

程序介绍

程序结构

Blob

Image8U

配置信息流

网络input output

Detect class

Detect

NMS算法

工具介绍

tensorRT

nvinfer1

caffè

gtest

protobuf

cuda c

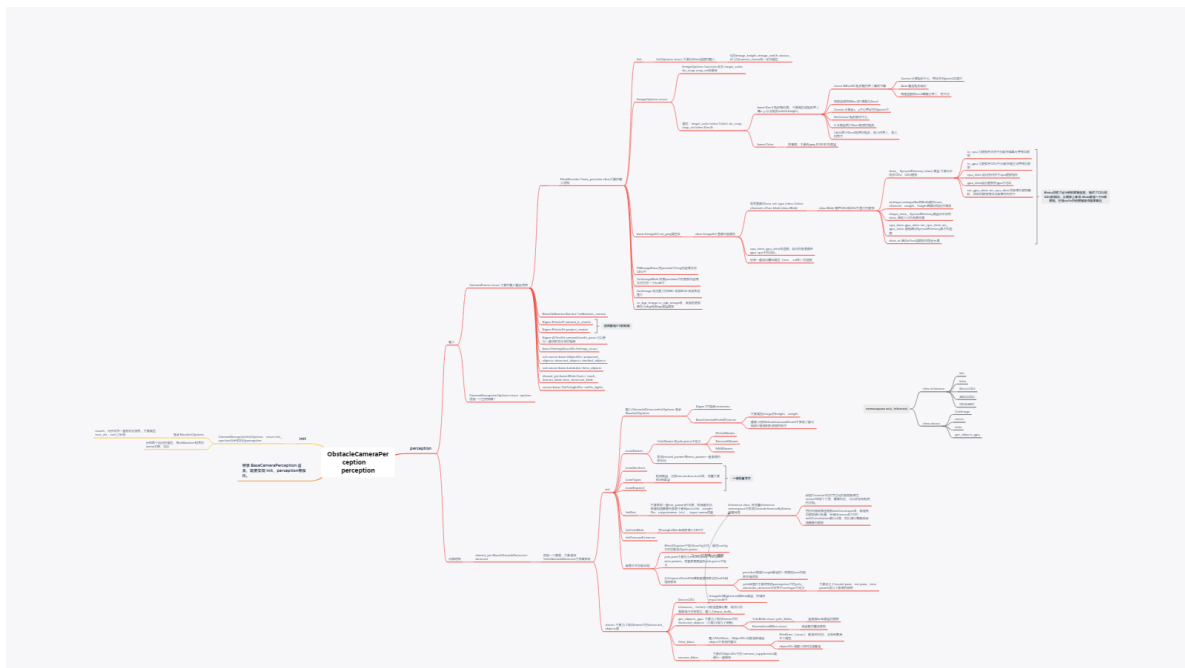
eigen

opencv

程序介绍

程序结构

我主要是阅读了camera perception相关的代码，其中包括了很多caffe相关、tensorRT相关、cuda c相关、c++线性代数库相关的知识，如需要可以找相应的材料进行学习。



Blob

其直观的可以把它看成一个有4维的结构体（包含数据和梯度），而实际上，它们只是一维的指针而已，其4维结构通过shape属性得以计算出来（根据C语言的数据顺序）。

其成员变量有

```

1  protected:
2      shared_ptr<SyncedMemory> data_;// 存放数据
3      shared_ptr<SyncedMemory> diff_;//存放梯度
4      vector<int> shape_;//存放形状
5      int count_;//数据个数
6

```

常见的成员函数有

```

1  const Dtype* cpu_data() const; //cpu使用的数据
2  void set_cpu_data(Dtype* data);//用数据块的值来blob里面的data。
3  const Dtype* gpu_data() const;//返回不可更改的指针，下同
4  const Dtype* cpu_diff() const;
5  const Dtype* gpu_diff() const;
6  Dtype* mutable_cpu_data();//返回可更改的指针，下同
7  Dtype* mutable_gpu_data();
8  Dtype* mutable_cpu_diff();
9  Dtype* mutable_gpu_diff();

```

带mutable开头的意味着可以对返回的指针内容进行更改，而不带mutable开头的返回const 指针，不能对其指针的内容进行修改，

- 其中封装了syncedmemory类，该类主要是进行了cpu跟GPU数据类型的同步
- 第一版的代码我延续时使用了Blob数据结构，主要为了兼容其他的算法。

对于具体的图像而言blob是个四维数组(num, channels, height, width)，计算
count=num×channels×height×width。
num是batch_size，channels是颜色通道数，RGB就是3；h和w分别就是图像的高和宽了。

对于全连接网络来说，blob是二维数组(num, datum)。

如果是卷积层，blob是四维数组，与图像数据相同；
如果是全连接层那么就是二维数组(num_inputs,num_outputs)。

Image8U

是对Blob类的一层封装，同时特定为image类型，不会有全连接类型，其中主要的属性有

```

1  int rows_;//图片row
2  int cols_;//图片col
3  color_type_;//rgb bgr gray三种
4  int channels_;
5  int width_step_;
6  std::shared_ptr<Blob<uint8_t>> blob_;//主要数据还是存储在blob中

