YOLACT Training README

San Jose State University, Department of Computer Engineering

Author: Adam Goldstein
Project Advisor: Dr. Harry Li

03/01/2023	Created Document	Adam Goldstein

Github Source Link: https://github.com/dbolya/yolact
Github Fork: https://github.com/adkap2/yolact

Table of Contents

I. Introduction

Yolact is a simple, fully convolutional model used for real-time instance segmentation. This readme discusses the steps necessary to train Yolact on a custom dataset to best process the indoor environment. For this application, Yolact is trained with a synthetically generated indoor environment dataset. The dataset's ground truth is automatically generated by the household assets located in the unity SyntheticHome application.

Figure 1. Synthetic Indoor Environment Image

Training Yolact to work on a custom dataset requires modifications to numerous training files. Additionally, a pretrained weights file must be selected as a base to begin the training from.
The primary precision metric used in training and evaluation is Mean Average Precision (mAP). mAP takes the evaluated precision values across Intersection Over Union (IOU) thresholds ranging from 50% - 90%. This metric is measured by how closely the predicted class and annotation location is to the ground truth annotation and class.

The readme is comprised of the following parts

1. Installation Instructions
2. Configuring the model to train with a custom dataset
3. Training the model
4. Evaluation and Inference with Yolact

II. Installation

Note: My computer specifications and installation dependencies are as follows

Computer Specifications

- Intel i5 9600k
- GTX 1080 ti (11gb ram)
- 32gb memory
- 200gb ssd

Operating System

- Ubuntu 20.04

Installation Dependencies

- Python=3.70
- cudatoolkit=11.1.74
- pytorch=1.8.2=py3.7_cuda11.1_cudnn8.0.5_0
- torchvision=0.2.1
- tornado=6.2
- opencv-python==4.6.0.66
- pycocotools=2.0.5
- pyqt5=5.6.0

1. Clone the github repository

```
git clone https://github.com/dbolya/yolact.git
```

2. cd into directory

```
cd yolact
```

3. Create a new conda environment

```
conda env create -f environment.yml
conda activate yolact-env
```

Note: If dependencies in environment.yml break, try installing each dependency manually based on my personal environment

4. Download pretrained weights file such as '[yolact_base_54_800000.pth]'
5. Make new directory 'weights'

```
mkdir weights
```

6. Place weight file inside weights directory
7. Place example image inside directory and name it "input_image.png'
8. Test Yolact by evaluating it on a single image

```
# Process an image and save it to another file.
python eval.py --trained_model=weights/yolact_base_54_800000.pth
--score_threshold=0.15 --top_k=15 --image=input_image.png:output_image.png
```

III. Configuring the model

To prepare Yolact for training with a custom dataset, modifications are necessary to the ‘config.py’ file and ‘yolact.py’. Yolact must be properly configured to train on the custom object classes associated with the indoor environment.

1. Modify the config.py file
   # Note: config.py is located inside the ‘data’ directory from within the yolact repository

```
tree
├── backbone.py
├── CHANGELOG.md
├── data
│   ├── coco.py
│   ├── config.py
│   ├── grid.npy
```

   a. Open ‘config.py’

```
code config.py
```

   b. Create a new dataset derived from dataset base and name it SyntheticHome
      # Note: DATASETS begin on line 106 in the config.py file

```
SyntheticHome = dataset_base.copy({
    'name': 'SyntheticHome'
})
```

   c. Within SyntheticHome, add the file path to the train and validation images and labels

```
SyntheticHome = dataset_base.copy({
   'name': 'SyntheticHome',
   'train_images': '/path/to/training/images/',
   'train_info':   '/path/to/training/labels/',
   'valid_images': '/path/to/validation/images/',
   'valid_info':   '/path/to/validation/labels',
})
```

