tf.keras.preprocessing.image.ImageDataGenerator

Class `ImageDataGenerator`

Defined in `tensorflow/python/keras/_impl/keras/preprocessing/image.py`.

Generate minibatches of image data with real-time data augmentation.

Arguments:

- `featurewise_center`: set input mean to 0 over the dataset.
- `samplewise_center`: set each sample mean to 0.
- `featurewise_std_normalization`: divide inputs by std of the dataset.
- `samplewise_std_normalization`: divide each input by its std.
- `zca_whitening`: apply ZCA whitening.
- `zca_epsilon`: epsilon for ZCA whitening. Default is 1e-6.
- `rotation_range`: degrees (0 to 180).
- `width_shift_range`: fraction of total width.
- `height_shift_range`: fraction of total height.
- `shear_range`: shear intensity (shear angle in radians).
- `zoom_range`: amount of zoom. if scalar z, zoom will be randomly picked in the range [1-z, 1+z]. A sequence of two can be passed instead to select this range.
- `channel_shift_range`: shift range for each channels.
- `fill_mode`: points outside the boundaries are filled according to the given mode ('constant', 'nearest', 'reflect' or 'wrap'). Default is 'nearest'.
- `cval`: value used for points outside the boundaries when fill_mode is 'constant'. Default is 0.
- `horizontal_flip`: whether to randomly flip images horizontally.
- `vertical_flip`: whether to randomly flip images vertically.
- `rescale`: rescaling factor. If None or 0, no rescaling is applied, otherwise we multiply the data by the value provided. This is applied after the `preprocessing_function` (if any provided) but before any other transformation.
- `preprocessing_function`: function that will be implied on each input. The function will run before any other modification on it. The function should take one argument: one image (Numpy tensor with rank 3), and should output a Numpy tensor with the same shape.
- `data_format`: 'channels_first' or 'channels_last'. In 'channels_first' mode, the channels dimension (the depth) is at

index 1, in 'channels_last' mode it is at index 3. It defaults to the `image_data_format` value found in your Keras config file at `~/.keras/keras.json`. If you never set it, then it will be "channels_last".

Methods

`__init__`

```
__init__(
    featurewise_center=False,
    samplewise_center=False,
    featurewise_std_normalization=False,
    samplewise_std_normalization=False,
    zca_whitening=False,
    zca_epsilon=1e-06,
    rotation_range=0.0,
    width_shift_range=0.0,
    height_shift_range=0.0,
    shear_range=0.0,
    zoom_range=0.0,
    channel_shift_range=0.0,
    fill_mode='nearest',
    cval=0.0,
    horizontal_flip=False,
    vertical_flip=False,
    rescale=None,
    preprocessing_function=None,
    data_format=None
)
```

`fit`

```
fit(
    x,
    augment=False,
    rounds=1,
    seed=None
)
```

Fits internal statistics to some sample data.

Required for featurewise_center, featurewise_std_normalization and zca_whitening.

Arguments:

- `x`: Numpy array, the data to fit on. Should have rank 4. In case of grayscale data, the channels axis should have value 1, and in case of RGB data, it should have value 3.
- `augment`: Whether to fit on randomly augmented samples
- `rounds`: If `augment`, how many augmentation passes to do over the data
- `seed`: random seed.

Raises:

- `ValueError`: in case of invalid input `x`.
- `ImportError`: if Scipy is not available.

`flow`

```
flow(
    x,
    y=None,
    batch_size=32,
    shuffle=True,
    seed=None,
    save_to_dir=None,
    save_prefix='',
    save_format='png'
)
```

`flow_from_directory`

```
flow_from_directory(
    directory,
    target_size=(256, 256),
    color_mode='rgb',
    classes=None,
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=None,
    save_to_dir=None,
    save_prefix='',
    save_format='png',
    follow_links=False
)
```

`random_transform`

```
random_transform(
    x,
    seed=None
)
```

Randomly augment a single image tensor.

Arguments:

- `x`: 3D tensor, single image.
- `seed`: random seed.

Returns:

A randomly transformed version of the input (same shape).

Raises:

- `ImportError`: if Scipy is not available.

`standardize`

```
standardize(x)
```

Apply the normalization configuration to a batch of inputs.

Arguments:

- `x`: batch of inputs to be normalized.

Returns:

The inputs, normalized.