```

分装了两种指针，Image8UPtr可变指针，Image8UConstPtr不可变指针。

除了blob中的xxx_data()属性之外，还有prt属性，可以指定一定的偏移。

配置信息流

- 首先全局的配置信息在
"/apollo/modules/perception/testdata/camera/app/conf/perception/camera/obstacle/obstacle.pt";中
- YoloObstacleDetector的配置信息
在/apollo/modules/perception/production/data/perception/camera/models/yolo_obstacle_detector/conf.pt中，会分为model_param、net_param、nms_param三个主要的message，定义在modules\perception\camera\lib\obstacle\detector\yolo\proto\yolo.proto中。
- 在init函数中，通过GetProtoFromFile从config file中加载具体的配置参数
- 调用initnet函数对输入输出接口进行初始化，主要初始化了inference类跟其的input与output，之后在InitYoloBlob中进行了blob的初始化。

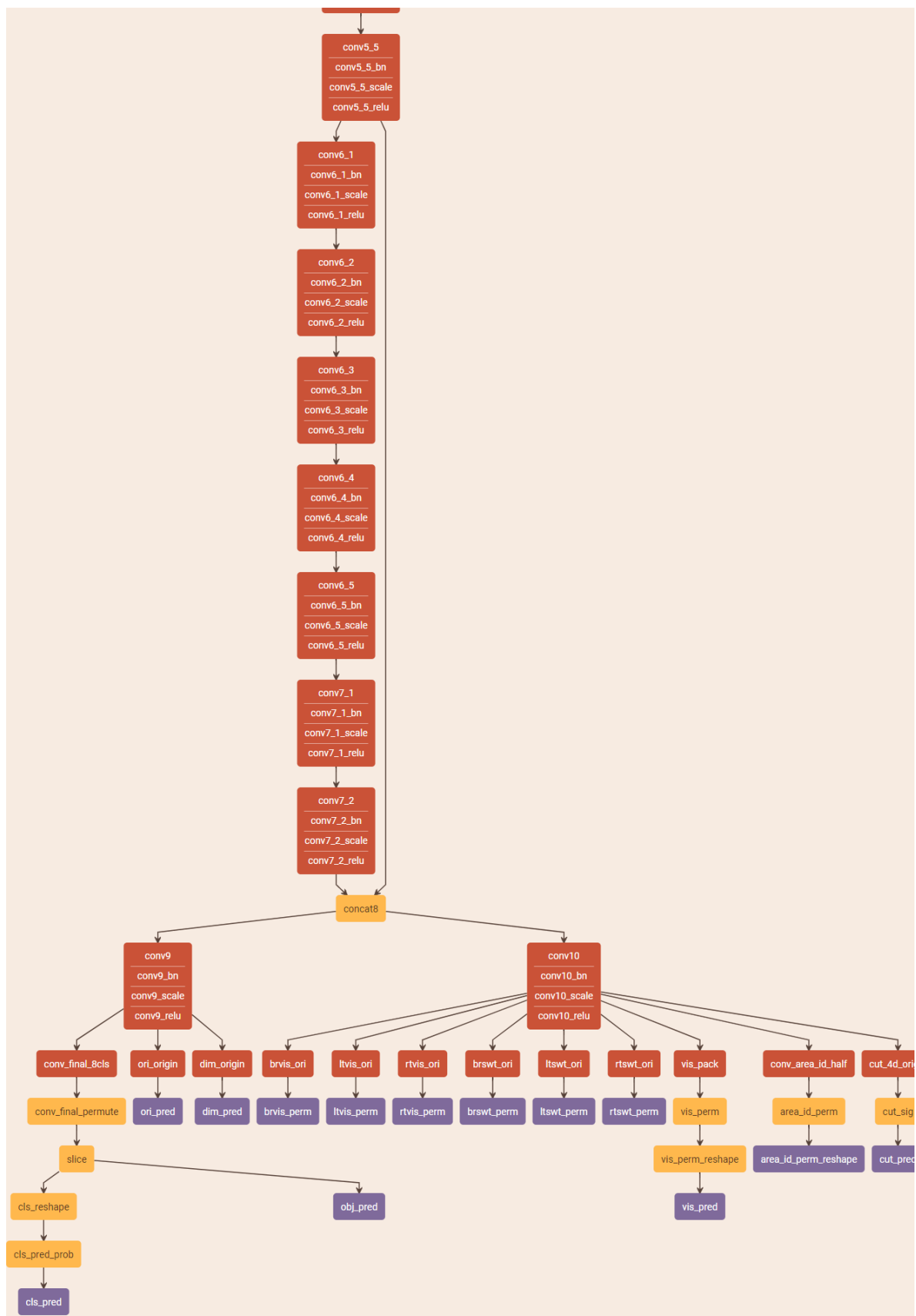
网络input output

使用[网站](#)查看caffe网络结构

apollo\modules\perception\production\data\perception\camera\models\yolo_obstacle_detector\3d-r4-half-config.pt