   d. Within SyntheticHome, add a label map offsetting the index from 0 to 1 with length equal to the number of class labels

```
SyntheticHome = dataset_base.copy({
   'name': 'SyntheticHome',
   'train_images': 'train_images': '/path/to/training/images/',
   'train_info':   '/path/to/training/labels/',
   'valid_images': '/path/to/validation/images/',
   'valid_info':   '/path/to/validation/labels',
   'label_map': {0:1, 1:2, 2:3, 3:4, 4:5, 5:6, 6:7, 7:8},
})
```

e. Within SyntheticHome, add alphabetically ordered ‘class_names’ derived from the annotation file

```
SyntheticHome = dataset_base.copy({
   'name': 'SyntheticHome',
   'train_images': 'train_images': '/path/to/training/images/',
   'train_info':   '/path/to/training/labels/',
   'valid_images': '/path/to/validation/images/',
   'valid_info':   '/path/to/validation/labels',
   'label_map': {0:1, 1:2, 2:3, 3:4, 4:5, 5:6, 6:7, 7:8},
'class_names': ("Bed", "Bench", "Chair", "Door", "Microwave", "Refrigerator", "Sofa",
"Table")
})
```

f. Under CONFIG DEFAULTS, create a new configuration object named ‘synthetic_home_config’
Note: CONFIG DEFAULTS begin around line 430

```
 synthetic_home_config = yolact_base_config.copy({
   'name': 'synthetic_home',
})
```

g. Under synthetic_home_config specify the SyntheticHome dataset and class length

```
synthetic_home_config = yolact_base_config.copy({
   'name': 'synthetic_home',
   # Dataset stuff
   'dataset': SyntheticHome,
   # Class length is + 1 to contain background class
   'num_classes': len(SyntheticHome.class_names) + 1,
})
```

h. Under synthetic_home_config, decrease the learning rate for optimal performance on custom dataset

```
synthetic_home_config = yolact_base_config.copy({
   'name': 'synthetic_home',
   # Dataset stuff
   'dataset': SyntheticHome,
   # Class length is + 1 to contain background class
   'num_classes': len(SyntheticHome.class_names) + 1,
   # Slow the learning rate
   'lr': 1e-5,
})
```

i. Modify the ‘cfg’ variable to read the ‘synthetic_home_config’ object
Note ‘cfg’ should be located around line 840

```
cfg = synthetic_home_config.copy()
```

2. Modify the yolact.py file
   # Note yolact.py is located within the base yolact directory

```
tree
├── [ 17K]  backbone.py
├── [2.9K]  CHANGELOG.md
├── [4.0K]  data
├── [ 928]  environment.yml
├── [ 46K]  eval.py
├── [4.0K]  external
├── [4.0K]  layers
├── [1.0K]  LICENSE
├── [4.0K]  logs
├── [4.0K]  __pycache__
├── [ 16K]  README.md
├── [4.0K]  results
├── [1.4K]  run_coco_eval.py
├── [4.0K]  scripts
├── [ 21K]  train.py
├── [4.0K]  utils
├── [4.0K]  web
├── [4.0K]  weights
└── [ 31K]  yolact.py
```

   a. Modify ‘pred_x’ to ‘pred_x.detach()’
      # Note: this is necessary for training on a custom dataset
      # Note: p = pred_layer(pred_x) is located around line 630

```
# p = pred_layer(pred_x)
p = pred_layer(pred_x.detach())
```

IV. Training the Model

Training the model on the custom SyntheticHomes dataset is best performed with transfer learning using the pretrained 'yolact_base_800000.pth' weights file. Additionally, this decreases the required training time from 7 days to 3 days.

