Release Notes

September 3rd, 2020

TF2 OD API models can now be converted to TensorFlow Lite! Only SSD models currently supported. See documentation.

Thanks to contributors: Sachin Joglekar

July 10th, 2020

We are happy to announce that the TF OD API officially supports TF2! Our release includes:

- New binaries for train/eval/export that are designed to run in eager mode.
- A suite of TF2 compatible (Keras-based) models; this includes migrations of our most popular TF1.x models (e.g., SSD with MobileNet, RetinaNet, Faster R-CNN, Mask R-CNN), as well as a few new architectures for which we will only maintain TF2 implementations:
  1. CenterNet - a simple and effective anchor-free architecture based on the recent Objects as Points paper by Zhou et al
  2. EfficientDet - a recent family of SOTA models discovered with the help of Neural Architecture Search.
- COCO pre-trained weights for all of the models provided as TF2 style object-based checkpoints.
- Access to Distribution Strategies for distributed training — our model are designed to be trainable using sync multi-GPU and TPU platforms.
- Colabs demo'ing eager mode training and inference.

See our release blogpost here. If you are an existing user of the TF OD API using TF 1.x, don't worry, we've got you covered.

Thanks to contributors: Akhil Chinnakotla, Allen Lavoie, Anirudh Vegesana, Anjali Sridhar, Austin Myers, Dan Kondratyuk, David Ross, Derek Chow, Jaeyoun Kim, Jing Li, Jonathan Huang, Jordi Pont-Tuset, Karmel Allison, Kathy Ruan, Kaushik Shivakumar, Lu He, Mingxing Tan, Pengchong Jin, Ronny Votel, Sara Beery, Sergi Caelles Prat, Shan Yang, Sudheendra Vijayanarasimhan, Tina Tian, Tomer Kaftan, Vighnesh Birodkar, Vishnu Banna, Vivek Rathod, Yanhui Liang, Yiming Shi, Yixin Shi, Yu-hui Chen, Zhichao Lu.

June 26th, 2020

We have released SSDLite with MobileDet GPU backbone, which achieves 17% mAP higher than the MobileNetV2 SSDLite (27.5 mAP vs 23.5 mAP) on a NVIDIA Jetson Xavier at comparable latency (3.2ms vs 3.3ms).

Along with the model definition, we are also releasing model checkpoints trained on the COCO dataset.

Thanks to contributors: Yongzhe Wang, Bo Chen, Hanxiao Liu, Le An (NVIDIA), Yu-Te Cheng (NVIDIA), Oliver Knieps (NVIDIA), and Josh Park (NVIDIA).

June 17th, 2020

We have released Context R-CNN, a model that uses attention to incorporate contextual information images (e.g. from temporally nearby frames taken by a static camera) in order to improve accuracy. Importantly, these contextual images need not be labeled.

- When applied to a challenging wildlife detection dataset (Snapshot Serengeti), Context R-CNN with context from up to a month of images outperforms a single-frame baseline by 17.9% mAP, and outperforms S3D (a 3d convolution based baseline) by 11.2% mAP.
- Context R-CNN leverages temporal context from the unlabeled frames of a novel camera deployment to improve performance at that camera, boosting model generalizeability.

Read about Context R-CNN on the Google AI blog here.

We have provided code for generating data with associated context here, and a sample config for a Context R-CNN model here.

Snapshot Serengeti-trained Faster R-CNN and Context R-CNN models can be found in the model zoo.

A colab demonstrating Context R-CNN is provided here.

Thanks to contributors: Sara Beery, Jonathan Huang, Guanhang Wu, Vivek Rathod, Ronny Votel, Zhichao Lu, David Ross, Pietro Perona, Tanya Birch, and the Wildlife Insights AI Team.

May 19th, 2020

We have released MobileDets, a set of high-performance models for mobile CPUs, DSPs and EdgeTPUs.

- MobileDets outperform MobileNetV3+SSDLite by 1.7 mAP at comparable mobile CPU inference latencies. MobileDets also outperform MobileNetV2+SSDLite by 1.9 mAP on mobile CPUs, 3.7 mAP on EdgeTPUs and 3.4 mAP on DSPs while running equally fast. MobileDets also offer up to 2x speedup over MnasFPN on EdgeTPUs and DSPs.

For each of the three hardware platforms we have released model definition, model checkpoints trained on the COCO14 dataset and converted TFLite models in fp32 and/or uint8.