3d-r4-half-config





input

name	shape	describe
input	1 x 800 x 1440 x 3	image

output

name	shape
det1_loc_blob: "loc_pred"	50 x 90 x 64
det1_obj_blob: "obj_pred"	50 x 90 x 16
det1_cls_blob: "cls_pred"	4500 x 128
det1_ori_blob: "ori_pred"	50 x 90 x 32
det1_dim_blob: "dim_pred"	50 x 90 x 48
det2_loc_blob: "detect2_loc_pred"	-
det2_obj_blob: "detect2_obj_pred"	-
det2_cls_blob: "detect2_cls_pred"	-
det2_ori_conf_blob: "detect2_ori_conf_pred"	-
det2_ori_blob: "detect2_ori_pred"	-
det2_dim_blob: "detect2_dim_pred"	-
det3_loc_blob: "detect3_loc_pred"	-
det3_obj_blob: "detect3_obj_pred"	-
det3_cls_blob: "detect3_cls_pred"	-
det3_ori_conf_blob: "detect3_ori_conf_pred"	-
det3_ori_blob: "detect3_ori_pred"	-
det3_dim_blob: "detect3_dim_pred"	-
lof_blob: "lof_pred"	-
lor_blob: "lor_pred"	-
brvis_blob: "brvis_pred"	50 x 90 x 16
brswt_blob: "brswt_pred"	50 x 90 x 16
ltvis_blob: "ltvis_pred"	50 x 90 x 16
ltswt_blob: "ltswt_pred"	50 x 90 x 16
rtvis_blob: "rtvis_pred"	50 x 90 x 16
rtswt_blob: "rtvis_pred"	50 x 90 x 16
feat_blob: "conv_feat"	-
area_id_blob: "area_id_pred"	72000 x 4
visible_ratio_blob: "vis_pred"	72000 x 4
cut_off_ratio_blob: "cut_pred"	50 x 90 x 64

yolo_param

name	shape
input	1 x 800 x 1440 x 3
data_perm	1 x 3 x 800 x 1440
conv1	16 x 800 x 1440 (batch 维度接下来省略)
pool1	16 x 400 x 720
conv2	32 x 400 x 720
pool2	32 x 200 x 360
conv3_1	64 x 200 x 360
conv3_2	32 x 200 x 360
conv3_3	64 x 200 x 360
pool3	64 x 100 x 180
conv4_1	128 x 100 x 180
conv4_2	64 x 100 x 180
conv4_3	128 x 100 x 180
pool4	128 x 50 x 90
conv5_1	256 x 50 x 90
conv5_2	128 x 50 x 90
conv5_3	256 x 50 x 90
conv5_4	128 x 50 x 90
conv5_5	256 x 50 x 90
conv6_1	512 x 50 x 90
conv6_2	256 x 50 x 90
conv6_3	512 x 50 x 90
conv6_4	256 x 50 x 90
conv6_5	512 x 50 x 90
conv7_1	512 x 50 x 90
conv7_2	256 x 50 x 90
concat8	(256 + 256) x 50 x 90
conv9	512 x 50 x 90
conv10	512 x 50 x 90
conv_final_8cls	208 x 50 x 90

name	shape
conv_final_permute	50 x 90 x 208
slice	loc_pred(50 x 90 x 64), obj_perm(50 x 90 x 16), cls_perm(50 x 90 x 128)
cls_reshape	72000 x 8
cls_pred_prob	72000 x 8
cls_pred	4500 x 128
obj_pred	50 x 90 x 16
loc_pred	50 x 90 x 64
dim_origin	48 x 50 x 90
dim_pred	50 x 90 x 48
ori_origin	32 x 50 x 90
ori_pred	50 x 90 x 32
brvis_ori	16 x 50 x 90
brvis_perm	50 x 90 x 16
ltvis_ori	16 x 50 x 90
ltvis_perm	50 x 90 x 16
rtvis_ori	16 x 50 x 90
rtvis_perm	50 x 90 x 16
brswt_ori	16 x 50 x 90
brswt_perm	50 x 90 x 16
ltswt_ori	16 x 50 x 90
ltswt_perm	50 x 90 x 16
rtswt_ori	16 x 50 x 90
rtswt_perm	50 x 90 x 16
vis_pack	64 x 50 x 90
vis_perm	50 x 90 x 64
vis_perm_reshape	72000 x 4
vis_pred	72000 x 4
conv_area_id_half	64 x 50 x 90
area_id_perm	50 x 90 x 64
area_id_perm_reshape	72000 x 4

name	shape
cut_4d_origin	64 x 50 x 90
cut_sig	64 x 50 x 90
cut_pred	50 x 90 x 64

Detect class

```

1  enum class ObjectSubType {
2      UNKNOWN = 0,
3      UNKNOWN_MOVABLE = 1,
4      UNKNOWN_UNMOVABLE = 2,
5      CAR = 3,
6      VAN = 4,
7      TRUCK = 5,
8      BUS = 6,
9      CYCLIST = 7,
10     MOTORCYCLIST = 8,
11     TRICYCLIST = 9,
12     PEDESTRIAN = 10,
13     TRAFFICONE = 11,
14     MAX_OBJECT_TYPE = 12,
15 };

```

Detect

- ResizeGPU

在进行resize之前需要将camerafarmer中的data_provider中的image放到image中，image_是一个image8U类型的数据，resize的结果放到input_blob中，这是在inference中已经申请了空间的数据结构，用于infer的输入。

- Infer

通过之前配置好的网络以及input_blob中的数据，进行inference

```

1  for (auto name : output_names_) {
2      auto blob = get_blob(name);
3      if (blob != nullptr) {
4          blob->mutable_gpu_data();
5      }

```

可以看到的是，他将得到网络所有output的结果，保存在最后保存在yolo_blob_中，感觉这个inference更多的是为后面的get_object_gpu (Yolo) 服务的