1. From the base yolact directory, call 'train.py' with the following parameters

Note: the --resume parameter specifies the weights file to begin training from
Note: --start_iter=-1 sets the first iteration to be what is pre-specified in the associated weights file
Note: --batch_size=5 modifies the number of epochs required for training, the default batch size of 8 causes the small training set to terminate immediately

```
python3.7 train.py --config=synthetic_home_config
--resume=weights/yolact_base_54_800000.pth --start_iter=-1 --batch_size=5
```

2. View the training and compare the first epoch time estimation and mAP values to those below

```
[18182]  800010 || B: 1.577 | C: 10.912 | M: 2.176 | S: 5.294 | T: 19.960 || ETA: 5 days, 15:44:24
|| timer: 0.422
[18182]  800020 || B: 1.440 | C: 10.894 | M: 2.292 | S: 5.199 | T: 19.825 || ETA: 4 days, 0:28:17 ||
timer: 0.433
[18182]  800030 || B: 1.663 | C: 10.810 | M: 2.395 | S: 5.101 | T: 19.969 || ETA: 3 days, 11:38:53
|| timer: 0.421
[18182]  800040 || B: 1.699 | C: 10.742 | M: 2.466 | S: 5.000 | T: 19.907 || ETA: 3 days, 5:14:58 ||
timer: 0.420
[18182]  800050 || B: 1.716 | C: 10.654 | M: 2.509 | S: 4.926 | T: 19.806 || ETA: 3 days, 1:14:28 ||
timer: 0.421

Computing validation mAP (this may take a while)...


Calculating mAP...

       |  all  |  .50  |  .55  |  .60  |  .65  |  .70  |  .75  |  .80  |  .85  |  .90  |  .95  |
-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+
   box |  0.12 |  0.17 |  0.17 |  0.17 |  0.17 |  0.17 |  0.17 |  0.17 |  0.00 |  0.00 |  0.00 |
  mask |  0.08 |  0.17 |  0.17 |  0.17 |  0.17 |  0.17 |  0.00 |  0.00 |  0.00 |  0.00 |  0.00 |
-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+
```

3. Continue training the model until completion

V. Evaluation and Inference with Yolact

1. Evaluate the yolact model trained on the custom indoor home dataset
    a. cd into the weights file directory

```
cd weights
```

    b. Locate the most recently trained weights file

```
ls -l
```

    c. Copy the filename

```
yolact_base_20454_900000.pth
```

    d. Call eval.py to evaluate the model with the specified weights file

```
python3.7 eval.py --config=synthetic_home_config
--trained_model=weights/yolact_base_20454_900000.pth --score_threshold=0.15 --top_k=15
```

    e. View the evaluation metrics and compare the mAP values to those shown below

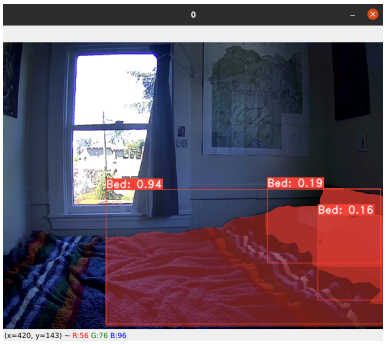
```
Processing Images [                                                  ]     55 /     55 (100.00%)     5.23 fps
Saving data...
Calculating mAP...

       |  all  |  .50  |  .55  |  .60  |  .65  |  .70  |  .75  |  .80  |  .85  |  .90  |  .95  |
-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+
   box | 29.14 | 38.92 | 38.07 | 37.57 | 35.50 | 34.62 | 30.69 | 29.18 | 24.66 | 16.42 |  5.81 |
  mask | 30.56 | 38.40 | 37.96 | 37.46 | 36.37 | 35.64 | 33.72 | 31.65 | 24.50 | 21.84 |  8.10 |
-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+-------+
```

2. Run inference from a live video camera
    a. From the yolact base directory, call

```
python3.7 eval.py --config=synthetic_home_config
--trained_model=weights/yolact_base_20454_900000.pth --score_threshold=0.15
--top_k=15 --video_multiframe=4 --video=0
```

    b. View the live videocam feed



Citation

[1] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, “YOLACT: Real-Time Instance Segmentation,” in 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.