README

An Yu, Ferhat Demirkiran, Jeremy Varghese, Peter Buonaiuto
Computer Science Department, University at Albany – SUNY

1 Module I: Animal Identification

1.1 Data Preparation

This guide provides detailed instructions for preparing your video dataset for training and prediction using the Zamba package, please refer to zamba_prepare_data.py and zamba_video_prepare.py files in our github website

1.1.1 Training Data

For training the model, your data must be organized into two key components:

- **Video Directory**: A directory containing all the video files you wish to use for training.

  Example directory structure:

  ```
  /../dataset/zamba_train_video
  ```

- **Labels CSV File**: A CSV file listing each video file along with its corresponding species label. This file should have two columns:

  1. `file_path`: Relative or absolute path to the video file.
  2. `species_label`: The label of the species recorded in the video.

  Example CSV (`zamba_train.csv`):

  ```
  file_path,species_label
  video1.mp4,bird
  video2.mp4,mammal
  video3.mp4,reptile
  ```

1.1.2 Testing Data

To prepare for making predictions with your trained model, simply organize your videos within a specific directory.

Example directory structure:

```
/../dataset/zamba_pred_video
```

- Example of the path to each video file for prediction:

  ```
  test_video1.mp4
  test_video2.mp4
  ```

1.2 Setting Up Zamba Working Environment

In this part, we gave the steps to configure our specific working environment for using Zamba.

```
!python --version
# python version 3.8
!conda install ffmpeg
!ffmpeg -version
# ffmpeg version 4.4.2
# Install Zamba from the latest release
!pip install https://github.com/drivendataorg/zamba/releases
/latest/download/zamba.tar.gz
# Verify Zamba installation , it is 2.3.2
!pip show zamba
!pip install opencv-python-headless==4.5.5.62
!pip install typer==0.9.0
```

1.3 Training the Model with Zamba

Before initiating the training process we modified the Zamba configuration for the time_distributed model.

```
# Early stopping criteria: mode = min, monitor = val_loss , patience = 3
```

To train your model using Zamba, run the following command in your terminal, adjusting paths as necessary to match your directory structure:

```
zamba train --data-dir /../dataset/zamba_train_video
--labels /../dataset/zamba_prep/zamba_trainV.csv
--save-dir /../zamba_prep/zamba_out
```

1.4 Making Predictions

After training, you can use the trained model to make predictions on new videos. Update your train_configuration.yaml file with the appropriate paths for conducting the prediction, predict_config is required as we will be running inference on unseen test data.

```
predict_config:
  checkpoint: /../zamba_prep/zamba_out/../time_distributed.ckpt
  data_dir: /../dataset/zamba_pred_video/
  save_dir: /../zamba_prep/zamba_out/
```

Important parameters in the 'train_configuration.yaml' include:

- **checkpoint**: Path to the checkpoint file of your trained model, which is required to load the model for making predictions.
- **data_dir**: The directory that holds the videos for which you intend to predict.
- **save_dir**: The directory where the outcomes of your predictions will be stored.

With your videos appropriately placed in the specified directory, proceed to predictions by executing the following command, referring to your configured YAML file:

```
zamba predict --config /../zamba_prep/zamba_out/train_configuration.yaml
```

Please refer to the zamba github website for training and testing details:
https://github.com/drivendataorg/zamba

2 Module II: Behavior Recognition

2.1 Setup

Install all dependencies for the program

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

2.2 Data Preparation

Our primary objective with data preparation was to split the dataset into multiple more specific datasets, which we hypothesized would improve our test scores if we have different specialized models which are invoked based on Zamba's prediction on which model is most applicable to the input. `datasets/transforms_ss.py` has functionality to prepare the data through various means of transformation to ensure input consistency.

2.2.1 Purging Redundancies

The data input is train.csv, with an accessory file `AR_metadata.xlsx`. The two files together define the animals and actions found in each frame of each video. `train.csv` was minimized to `train_light.csv` (and the same for `val.csv`) to remove redundant information.

2.2.2 Training Specialized Models

The data is split by running `split_train.py` which generates different data csv files for each subclass of animal category.

2.3 Training and Testing

2.3.1 Training

Run `main_split.py` to train one model for each subclass of animal detected from the training data. One can specify to perform this in parallel if there is sufficient hardware by using the argument

```
  --parallel True
```

The output will consist of a checkpoint.pth file containing weights for evaluation for each specialized model, located in Output/currentdate.

2.3.2 Testing

To test the model on the Animal Behavior dataset, you can run `main.py`, either full or split, with the parameter

```
    --train False
```

The validation data provided in the datasets will also be split during data preparation.

3 Module III: Evaluation

Before running evaluation, please modify the all relative file paths in evaluation.py
Please execute the following command:

```
!python3 evaluation.py
```

4 Access to AnimalKingdom dataset

Please refer to the AnimalKingdom github website:
https://sutdcv.github.io/Animal-Kingdom/

For all checkpoints files and results from zamba, please contact any one of the authors.