- get_objects_gpu
 - 实现在modules\perception\camera\lib\obstacle\detector\yolo\region_output.cu中。
 - 其中主要是对infer的输出进行处理，输入参数说明如下所示

```

1 yolo_blobs_; //inference得到的结果
2 stream_; //cuda所用到的流
3 types_; //type类型，主要有car, per等
4 nms_; //NMSParam
5 yolo_param_.model_param(); //yolo的module param
6 light_vis_conf_threshold_; //float类型的数据，阈值
7 light_swt_conf_threshold_; //float类型的数据，阈值
8 overlapped_.get(); //bool类型的blob
9 idx_sm_.get(); //int类型的blob
10 &(frame->detected_objects); //传入的引用，最终改函数的结果保存在这。

```

- 开始是进行一些准备信息的工作，提取出yolo_blob_中的信息，判断lof、lor等信息，其中还有一些具体的维度计算没有跟踪。
- get_object_kernel函数，猜测可能是检测用的网络用的函数，没太看明白中间在干啥。
- 对于每一种物体类别，调用apply_nms_gpu()函数去除多余的检测框。两个apply_nms_gpu，第一个在for之外，其实是对最大的confidence的使用NMS算法，在for之中的是分别对剩下的使用NMS算法，最终的结果保存在indices中。
- 最后对剩下的框，填充物体的种类跟score，最后的fill box相关就是将obj->camera_supplement.box.xmin这些相关的信息填入cameraframe中。

对于每一种物体类别，调用apply_nms_gpu()函数去除多余的检测框。这个函数首先用阈值过滤下，把confidence小于0.8的干掉，剩下的框的index和confidence放在idx和confidence两个vector中。然后把剩下的元素按confidence排个序。接着调用compute_overlapped_by_idx_gpu()函数计算这些框间的重叠关系。当两个框间的IoU（即Jaccard overlap）大于0.4时，算两者重叠。基于这个结果，就可以调用apply_nms()执行NMS算法。结果放在indices这个成员中。indices是类别到物体框index数组的映射。

• filter_bbox

这就是将不合理的框筛选出去，如左上角的点的坐标大于右上坐标的点。

• feature_extractor_->Extract

根据初始化中使用的是“TrackingFeatureExtractor”，所以调用的是modules\perception\camera\lib\feature_extractor\tfe\tracking_feat_extractor.cc中的extract。