Thanks to contributors: Yunyang Xiong, Hanxiao Liu, Suyog Gupta, Berkin Akin, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Vikas Singh, Bo Chen, Quoc Le, Zhichao Lu.

May 7th, 2020

We have released a mobile model with the MnasFPN head.

- MnasFPN with MobileNet-V2 backbone is the most accurate (26.6 mAP at 183ms on Pixel 1) mobile detection model we have released to date. With depth-multiplier, MnasFPN with MobileNet-V2 backbone is 1.8 mAP higher than MobileNet-V3-Large with SSDLite (23.8 mAP vs 22.0 mAP) at similar latency (120ms) on Pixel 1.

We have released model definition, model checkpoints trained on the COCO14 dataset and a converted TFLite model.

Thanks to contributors: Bo Chen, Golnaz Ghiasi, Hanxiao Liu, Tsung-Yi Lin, Dmitry Kalenichenko, Hartwig Adam, Quoc Le, Zhichao Lu, Jonathan Huang, Hao Xu.

Nov 13th, 2019

We have released MobileNetEdgeTPU SSDLite model.

- SSDLite with MobileNetEdgeTPU backbone, which achieves 10% mAP higher than MobileNetV2 SSDLite (24.3 mAP vs 22 mAP) on a Google Pixel4 at comparable latency (6.6ms vs 6.8ms).

Along with the model definition, we are also releasing model checkpoints trained on the COCO dataset.

Thanks to contributors: Yunyang Xiong, Bo Chen, Suyog Gupta, Hanxiao Liu, Gabriel Bender, Mingxing Tan, Berkin Akin, Zhichao Lu, Quoc Le

Oct 15th, 2019

We have released two MobileNet V3 SSDLite models (presented in Searching for MobileNetV3).

- SSDLite with MobileNet-V3-Large backbone, which is 27% faster than Mobilenet V2 SSDLite (119ms vs 162ms) on a Google Pixel phone CPU at the same mAP.
- SSDLite with MobileNet-V3-Small backbone, which is 37% faster than MnasNet SSDLite reduced with depth-multiplier (43ms vs 68ms) at the same mAP.

Along with the model definition, we are also releasing model checkpoints trained on the COCO dataset.

Thanks to contributors: Bo Chen, Zhichao Lu, Vivek Rathod, Jonathan Huang

July 1st, 2019

We have released an updated set of utils and an updated tutorial for all three tracks of the Open Images Challenge 2019!

The Instance Segmentation metric for Open Images V5 and Challenge 2019 is part of this release. Check out the metric description on the Open Images website.

Thanks to contributors: Alina Kuznetsova, Rodrigo Benenson

Feb 11, 2019

We have released detection models trained on the Open Images Dataset V4 in our detection model zoo, including

- Faster R-CNN detector with Inception Resnet V2 feature extractor
- SSD detector with MobileNet V2 feature extractor
- SSD detector with ResNet 101 FPN feature extractor (aka RetinaNet-101)

Thanks to contributors: Alina Kuznetsova, Yinxiao Li

Sep 17, 2018

We have released Faster R-CNN detectors with ResNet-50 / ResNet-101 feature extractors trained on the iNaturalist Species Detection Dataset. The models are trained on the training split of the iNaturalist data for 4M iterations, they achieve 55% and 58% mean AP@.5 over 2854 classes respectively. For more details please refer to this paper.

Thanks to contributors: Chen Sun

July 13, 2018

There are many new updates in this release, extending the functionality and capability of the API:

- Moving from slim-based training to Estimator-based training.
- Support for RetinaNet, and a MobileNet adaptation of RetinaNet.
- A novel SSD-based architecture called the Pooling Pyramid Network (PPN).
- Releasing several TPU-compatible models. These can be found in the `samples/configs/` directory with a comment in the pipeline configuration files indicating TPU compatibility.
- Support for quantized training.
- Updated documentation for new binaries, Cloud training, and TensorFlow Lite.

See also our expanded announcement blogpost and accompanying tutorial at the TensorFlow blog.

Thanks to contributors: Sara Robinson, Aakanksha Chowdhery, Derek Chow, Pengchong Jin, Jonathan Huang, Vivek Rathod, Zhichao Lu, Ronny Votel

June 25, 2018

Additional evaluation tools for the Open Images Challenge 2018 are out. Check out our short tutorial on data preparation and running evaluation here!

