Computer vision with TensorFlow

TensorFlow provides a number of computer vision (CV) and image classification tools. This document introduces some of these tools and provides an overview of resources to help you get started with common CV tasks.

Vision libraries and tools

TensorFlow provides CV tools through the higher-level Keras libraries and the lower-level `tf.image` (https://www.tensorflow.org/api_docs/python/tf/image) module. For most use cases, the Keras libraries will be more convenient than the built-in TensorFlow alternatives. But if the Keras options don't fit your use case, or you want lower-level control over image preprocessing, you might need the lower-level TensorFlow tools.

KerasCV

If you're just getting started with a CV project, and you're not sure which libraries and tools you'll need, KerasCV (https://keras.io/keras_cv/) is a good place to start. KerasCV is a library of modular CV components built on Keras Core. KerasCV includes models, layers, metrics, callbacks, and other tools that extend the high-level Keras API for CV tasks. The KerasCV APIs can help with data augmentation, classification, object detection, segmentation, image generation, and other common CV workflows. You can use KerasCV to quickly assemble production-grade, state-of-the-art training and inference pipelines.

Keras utilities

`tf.keras.utils` (https://www.tensorflow.org/api_docs/python/tf/keras/utils) provides several high-level image preprocessing utilities. For example, `tf.keras.utils.image_dataset_from_directory` (https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image_dataset_from_directory) generates a `tf.data.Dataset` (https://www.tensorflow.org/api_docs/python/tf/data/Dataset) from a directory of images on disk.

`tf.image` (https://www.tensorflow.org/api_docs/python/tf/image)

If KerasCV doesn't fit your use case, you can use `tf.image` (https://www.tensorflow.org/api_docs/python/tf/image) and `tf.data` (https://www.tensorflow.org/api_docs/python/tf/data) to write your own data augmentation pipelines or layers.

The `tf.image` (https://www.tensorflow.org/api_docs/python/tf/image) module contains various functions for image processing, such as `tf.image.flip_left_right` (https://www.tensorflow.org/api_docs/python/tf/image/flip_left_right), `tf.image.rgb_to_grayscale` (https://www.tensorflow.org/api_docs/python/tf/image/rgb_to_grayscale), `tf.image.adjust_brightness` (https://www.tensorflow.org/api_docs/python/tf/image/adjust_brightness), `tf.image.central_crop` (https://www.tensorflow.org/api_docs/python/tf/image/central_crop), and `tf.image.stateless_random*`.

The `tf.data` (https://www.tensorflow.org/api_docs/python/tf/data) API enables you to build complex input pipelines from simple, reusable pieces.

TensorFlow Datasets

TensorFlow Datasets (https://www.tensorflow.org/datasets) is a collection of datasets ready to use with TensorFlow. Many of the datasets (for example, MNIST (https://www.tensorflow.org/datasets/catalog/mnist), Fashion-MNIST (https://www.tensorflow.org/datasets/catalog/fashion_mnist), and TF Flowers (https://www.tensorflow.org/datasets/catalog/tf_flowers)) can be used to develop and test computer vision algorithms.

Where to start

The following resources will help you get up and running with TensorFlow and Keras CV tools.

- KerasCV (https://keras.io/keras_cv/): Documentation and resources for KerasCV.
- KerasCV developer guides (https://keras.io/guides/keras_cv/): Guides to performing common CV tasks using KerasCV. If you're new to KerasCV, Classification with KerasCV (https://keras.io/guides/keras_cv/classification_with_keras_cv/) is a good place to start.
- TensorFlow tutorials (https://www.tensorflow.org/tutorials): The core TensorFlow documentation (this guide) includes a number of CV and image processing tutorials.

- Basic classification: Classify images of clothing (https://www.tensorflow.org/tutorials/keras/classification): Train a neural network model to classify images of clothing, like sneakers and shirts.
- Load and preprocess images (https://www.tensorflow.org/tutorials/load_data/images): Load and preprocess an image dataset in three ways:
    1. Use high-level Keras preprocessing utilities to read a directory of images on disk.
    2. Write your own input pipeline from scratch using `tf.data` (https://www.tensorflow.org/guide/data).
    3. Download a dataset from the large catalog (https://www.tensorflow.org/datasets/catalog/overview) available in TensorFlow Datasets (https://www.tensorflow.org/datasets).
- Load video data (https://www.tensorflow.org/tutorials/load_data/video): Load and preprocess AVI video data using the UCF101 human action dataset (https://www.tensorflow.org/datasets/catalog/ucf101).
- Convolutional Neural Network (CNN) (https://www.tensorflow.org/tutorials/images/cnn): Train a simple Convolutional Neural Network (https://developers.google.com/machine-learning/glossary/#convolutional_neural_network) (CNN) to classify CIFAR images (https://www.cs.toronto.edu/%7Ekriz/cifar.html) using the Keras API (https://www.tensorflow.org/guide/keras/overview).
- Image classification (https://www.tensorflow.org/tutorials/images/classification): Classify images of flowers using a `tf.keras.Sequential` (https://www.tensorflow.org/api_docs/python/tf/keras/Sequential) model and load data using `tf.keras.utils.image_dataset_from_directory` (https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image_dataset_from_directory)
.
- Transfer learning and fine-tuning (https://www.tensorflow.org/tutorials/images/transfer_learning): Classify images of cats and dogs by using transfer learning from a pre-trained network.
- Data augmentation (https://www.tensorflow.org/tutorials/images/data_augmentation): Increase the diversity of your training set by applying random (but realistic) transformations, such as image rotation.

- Image segmentation (https://www.tensorflow.org/tutorials/images/segmentation): Perform image segmentation, using a modified U-Net (https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/).
- Video classification with a 3D convolutional neural network (https://www.tensorflow.org/tutorials/video/video_classification): Train a 3D convolutional neural network (CNN) for video classification using the UCF101 (https://www.crcv.ucf.edu/data/UCF101.php) action recognition dataset.
- Transfer learning for video classification with MoViNet (https://www.tensorflow.org/tutorials/video/transfer_learning_with_movinet): Use a pre-trained MoViNet model and the UCF101 dataset (https://www.crcv.ucf.edu/data/UCF101.php) to classify videos for an action recognition task.