Aeye

Release 0.1.5

Jun 18, 2020

Contents

A-eye

AI for the blind

Aeye aspires to be an open-source AI-as-a-Service platform for hobbyists and tinkerers to develops apps and services for the visually challenged. We want to provide the best tools and services for you to go out there and build cool stuff for our less privileged brothers and sisters, to empower them to prevail over their shortcomings so that they too can come forth and contribute and fully be a part of our society.

The latest advancements in Computer vision thanks to technologies like Deep Learning we are now able to augment human vision with computer vision models that help detect the objects around us, detect faces and identify who they are, describe the environment you are in and much more. This can be of great use to visually challenged people, to help them perform their day to day tasks more efficiently.

The service offers simple to use API end points that help deliver basic deep learning modules like Object Detection and Image Captioning.

To get started with the service get over to getting started page.

If your interested to contribute to the project happily head over to the Contributing page to know more.

CHAPTER 1

Getting started

Note: *The most reliable way to use this service is to host it on your own local machine. It does not take huge resources and standard cpu with 4gigs or ram is adequate. Aeye does provide an online service but it is hosted on a trial account in GCP and we don't know how often we can afford to run it.*

1.1 Using the Online Service

This is the easiest way to use the API. It accepts simple POST requests with the images and returns the corresponding responses in json format. Lets jump right in and test out the Image Captioning service.

The following curl request should generate the captions for the image

```
#TODO: host it
curl POST https://aeye-service.herokuapp.com/captioning -H "Content-Type:
image/*" --data-binary @YOUR-IMG.jpg -v
```

This will set a POST request to the service and will return a string with the caption that the model generated.

If you want to use it in a program, we recommend that you use libraries like `requests` for handling http requests to the service.

```
import requests

#TODO: Remaining
```

1.2 Offline Serving

You can also run the Aeye service in your local environment using BentoML. BentoML is used to pack and save the models that we use. Since the serving is based on BentoML I suggest you check it out to get a better idea about the library.

First git clone this repo to your local environment and install all the requirments.

```
$ git clone https://github.com/jjmachan/aeye
$ cd aeye
$ pip install -r requirements.text
```

Now you have all the dependencies setup. The next step is to download all the artifacts and put it in the artifacts page. Files like the trained weights for the models, the wordmap used are all downloaded from the appropriate links given bellow and all of them are to be placed in the *artifacts/* directory.

- Image Captioning:
  - Encoder - https://localhost
  - Decoder - https://localhost
  - WordMap - https://localhost
- Object Dection:
- Facial Recognition

With all of the artifacts downloaded we are now ready to pack the models and create the containerized version that can be hosted. Now BentoML is handling all of the hard work for us and all we have to do is run a couple of commands.

First lets save and back the model to a Bento repository.

```
$ python saveToBento.py
```

This will save the models, artifacts and pack all of it with the dependencies into a deployable containerized package. Now use BentoML to serve it.

```
$ bentoml serve AeyeService:latest
```

This launches a development server and you can head over to *localhost:5000* to access the built in API tester.

Now since we have the service up and running in your local environment all the steps for using the online service can be carried over. You can use curl or requests to sent HTTP requests to the APIs.

Now you set to build your stuff using our platform!

CHAPTER 2

Image Captioning Module

As you know Aeye has a modular approach when it comes to the various AI functionalities. Each Deep Learning model is in its own pypi package and only the model weights have to be downloaded and added to the *artifacts/* folder.

CHAPTER 3

Object Detection Module

The Object Detection Module is still under development and will be added asap stay Tuned!

CHAPTER 4

Facial Recognition Module

This module handles the Facial Recognition functionalities.The database for the different faces and detecting and matching faces to our database are all managed from this module.

CHAPTER 5

API Endpoints

These are the endpoints for our services.

CHAPTER 6

Contributing

Aeye was something my team build as part of our final year project so its just our humble side project but if liked it and found value in it we will love to hear it from you. Sent us your mails through jamesjithin97@gmail.com. We love to hear from you.

If you want to contribute we would suggest the following ways.

6.1 Building new Deeplearning Modules

Create new modules for Aeye. We just have 3 modules set up in this service and we will love to have more modules included. If you know about deep learning technologies and have some suggestions for some modules start a new github issue and we can discuss more about it.

6.2 Improving Existing Modules

Our existing modules still need work. As you know Deep Learning is a rapidly evolving space and we want the best of those innovations to reach our end-users. Currently we don't use the state of the art model in our modules but then again we are not interested in keeping our modules consistent with the bleeding edge. Even without top algorithms we can always improve the datasets and train it on new data to get better accuracies. Studies have shown that there is difference in the datasets need for training models for serving the visually challenged and those that standard computer vision tasks use and their interests are not always in alignment. So we hope incorporate changes that reflects this into your system.