Thanks to contributors: Alina Kuznetsova

June 5, 2018

We have released the implementation of evaluation metrics for both tracks of the Open Images Challenge 2018 as a part of the Object Detection API - see the evaluation protocols for more details. Additionally, we have released a tool for hierarchical labels expansion for the Open Images Challenge: check out oid_hierarchical_labels_expansion.py.

Thanks to contributors: Alina Kuznetsova, Vittorio Ferrari, Jasper Uijlings

April 30, 2018

We have released a Faster R-CNN detector with ResNet-101 feature extractor trained on AVA v2.1. Compared with other commonly used object detectors, it changes the action classification loss function to per-class Sigmoid loss to handle boxes with multiple labels. The model is trained on the training split of AVA v2.1 for 1.5M iterations, it achieves mean AP of 11.25% over 60 classes on the validation split of AVA v2.1. For more details please refer to this paper.

Thanks to contributors: Chen Sun, David Ross

April 2, 2018

Supercharge your mobile phones with the next generation mobile object detector! We are adding support for MobileNet V2 with SSDLite presented in MobileNetV2: Inverted Residuals and Linear Bottlenecks. This model is 35% faster than Mobilenet V1 SSD on a Google Pixel phone CPU (200ms vs. 270ms) at the same accuracy. Along with the model definition, we are also releasing a model checkpoint trained on the COCO dataset.

Thanks to contributors: Menglong Zhu, Mark Sandler, Zhichao Lu, Vivek Rathod, Jonathan Huang

February 9, 2018

We now support instance segmentation!! In this API update we support a number of instance segmentation models similar to those discussed in the Mask R-CNN paper. For further details refer to our slides from the 2017 Coco + Places Workshop. Refer to the section on Running an Instance Segmentation Model

for instructions on how to configure a model that predicts masks in addition to object bounding boxes.

Thanks to contributors: Alireza Fathi, Zhichao Lu, Vivek Rathod, Ronny Votel, Jonathan Huang

November 17, 2017

As a part of the Open Images V3 release we have released:

- An implementation of the Open Images evaluation metric and the protocol.
- Additional tools to separate inference of detection and evaluation (see this tutorial).
- A new detection model trained on the Open Images V2 data release (see Open Images model).

See more information on the Open Images website!

Thanks to contributors: Stefan Popov, Alina Kuznetsova

November 6, 2017

We have re-released faster versions of our (pre-trained) models in the model zoo. In addition to what was available before, we are also adding Faster R-CNN models trained on COCO with Inception V2 and Resnet-50 feature extractors, as well as a Faster R-CNN with Resnet-101 model trained on the KITTI dataset.

Thanks to contributors: Jonathan Huang, Vivek Rathod, Derek Chow, Tal Remez, Chen Sun.

October 31, 2017

We have released a new state-of-the-art model for object detection using the Faster-RCNN with the NASNet-A image featurization. This model achieves mAP of 43.1% on the test-dev validation dataset for COCO, improving on the best available model in the zoo by 6% in terms of absolute mAP.

Thanks to contributors: Barret Zoph, Vijay Vasudevan, Jonathon Shlens, Quoc Le

August 11, 2017

We have released an update to the Android Detect demo which will now run models trained using the TensorFlow Object Detection API on an Android device. By default, it currently runs a frozen SSD w/Mobilenet detector trained on COCO, but we encourage you to try out other detection models!

Thanks to contributors: Jonathan Huang, Andrew Harp

June 15, 2017

In addition to our base TensorFlow detection model definitions, this release includes:

- A selection of trainable detection models, including:
  - Single Shot Multibox Detector (SSD) with MobileNet,
  - SSD with Inception V2,
  - Region-Based Fully Convolutional Networks (R-FCN) with Resnet 101,
  - Faster RCNN with Resnet 101,
  - Faster RCNN with Inception Resnet v2
- Frozen weights (trained on the COCO dataset) for each of the above models to be used for out-of-the-box inference purposes.
- A Jupyter notebook for performing out-of-the-box inference with one of our released models
- Convenient training and evaluation instructions for local runs and Google Cloud.

Thanks to contributors: Jonathan Huang, Vivek Rathod, Derek Chow, Chen Sun, Menglong Zhu, Matthew Tang, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, Jasper Uijlings, Viacheslav Kovalevskyi, Kevin Murphy