EmoPy Documentation

Release 1.0

AP

Sep 21, 2018

Contents:

CHAPTER 1

FERModel

class fermodel.**FERModel**(*target_emotions*, *verbose=False*)

Pretrained deep learning model for facial expression recognition.

Parameters

- **target_emotions** – set of target emotions to classify
- **verbose** – if true, will print out extra process information

Example:

```
from fermodel import FERModel

target_emotions = ['happiness', 'disgust', 'surprise']
model = FERModel(target_emotions, verbose=True)
```

POSSIBLE_EMOTIONS = ['anger', 'fear', 'calm', 'sadness', 'happiness', 'surprise', 'dis

predict(*image_file*)

Predicts discrete emotion for given image.

Parameters **images** – image file (jpg or png format)

CHAPTER 2

FERNeuralNets

class neuralnets.**ConvolutionalLstmNN**(*image_size*, *channels*, *emotion_map*, *time_delay=2*, *filters=10*, *kernel_size=(4, 4)*, *activation='sigmoid'*, *verbose=False*)

Convolutional Long Short Term Memory Neural Network.

Parameters

- **image_size** – dimensions of input images
- **channels** – number of image channels
- **emotion_map** – dict of target emotion label keys with int values corresponding to the index of the emotion probability in the prediction output array
- **time_delay** – number of time steps for lookback
- **filters** – number of filters/nodes per layer in CNN
- **kernel_size** – size of sliding window for each layer of CNN
- **activation** – name of activation function for CNN
- **verbose** – if true, will print out extra process information

Example:

```
net = ConvolutionalLstmNN(target_dimensions=(64,64), channels=1, target_labels=[0,
↪1,2,3,4,5,6], time_delay=3)
net.fit(features, labels, validation_split=0.15)
```

fit(*features*, *labels*, *validation_split*, *batch_size=10*, *epochs=50*)

Trains the neural net on the data provided.

Parameters

- **features** – Numpy array of training data.
- **labels** – Numpy array of target (label) data.

- **validation_split** – Float between 0 and 1. Percentage of training data to use for validation
- **batch_size** –
- **epochs** – number of times to train over input dataset.

class neuralnets.**ConvolutionalNN**(*image_size*, *channels*, *emotion_map*, *filters=10*, *kernel_size=(4, 4)*, *activation='relu'*, *verbose=False*)

2D Convolutional Neural Network

Parameters

- **image_size** – dimensions of input images
- **channels** – number of image channels
- **emotion_map** – dict of target emotion label keys with int values corresponding to the index of the emotion probability in the prediction output array
- **filters** – number of filters/nodes per layer in CNN
- **kernel_size** – size of sliding window for each layer of CNN
- **activation** – name of activation function for CNN
- **verbose** – if true, will print out extra process information

Example:

```
net = ConvolutionalNN(target_dimensions=(64,64), channels=1, target_labels=[0,1,2,
↪3,4,5,6], time_delay=3)
net.fit(features, labels, validation_split=0.15)
```

fit(*image_data*, *labels*, *validation_split*, *epochs=50*)

Trains the neural net on the data provided.

Parameters

- **image_data** – Numpy array of training data.
- **labels** – Numpy array of target (label) data.
- **validation_split** – Float between 0 and 1. Percentage of training data to use for validation
- **batch_size** –
- **epochs** – number of times to train over input dataset.

class neuralnets.**TimeDelayConvNN**(*image_size*, *channels*, *emotion_map*, *time_delay*, *filters=32*, *kernel_size=(1, 4, 4)*, *activation='relu'*, *verbose=False*)

The Time-Delayed Convolutional Neural Network model is a 3D-Convolutional network that trains on 3-dimensional temporal image data. One training sample will contain n number of images from a series and its emotion label will be that of the most recent image.

