train_object_detection_in_colab

February 22, 2020

[ ]:
```
# Original Notebook
# https://colab.research.google.com/github/Tony607/object_detection_demo/blob/master/tensorflow_object_detection_training_colab.ipynb
```

[ ]:
```
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')

# Install Tensorflow Object Detection Api
!apt-get install protobuf-compiler python-pil python-lxml python-tk
!pip install Cython
!git clone https://github.com/tensorflow/models.git
%cd /content/models/research
!protoc object_detection/protos/*.proto --python_out=.
%set_env PYTHONPATH=$PYTHONPATH:/content/models/research:/content/models/research/slim:/content/models/models/research/object_detection
!python object_detection/builders/model_builder_test.py

# Inatall Tensorflow Gpu V1.14
!pip uninstall -y tensorflow
!pip install tensorflow-gpu==1.14
import tensorflow as tf
import os
```

[ ]:
```
# Name of Dataset
dataset_name = 'OI_Animals_Augmented_9_3000'

selected_model = 'faster_rcnn_inception_v2'

epochs = 50

eval_metrics = 'coco_detection_metrics'

convert_to_grayscale = False

fixed_size = False # fÃ¼r faster rcnn
```

```
is_faster_rcnn = True

use_dropout = False

l2_regularizer = 0

output_postfix = ''

MODELS_CONFIG = {
    'ssd_mobilenet_v2': {
        'model_name': 'ssd_mobilenet_v2_coco_2018_03_29',
        'pipeline_file': 'ssd_mobilenet_v2_coco.config',
        'batch_size': 12
    },
    'ssd_inception_v2': {
        'model_name': 'ssd_inception_v2_coco_2018_01_28',
        'pipeline_file': 'ssd_inception_v2_coco.config',
        'batch_size': 12
    },
    'faster_rcnn_inception_v2': {
        'model_name': 'faster_rcnn_inception_v2_coco_2018_01_28',
        'pipeline_file': 'faster_rcnn_inception_v2_coco.config',
        'batch_size': 1
    },
    'rfcn_resnet101': {
        'model_name': 'rfcn_resnet101_coco_2018_01_28',
        'pipeline_file': 'rfcn_resnet101_pets.config',
        'batch_size': 1
    },
    'faster_rcnn_resnet50': {
        'model_name': 'faster_rcnn_resnet50_coco_2018_01_28',
        'pipeline_file': 'faster_rcnn_resnet50_coco.config',
        'batch_size': 1
    }
}

MODEL = MODELS_CONFIG[selected_model]['model_name']
pipeline_file = MODELS_CONFIG[selected_model]['pipeline_file']
batch_size = MODELS_CONFIG[selected_model]['batch_size']
output_dir = '/content/drive/My\ Drive/' + dataset_name + '/' + MODEL + '_out' +␣
 ↪output_postfix
```

[ ]:
```
# option 1: clone dataset from git reposity

# replace: <user>, <passwort>, <repo_name>
% cd /content/
!git clone https://<user>:<passwort>!@gitlab.com/user/<repo_name>.git
```

```
!mkdir /content/$dataset_name
!mkdir /content/$dataset_name/data
% cd /content/$dataset_name
!mv /content/<repo_name>/* data
!rm -r /content/<repo_name>
```

[ ]:
```
# option 2: copy from drive
!mkdir /content/$dataset_name
% cd /content/$dataset_name
!cp -r /content/drive/My\ Drive/$dataset_name/data/train.record data
```

[ ]:
```
% cd /content/$dataset_name
# calculate train and eval steps (optional)
num_classes = len([c for c in open('data/label_map.pbtxt').readlines() if 'item␣
 ↪{' in c])
num_eval_examples = sum(1 for _ in tf.python_io.tf_record_iterator('data/test.
 ↪record'))
num_train_examples=sum(1 for _ in tf.python_io.tf_record_iterator('data/train.
 ↪record'))
num_steps = epochs * (num_train_examples // batch_size)
num_steps
```

[ ]:
```
# Download model and config file
% cd /content/$dataset_name

import os
import shutil
import glob
import urllib.request
import tarfile
MODEL_FILE = MODEL + '.tar.gz'
DOWNLOAD_BASE = 'http://download.tensorflow.org/models/object_detection/'
DEST_DIR = '/content/' + dataset_name + '/'

if not (os.path.exists(MODEL_FILE)):
    urllib.request.urlretrieve(DOWNLOAD_BASE + MODEL_FILE, DEST_DIR + MODEL_FILE)
tar = tarfile.open(DEST_DIR + MODEL_FILE)
tar.extractall()
tar.close()
os.remove(DEST_DIR + MODEL_FILE)

fine_tune_checkpoint = os.path.join(DEST_DIR + MODEL, "model.ckpt")

pipeline_fname = os.path.join('/content/models/research/object_detection/samples/
 ↪configs/', pipeline_file)
assert os.path.isfile(pipeline_fname), '`{}` not exist'.format(pipeline_fname)
```

