Deep Learning - lab 8

Data augmentation and transfer learning

Prof. Stefano Carrazza

University of Milan and INFN Milan

Loading data from files

Keras get file

```
import tensorflow as tf

# example tgz file from:
dataset_url = 'https://somelocation/../../data.tgz'
data_dir = tf.keras.utils.get_file('save_location',
                                   origin=dataset_url,
                                   extract=True)
```

Keras get file

```
import tensorflow as tf

# example zip file from:
dataset_url = 'https://somelocation/../../data.zip'
data_dir = tf.keras.utils.get_file('data_features.zip',
                                   origin=dataset_url,
                                   extract=True)
```

Keras get file

Suppose your data directory structure is:

```
main_directory/
...class_a/
......a_image_1.jpg
......a_image_2.jpg
...class_b/
......b_image_1.jpg
......b_image_2.jpg
```

Keras image dataset from directory

```
train_ds = tf.keras.utils.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="training",
  image_size=(img_height, img_width),
  batch_size=batch_size)

# extract class names
class_names = train_ds.class_names

# train_ds is a tf.Dataset
```

Keras image dataset from directory

```
val_ds = tf.keras.utils.image_dataset_from_directory(
  data_dir,
  validation_split=0.2,
  subset="validation",
  image_size=(img_height, img_width),
  batch_size=batch_size)
```

tf.data.Dataset

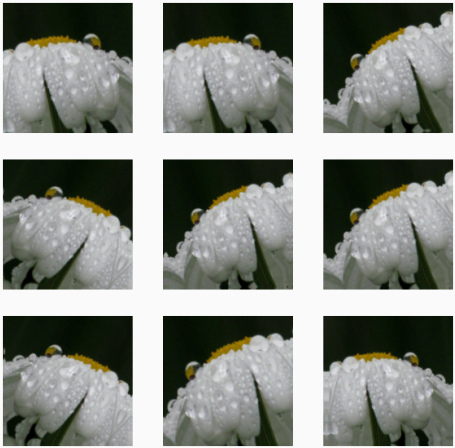
```
# creates dataset with at most 1 element
train_ds.take(1)

# keeps the images in memory after they're loaded off disk at epoch=1
# if dataset is too large to fit into memory, this method can create a
# performance on-disk cache.
train_ds.cache()

# Randomly shuffles the elements of this dataset.
train_ds.cache().shuffle(1000)

# overlaps data preprocessing and model execution while training
train_ds.cache().prefetch(tf.data.AUTOTUNE)
```

Data augmentation

Builtin methods

```
data_augmentation = tf.keras.models.Sequential(
  [
    tf.keras.layers.RandomFlip("horizontal",
                               input_shape=(img_height, img_width, 3)),
    tf.keras.RandomRotation(0.1),
    tf.keras.RandomZoom(0.1),
  ]
)
```

Builtin methods



Transfer learning

Loading model

```
base_model = tf.keras.applications.MobileNetV2(input_shape=(IMG_SHAPE),
                                               include_top=False,
                                               weight='imagenet')
```