这里是使用输入的box的位置信息，得到该box表示的是什么样的东西（如person等），主要注意的是这里也有一个网络。

```

1 float *rois_data =
2     feature_extractor_layer_ptr->rois_blob->mutable_cpu_data();
3 for (const auto &obj : frame->detected_objects) {
4     rois_data[0] = 0;
5     rois_data[1] =
6         obj->camera_supplement.box.xmin * static_cast<float>
7         (feat_width_);
8     rois_data[2] =
9         obj->camera_supplement.box.ymin * static_cast<float>
10        (feat_height_);
11    rois_data[3] =
12        obj->camera_supplement.box.xmax * static_cast<float>
13        (feat_width_);
14    rois_data[4] =
15        obj->camera_supplement.box.ymax * static_cast<float>
16        (feat_height_);
17    rois_data[5] = obj->score;
18    rois_data[6] = obj->label;
19    rois_data[7] = obj->score;
20    rois_data[8] = obj->label;
21    rois_data[9] = obj->score;
22    rois_data[10] = obj->label;
23    rois_data[11] = obj->score;
24    rois_data[12] = obj->label;
25    rois_data[13] = obj->score;
26    rois_data[14] = obj->label;
27    rois_data[15] = obj->score;
28    rois_data[16] = obj->label;
29    rois_data[17] = obj->score;
30    rois_data[18] = obj->label;
31    rois_data[19] = obj->score;
32    rois_data[20] = obj->label;
33    rois_data[21] = obj->score;
34    rois_data[22] = obj->label;
35    rois_data[23] = obj->score;
36    rois_data[24] = obj->label;
37    rois_data[25] = obj->score;
38    rois_data[26] = obj->label;
39    rois_data[27] = obj->score;
40    rois_data[28] = obj->label;
41    rois_data[29] = obj->score;
42    rois_data[30] = obj->label;
43    rois_data[31] = obj->score;
44    rois_data[32] = obj->label;
45    rois_data[33] = obj->score;
46    rois_data[34] = obj->label;
47    rois_data[35] = obj->score;
48    rois_data[36] = obj->label;
49    rois_data[37] = obj->score;
50    rois_data[38] = obj->label;
51    rois_data[39] = obj->score;
52    rois_data[40] = obj->label;
53    rois_data[41] = obj->score;
54    rois_data[42] = obj->label;
55    rois_data[43] = obj->score;
56    rois_data[44] = obj->label;
57    rois_data[45] = obj->score;
58    rois_data[46] = obj->label;
59    rois_data[47] = obj->score;
60    rois_data[48] = obj->label;
61    rois_data[49] = obj->score;
62    rois_data[50] = obj->label;
63    rois_data[51] = obj->score;
64    rois_data[52] = obj->label;
65    rois_data[53] = obj->score;
66    rois_data[54] = obj->label;
67    rois_data[55] = obj->score;
68    rois_data[56] = obj->label;
69    rois_data[57] = obj->score;
70    rois_data[58] = obj->label;
71    rois_data[59] = obj->score;
72    rois_data[60] = obj->label;
73    rois_data[61] = obj->score;
74    rois_data[62] = obj->label;
75    rois_data[63] = obj->score;
76    rois_data[64] = obj->label;
77    rois_data[65] = obj->score;
78    rois_data[66] = obj->label;
79    rois_data[67] = obj->score;
80    rois_data[68] = obj->label;
81    rois_data[69] = obj->score;
82    rois_data[70] = obj->label;
83    rois_data[71] = obj->score;
84    rois_data[72] = obj->label;
85    rois_data[73] = obj->score;
86    rois_data[74] = obj->label;
87    rois_data[75] = obj->score;
88    rois_data[76] = obj->label;
89    rois_data[77] = obj->score;
90    rois_data[78] = obj->label;
91    rois_data[79] = obj->score;
92    rois_data[80] = obj->label;
93    rois_data[81] = obj->score;
94    rois_data[82] = obj->label;
95    rois_data[83] = obj->score;
96    rois_data[84] = obj->label;
97    rois_data[85] = obj->score;
98    rois_data[86] = obj->label;
99    rois_data[87] = obj->score;
100   rois_data[88] = obj->label;
101   rois_data[89] = obj->score;
102   rois_data[90] = obj->label;
103   rois_data[91] = obj->score;
104   rois_data[92] = obj->label;
105   rois_data[93] = obj->score;
106   rois_data[94] = obj->label;
107   rois_data[95] = obj->score;
108   rois_data[96] = obj->label;
109   rois_data[97] = obj->score;
110   rois_data[98] = obj->label;
111   rois_data[99] = obj->score;
112   rois_data[100] = obj->label;
113   rois_data[101] = obj->score;
114   rois_data[102] = obj->label;
115   rois_data[103] = obj->score;
116   rois_data[104] = obj->label;
117   rois_data[105] = obj->score;
118   rois_data[106] = obj->label;
119   rois_data[107] = obj->score;
120   rois_data[108] = obj->label;
121   rois_data[109] = obj->score;
122   rois_data[110] = obj->label;
123   rois_data[111] = obj->score;
124   rois_data[112] = obj->label;
125   rois_data[113] = obj->score;
126   rois_data[114] = obj->label;
127   rois_data[115] = obj->score;
128   rois_data[116] = obj->label;
129   rois_data[117] = obj->score;
130   rois_data[118] = obj->label;
131   rois_data[119] = obj->score;
132   rois_data[120] = obj->label;
133   rois_data[121] = obj->score;
134   rois_data[122] = obj->label;
135   rois_data[123] = obj->score;
136   rois_data[124] = obj->label;
137   rois_data[125] = obj->score;
138   rois_data[126] = obj->label;
139   rois_data[127] = obj->score;
140   rois_data[128] = obj->label;
141   rois_data[129] = obj->score;
142   rois_data[130] = obj->label;
143   rois_data[131] = obj->score;
144   rois_data[132] = obj->label;
145   rois_data[133] = obj->score;
146   rois_data[134] = obj->label;
147   rois_data[135] = obj->score;
148   rois_data[136] = obj->label;
149   rois_data[137] = obj->score;
150   rois_data[138] = obj->label;
151   rois_data[139] = obj->score;