Parameters

- **image_size** – dimensions of input images
- **time_delay** – number of past time steps included in each training sample
- **channels** – number of image channels
- **emotion_map** – dict of target emotion label keys with int values corresponding to the index of the emotion probability in the prediction output array

- **filters** – number of filters/nodes per layer in CNN
- **kernel_size** – size of sliding window for each layer of CNN
- **activation** – name of activation function for CNN
- **verbose** – if true, will print out extra process information

Example:

```
model = TimeDelayConvNN(target_dimensions={64,64), time_delay=3, channels=1,␣
↪label_count=6)
model.fit(image_data, labels, validation_split=0.15)
```

fit (*image_data*, *labels*, *validation_split*, *epochs=50*)
Trains the neural net on the data provided.

Parameters

- **image_data** – Numpy array of training data.
- **labels** – Numpy array of target (label) data.
- **validation_split** – Float between 0 and 1. Percentage of training data to use for validation
- **batch_size** –
- **epochs** – number of times to train over input dataset.

class neuralnets.**TransferLearningNN** (*model_name*, *emotion_map*)
Transfer Learning Convolutional Neural Network initialized with pretrained weights.

Parameters

- **model_name** – name of pretrained model to use for initial weights. Options: ['Xception', 'VGG16', 'VGG19', 'ResNet50', 'InceptionV3', 'InceptionResNetV2']
- **emotion_map** – dict of target emotion label keys with int values corresponding to the index of the emotion probability in the prediction output array

Example:

```
model = TransferLearningNN(model_name='inception_v3', target_labels=[0,1,2,3,4,5,
↪6])
model.fit(images, labels, validation_split=0.15)
```

fit (*features*, *labels*, *validation_split*, *epochs=50*)
Trains the neural net on the data provided.

Parameters

- **features** – Numpy array of training data.
- **labels** – Numpy array of target (label) data.
- **validation_split** – Float between 0 and 1. Percentage of training data to use for validation
- **epochs** – Max number of times to train over dataset.

CHAPTER 3

CSVDataLoader

class csv_data_loader.**CSVDataLoader**(*target_emotion_map*, *datapath=None*, *validation_split=0.2*, *image_dimensions=None*, *csv_label_col=None*, *csv_image_col=None*, *out_channels=1*)

DataLoader subclass loads image and label data from csv file.

Parameters

- **emotion_map** – Dict of target emotion label values and their corresponding label vector index values.
- **datapath** – Location of image dataset.
- **validation_split** – Float percentage of data to use as validation set.
- **image_dimensions** – Dimensions of sample images (height, width).
- **csv_label_col** – Index of label value column in csv.
- **csv_image_col** – Index of image column in csv.
- **out_channels** – Number of image channels.

load_data()

Loads image and label data from specified csv file path.

Returns Dataset object containing image and label data.

CHAPTER 4

DirectoryDataLoader

class directory_data_loader.**DirectoryDataLoader**(*target_emotion_map=None*, *datapath=None*, *validation_split=0.2*, *out_channels=1*, *time_delay=None*)

DataLoader subclass loads image and label data from directory.

Parameters

- **target_emotion_map** – Optional dict of target emotion label values/strings and their corresponding label vector index values.
- **datapath** – Location of image dataset.
- **validation_split** – Float percentage of data to use as validation set.
- **out_channels** – Number of image channels.
- **time_delay** – Number of images to load from each time series sample. Parameter must be provided to load time series data and unspecified if using static image data.

load_data()

Loads image and label data from specified directory path.

Returns Dataset object containing image and label data.

CHAPTER 5

DataGenerator

class data_generator.**DataGenerator**(*time_delay=None*)

config_augmentation(*zca_whitening=False*, *rotation_angle=90*, *shift_range=0.2*, *horizontal_flip=True*, *time_delay=None*)

fit(*images*, *labels*)

generate(*target_dimensions=None*, *batch_size=10*)

get_next_batch(*batch_size=10*, *target_dimensions=None*)

CHAPTER 6

Dataset

class dataset.**Dataset**(*train_images*, *test_images*, *train_labels*, *test_labels*, *emotion_index_map*, *time_delay=None*)

get_emotion_index_map()

get_test_data()

get_time_delay()

get_training_data()

num_images()

num_test_images()

num_train_images()

print_data_details()

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index