hand-training-motion-tracking

Deep CNN, KMeans model and software for hand motion tracking

BENCHMARK SCRIPT WARNING

Please note that the benchmark script located in /model/benchmark will download the dataset by default!!!

We have included it for the curious reader, however running the script will take a while, if the data is not present in the root/dataset folder. Also, benchmarking on CPU will take several hours.

Should you still be interested (in either benchmarking or downloading the data), you run it from the model folder:

```
cd model
python -m benchmark.benchmark
```

Dataset

The model is trained using the HanCo dataset.
A copy is stored in a GCP Bucket along with a smaller subset for model experimentation.

Set up config file

Create a yaml file as follows:

```
SUBSET: Null # Null uses full dataset, otherwise specify int
IMAGENET_PARAMS:
  mean: [0.485, 0.456, 0.406]
  std: [0.229, 0.224, 0.225]
DATA_PATH: C:\Users\pete_\Documents\HanCo Dataset\HanCo_full
BATCH_SIZE: 256
RANDOM_SEED: 42
EPOCHS: 100
```

Virtual environment and dependencies

Create a virtual environment:

```
python -m venv venv
```

Activate the environment:

```
venv/Scripts/activate
```

Install dependencies from the requirements file:

```
pip install -r requirements.txt
```

Model training, scripts and modules

All following examples assume the current folder location is the model folder (located in the root folder). From the cmd root folder, simply run:

```
cd model
```

Training a deep CNN hand model

To train a hand model, simply run the following with a --config flag, following a path to a valid configuration file.

```
python train_handmodel.py --config path/to/config.yaml
```

Tensorboard

You may track progress via Tensorboard by navigating to hand-detection/model in a separate command line and running:

```
python -m tensorboard --logdir=logs --port 6006
```

You may then view Tensorboard by navigating to http://localhost:6006 in your browser.

Initialize models

To initialize a CNN model, load the config file and run the get_handmodel function from the model_setup module. If no valid model exists in the specified model path, it will default to a pretrained ResNet50.

A model instance is required by several parts of the code.

```
model = model_setup.get_handmodel("path/to/config.yaml")
```

Similarly, you can load a KMeans model:

```
kmeans = model_setup.get_kmeans("path/to/kmeans.pkl")
```

Load data

Loading a dataset is simply done using the HandDataset class.

The class expects that you have the rgb, xyz and calib parts of the HanCo dataset in your dataset path.

Setting apply_augmentation to True (default) will apply dataset level augmentation (random rotation and flipping)

```
dataset = HandDataset("path/to/dataset/")
```

Visualizations

Some visualizations can be run directly, but require a path specification to be saved. Others might require some input, such as images or an intrinsic matrix K.

Below is an example of a cluster visualization, saving the plot to a folder.

```
visualizations.visualize_clusters(exercise = "fstretch", exercise_state = "0000", kmeans_path =
```

Running inference on images

Running inference on images located in ./model/images can be done by simply running:

```
python run_inference.py --config path/to/config.yaml
```