152   rois_data[140] = obj->label;
153   rois_data[141] = obj->score;
154   rois_data[142] = obj->label;
155   rois_data[143] = obj->score;
156   rois_data[144] = obj->label;
157   rois_data[145] = obj->score;
158   rois_data[146] = obj->label;
159   rois_data[147] = obj->score;
160   rois_data[148] = obj->label;
161   rois_data[149] = obj->score;
162   rois_data[150] = obj->label;
163   rois_data[151] = obj->score;
164   rois_data[152] = obj->label;
165   rois_data[153] = obj->score;
166   rois_data[154] = obj->label;
167   rois_data[155] = obj->score;
168   rois_data[156] = obj->label;
169   rois_data[157] = obj->score;
170   rois_data[158] = obj->label;
171   rois_data[159] = obj->score;
172   rois_data[160] = obj->label;
173   rois_data[161] = obj->score;
174   rois_data[162] = obj->label;
175   rois_data[163] = obj->score;
176   rois_data[164] = obj->label;
177   rois_data[165] = obj->score;
178   rois_data[166] = obj->label;
179   rois_data[167] = obj->score;
180   rois_data[168] = obj->label;
181   rois_data[169] = obj->score;
182   rois_data[170] = obj->label;
183   rois_data[171] = obj->score;
184   rois_data[172] = obj->label;
185   rois_data[173] = obj->score;
186   rois_data[174] = obj->label;
187   rois_data[175] = obj->score;
188   rois_data[176] = obj->label;
189   rois_data[177] = obj->score;
190   rois_data[178] = obj->label;
191   rois_data[179] = obj->score;
192   rois_data[180] = obj->label;
193   rois_data[181] = obj->score;
194   rois_data[182] = obj->label;
195   rois_data[183] = obj->score;
196   rois_data[184] = obj->label;
197   rois_data[185] = obj->score;
198   rois_data[186] = obj->label;
199   rois_data[187] = obj->score;
200   rois_data[188] = obj->label;
201   rois_data[189] = obj->score;
202   rois_data[190] = obj->label;
203   rois_data[191] = obj->score;
204   rois_data[192] = obj->label;
205   rois_data[193] = obj->score;
206   rois_data[194] = obj->label;
207   rois_data[195] = obj->score;
208   rois_data[196] = obj->label;
209   rois_data[197] = obj->score;
210   rois_data[198] = obj->label;
211   rois_data[199] = obj->score;
212   rois_data[200] = obj->label;
213   rois_data[201] = obj->score;
214   rois_data[202] = obj->label;
215   rois_data[203] = obj->score;
216   rois_data[204] = obj->label;
217   rois_data[205] = obj->score;
218   rois_data[206] = obj->label;
219   rois_data[207] = obj->score;
220   rois_data[208] = obj->label;
221   rois_data[209] = obj->score;
222   rois_data[210] = obj->label;
223   rois_data[211] = obj->score;
224   rois_data[212] = obj->label;
225   rois_data[213] = obj->score;
226   rois_data[214] = obj->label;
227   rois_data[215] = obj->score;
228   rois_data[216] = obj->label;
229   rois_data[217] = obj->score;
230   rois_data[218] = obj->label;
231   rois_data[219] = obj->score;
232   rois_data[220] = obj->label;
233   rois_data[221] = obj->score;
234   rois_data[222] = obj->label;
235   rois_data[223] = obj->score;
236   rois_data[224] = obj->label;
237   rois_data[225] = obj->score;
238   rois_data[226] = obj->label;
239   rois_data[227] = obj->score;
240   rois_data[228] = obj->label;
241   rois_data[229] = obj->score;
242   rois_data[230] = obj->label;
243   rois_data[231] = obj->score;
244   rois_data[232] = obj->label;
245   rois_data[233] = obj->score;
246   rois_data[234] = obj->label;
247   rois_data[235] = obj->score;
248   rois_data[236] = obj->label;
249   rois_data[237] = obj->score;
250   rois_data[238] = obj->label;
251   rois_data[239] = obj->score;
252   rois_data[240] = obj->label;
253   rois_data[241] = obj->score;
254   rois_data[242] = obj->label;
255   rois_data[243] = obj->score;
256   rois_data[244] = obj->label;
257   rois_data[245] = obj->score;
258   rois_data[246] = obj->label;
259   rois_data[247] = obj->score;
260   rois_data[248] = obj->label;
261   rois_data[249] = obj->score;
262   rois_data[250] = obj->label;
263   rois_data[251] = obj->score;
264   rois_data[252] = obj->label;
265   rois_data[253] = obj->score;
266   rois_data[254] = obj->label;
267   rois_data[255] = obj->score;
268   rois_data[256] = obj->label;
269   rois_data[257] = obj->score;
270   rois_data[258] = obj->label;
271   rois_data[259] = obj->score;
272   rois_data[260] = obj->label;
273   rois_data[261] = obj->score;
274   rois_data[262] = obj->label;
275   rois_data[263] = obj->score;
276   rois_data[264] = obj->label;
277   rois_data[265] = obj->score;
278   rois_data[266] = obj->label;
279   rois_data[267] = obj->score;
280   rois_data[268] = obj->label;
281   rois_data[269] = obj->score;
282   rois_data[270] = obj->label;
283   rois_data[271] = obj->score;
284   rois_data[272] = obj->label;
285   rois_data[273] = obj->score;
286   rois_data[274] = obj->label;
287   rois_data[275] = obj->score;
288   rois_data[276] = obj->label;
289   rois_data[277] = obj->score;
290   rois_data[278] = obj->label;
291   rois_data[279] = obj->score;
292   rois_data[280] = obj->label;
293   rois_data[281] = obj->score;
294   rois_data[282] = obj->label;
295   rois_data[283] = obj->score;
296   rois_data[284] = obj->label;
297   rois_data[285] = obj->score;
298   rois_data[286] = obj->label;
299   rois_data[287] = obj->score;
300   rois_data[288] = obj->label;
301   rois_data[289] = obj->score;