[ ]:
```
# configfile settings and create copy in drive outputdir

import re
data_dir = '/content/' + dataset_name + '/data/'
pipeline_config_file = data_dir + pipeline_file

with open(pipeline_fname) as f:
    s = f.read()
with open(pipeline_config_file, 'w') as f:

    # fine_tune_checkpoint
    s = re.sub('fine_tune_checkpoint: ".*?"',
               'fine_tune_checkpoint: "{}"'.format(fine_tune_checkpoint), s)

    # tfrecord files train and test.
    s = re.sub(
        '(input_path: ".*?)(train.record)(.*?")', 'input_path: "{}"'.
 ↪format(data_dir + 'train.record'), s)
    s = re.sub(
        '(input_path: ".*?)(val.record)(.*?")', 'input_path: "{}"'.
 ↪format(data_dir + 'test.record'), s)

    # label_map_path
    s = re.sub(
        'label_map_path: ".*?"', 'label_map_path: "{}"'.format(data_dir +␣
 ↪'label_map.pbtxt'), s)

    # Set training batch_size.
    s = re.sub('batch_size: [0-9]+',
               'batch_size: {}'.format(batch_size), s)

    # Set training steps, num_steps
    s = re.sub('num_steps: [0-9]+',
               'num_steps: {}'.format(num_steps), s)

    # Set number of eval examples
    s = re.sub('num_examples: [0-9]+',
               'num_examples: {}'.format(num_eval_examples), s)

    # Set eval metrics
    metrics_set: "open_images_V2_detection_metrics"
    s = re.sub('metrics_set: "open_images_V2_detection_metrics"',
               'metrics_set: "{}"'.format(eval_metrics), s)

    s = re.sub('max_evals: [0-9]+',
              'metrics_set: "{}"'.format(eval_metrics), s)
```

```
    # Set number of classes num_classes.
    s = re.sub('num_classes: [0-9]+',
               'num_classes: {}'.format(num_classes), s)

    if l2_regularizer != 0:
      s = re.sub('  weight: 0.0',
                 '  weight: {}'.format(l2_regularizer), s)

    if use_dropout:
      s = re.sub('use_dropout: false',
                 'use_dropout: true', s)

    if is_faster_rcnn:
      if fixed_size:

        s = re.sub('keep_aspect_ratio_resizer \{\n        min_dimension: 600\n  ␣
↪     max_dimension: 1024\n      \}',
                   'fixed_shape_resizer {\n        height: 600\n        width:␣
↪600\n        convert_to_grayscale: ' + str(convert_to_grayscale).lower() + '\n␣
↪       resize_method: AREA\n      }', s)

      else:
        if convert_to_grayscale:

          s = re.sub('keep_aspect_ratio_resizer \{\n        min_dimension: 600\n␣
↪       max_dimension: 1024\n      \}',
                   'keep_aspect_ratio_resizer {\n        min_dimension: 600\n    ␣
↪   max_dimension: 1024\n        convert_to_grayscale: ' +␣
↪str(convert_to_grayscale).lower() + '\n      }', s)

    elif convert_to_grayscale: # ssd kann nur fixed

      s = re.sub('fixed_shape_resizer \{\n        height: 300\n        width:␣
↪300\n      \}',
                 'fixed_shape_resizer {\n        height: 300\n        width:␣
↪300\n        convert_to_grayscale: true\n      }', s)

    f.write(s)
!mkdir $output_dir
!cp -r $pipeline_config_file $output_dir
```

[ ]:
```
# copy config file from drive outputdir
!cp $output_dir/$pipeline_file data/$pipeline_file
```

[ ]:
```
# output config file
!cat {pipeline_config_file}
```

[ ]:
```
# start and init tensorboard (only when training already started once)
%load_ext tensorboard
%tensorboard --logdir $output_dir
```

[ ]:
```
# start training
!python /content/models/research/object_detection/model_main.py \
    --pipeline_config_path={pipeline_config_file} \
    --model_dir={output_dir} \
    --alsologtostderr \
    --num_train_steps={num_steps}
```