts as the main Flask application responsible for image scanning and prediction in the Virtual Assistant for Image Classification (VAfIC) project. This file serves as the backend, leveraging various dependencies to provide a seamless and efficient image processing experience.

This file provides the core functionalities for handling incoming image data, performing predictions based on the specified category, and delivering the results through the defined API endpoint. It is the backbone of the VAfIC project's image processing capabilities.