302   rois_data[290] = obj->label;
303   rois_data[291] = obj->score;
304   rois_data[292] = obj->label;
305   rois_data[293] = obj->score;
306   rois_data[294] = obj->label;
307   rois_data[295] = obj->score;
308   rois_data[296] = obj->label;
309   rois_data[297] = obj->score;
310   rois_data[298] = obj->label;
311   rois_data[299] = obj->score;
312   rois_data[300] = obj->label;
313   rois_data[301] = obj->score;
314   rois_data[302] = obj->label;
315   rois_data[303] = obj->score;
316   rois_data[304] = obj->label;
317   rois_data[305] = obj->score;
318   rois_data[306] = obj->label;
319   rois_data[307] = obj->score;
320   rois_data[308] = obj->label;
321   rois_data[309] = obj->score;
322   rois_data[310] = obj->label;
323   rois_data[311] = obj->score;
324   rois_data[312] = obj->label;
325   rois_data[313] = obj->score;
326   rois_data[314] = obj->label;
327   rois_data[315] = obj->score;
328   rois_data[316] = obj->label;
329   rois_data[317] = obj->score;
330   rois_data[318] = obj->label;
331   rois_data[319] = obj->score;
332   rois_data[320] = obj->label;
333   rois_data[321] = obj->score;
334   rois_data[322] = obj->label;
335   rois_data[323] = obj->score;
336   rois_data[324] = obj->label;
337   rois_data[325] = obj->score;
338   rois_data[326] = obj->label;
339   rois_data[327] = obj->score;
340   rois_data[328] = obj->label;
341   rois_data[329] = obj->score;
342   rois_data[330] = obj->label;
343   rois_data[331] = obj->score;
344   rois_data[332] = obj->label;
345   rois_data[333] = obj->score;
346   rois_data[334] = obj->label;
347   rois_data[335] = obj->score;
348   rois_data[336] = obj->label;
349   rois_data[337] = obj->score;
350   rois_data[338] = obj->label;
351   rois_data[339] = obj->score;
352   rois_data[340] = obj->label;
353   rois_data[341] = obj->score;
354   rois_data[342] = obj->label;
355   rois_data[343] = obj->score;
356   rois_data[344] = obj->label;
357   rois_data[345] = obj->score;
358   rois_data[346] = obj->label;
359   rois_data[347] = obj->score;
360   rois_data[348] = obj->label;
361   rois_data[349] = obj->score;
362   rois_data[350] = obj->label;
363   rois_data[351] = obj->score;
364   rois_data[352] = obj->label;
365   rois_data[353] = obj->score;
366   rois_data[354] = obj->label;
367   rois_data[355] = obj->score;
368   rois_data[356] = obj->label;
369   rois_data[357] = obj->score;
370   rois_data[358] = obj->label;
371   rois_data[359] = obj->score;
372   rois_data[360] = obj->label;
373   rois_data[361] = obj->score;
374   rois_data[362] = obj->label;
375   rois_data[363] = obj->score;
376   rois_data[364] = obj->label;
377   rois_data[365] = obj->score;
378   rois_data[366] = obj->label;
379   rois_data[367] = obj->score;
380   rois_data[368] = obj->label;
381   rois_data[369] = obj->score;
382   rois_data[370] = obj->label;
383   rois_data[371] = obj->score;
384   rois_data[372] = obj->label;
385   rois_data[373] = obj->score;
386   rois_data[374] = obj->label;
387   rois_data[375] = obj->score;
388   rois_data[376] = obj->label;
389   rois_data[377] = obj->score;
390   rois_data[378] = obj->label;
391   rois_data[379] = obj->score;
392   rois_data[380] = obj->label;
393   rois_data[381] = obj->score;
394   rois_data[382] = obj->label;
395   rois_data[383] = obj->score;
396   rois_data[384] = obj->label;
397   rois_data[385] = obj->score;
398   rois_data[386] = obj->label;
399   rois_data[387] = obj->score;
400   rois_data[388] = obj->label;
401   rois_data[389] = obj->score;
402   rois_data[390] = obj->label;
403   rois_data[391] = obj->score;
404   rois_data[392] = obj->label;
405   rois_data[393] = obj->score;
406   rois_data[394] = obj->label;
407   rois_data[395] = obj->score;
408   rois_data[396] = obj->label;
409   rois_data[397] = obj->score;
410   rois_data[398] = obj->label;
411   rois_data[399] = obj->score;
412   rois_data[400] = obj->label;
413   rois_data[401] = obj->score;
414   rois_data[402] = obj->label;
415   rois_data[403] = obj->score;
416   rois_data[404] = obj->label;
417   rois_data[405] = obj->score;
418   rois_data[406] = obj->label;
419   rois_data[407] = obj->score;
420   rois_data[408] = obj->label;
421   rois_data[409] = obj->score;
422   rois_data[410] = obj->label;
423   rois_data[411] = obj->score;
424   rois_data[412] = obj->label;
425   rois_data[413] = obj->score;
426   rois_data[414] = obj->label;
427   rois_data[415] = obj->score;
428   rois_data[416] = obj->label;
429   rois_data[417] = obj->score;
430   rois_data[418] = obj->label;
431   rois_data[419] = obj->score;
432   rois_data[420] = obj->label;
433   rois_data[421] = obj->score;
434   rois_data[422] = obj->label;
435   rois_data[423] = obj->score;
436   rois_data[424] = obj->label;
437   rois_data[425] = obj->score;
438   rois_data[426] = obj->label;
439   rois_data[427] = obj->score;
440   rois_data[428] = obj->label;
441   rois_data[429] = obj->score;
442   rois_data[430] = obj->label;
443   rois_data[431] = obj->score;
444   rois_data[432] = obj->label;
445   rois_data[433] = obj->score;
446   rois_data[434] = obj->label;
447   rois_data[435] = obj->score;
448   rois_data[436] = obj->label;
449   rois_data[437] = obj->score;
450   rois_data[438] = obj->label;
451   rois_data[439] = obj->score;
452   rois_data[440] = obj->label;
453   rois_data[441] = obj->score;
454   rois_data[442] = obj->label;
455   rois_data[443] = obj->score;
456   rois_data[444] = obj->label;
457   rois_data[445] = obj->score;
458   rois_data[446] = obj->label;
459   rois_data[447] = obj->score;
460   rois_data[448] = obj->label;
461   rois_data[449] = obj->score;
462   rois_data[450] = obj->label;
463   rois_data[451] = obj->score;
464   rois_data[452] = obj->label;
465   rois_data[453] = obj->score;
466   rois_data[454] = obj->label;
467   rois_data[455] = obj->score;
468   rois_data[456] = obj->label;
469   rois_data[457] = obj->score;
470   rois_data[458] = obj->label;
471   rois_data[459] = obj->score;
472   rois_data[460] = obj->label;
473   rois_data[461] = obj->score;
474   rois_data[462] = obj->label;
475   rois_data[463] = obj->score;
476   rois_data[464] = obj->label;
477   rois_data[465] = obj->score;
478   rois_data[466] = obj->label;
479   rois_data[467] = obj->score;
480   rois_data[468] = obj->label;
481   rois_data[469] = obj->score;
482   rois_data[470] = obj->label;
483   rois_data[471] = obj->score;
484   rois_data[472] = obj->label;
485   rois_data[473] = obj->score;
486   rois_data[474] = obj->label;
487   rois_data[475] = obj->score;
488   rois_data[476] = obj->label;
489   rois_data[477] = obj->score;
490   rois_data[478] = obj->label;
491   rois_data[479] = obj->score;
492   rois_data[480] = obj->label;
493   rois_data[481] = obj->score;
494   rois_data[482] = obj->label;
495   rois_data[483] = obj->score;
496   rois_data[484] = obj->label;
497   rois_data[485] = obj->score;
498   rois_data[486] = obj->label;
499   rois_data[487] = obj->score;
500   rois_data[488] = obj->label;
501   rois_data[489] = obj->score;
502   rois_data[490] = obj->label;
503   rois_data[491] = obj->score;
504   rois_data[492] = obj->label;
505   rois_data[493] = obj->score;
506   rois_data[494] = obj->label;
507   rois_data[495] = obj->score;
508   rois_data[496] = obj->label;
509   rois_data[497] = obj->score;
510   rois_data[498] = obj->label;
511   rois_data[499] = obj->score;
512   rois_data[500] = obj->label;
513   rois_data[501] = obj->score;
514   rois_data[502] = obj->label;
515   rois_data[503] = obj->score;
516   rois_data[504] = obj->label;
517   rois_data[505] = obj->score;
518   rois_data[506] = obj->label;
519   rois_data[507] = obj->score;
520   rois_data[508] = obj->label;
521   rois_data[509] = obj->score;
522   rois_data[510] = obj->label;
523   rois_data[511] = obj->score;
524   rois_data[512] = obj->label;
525   rois_data[513] = obj->score;
526   rois_data[514] = obj->label;
527   rois_data[515] = obj->score;
528   rois_data[516] = obj->label;
529   rois_data[517] = obj->score;
530   rois_data[518] = obj->label;
531   rois_data[519] = obj->score;
532   rois_data[520] = obj->label;
533   rois_data[521] = obj->score;
534   rois_data[522] = obj->label;
535   rois_data[523] = obj->score;
536   rois_data[524] = obj->label;
537   rois_data[525] = obj->score;
538   rois_data[526] = obj->label;
539   rois_data[527] = obj->score;
540   rois_data[528] = obj->label;
541   rois_data[529] = obj->score;
542   rois_data[530] = obj->label;
543   rois_data[531] = obj->score;
544   rois_data[532] = obj->label;
545   rois_data[533] = obj->score;
546   rois_data[534] = obj->label;
547   rois_data[535] = obj->score;
548   rois_data[536] = obj->label;
549   rois_data[537] = obj->score;
550   rois_data[538] = obj->label;
551   rois_data[539] = obj->score;
552   rois_data[540] = obj->label;
553   rois_data[541] = obj->score;
554   rois_data[542] = obj->label;
555   rois_data[543] = obj->score;
556   rois_data[544] = obj->label;
557   rois_data[545] = obj->score;
558   rois_data[546] = obj->label;
559   rois_data[547] = obj->score;
560   rois_data[548] = obj->label;
561   rois_data[549] = obj->score;
562   rois_data[550] = obj->label;
563   rois_data[551] = obj->score;
564   rois_data[552] = obj->label;
565   rois_data[553] = obj->score;
566   rois_data[554] = obj->label;
567   rois_data[555] = obj->score;
568   rois_data[556] = obj->label;
569   rois_data[557] = obj->score;
570   rois_data[558] = obj->label;
571   rois_data[559] = obj->score;
572   rois_data[560] = obj->label;
573   rois_data[561] = obj->score;
574   rois_data[562] = obj->label;
575   rois_data[563] = obj->score;
576   rois_data[564] = obj->label;
577   rois_data[565] = obj->score;
578   rois_data[566] = obj->label;
579   rois_data[567] = obj->score;
580   rois_data[568] = obj->label;
581   rois_data[569] = obj->score;
582   rois_data[570] = obj->label;
583   rois_data[571] = obj->score;
584   rois_data[572] = obj->label;
585   rois_data[573] = obj->score;
586   rois_data[574] = obj->label;
587   rois_data[575] = obj->score;
588   rois_data[576] = obj->label;
589   rois_data[577] = obj->score;
590   rois_data[578] = obj->label;
591   rois_data[
```

```

12     obj->camera_supplement.box.ymax * static_cast<float>
    (feat_height_);
13     ADEBUG << rois_data[0] << " " << rois_data[1] << " " <<
    rois_data[2]
14         << " " << rois_data[3] << " " << rois_data[4];
15     rois_data += feature_extractor_layer_ptr->rois_blob->offset(1);
16 }
17 feature_extractor_layer_ptr->pooling_layer->ForwardGPU(
18     {feat_blob_, feature_extractor_layer_ptr->rois_blob},
19     {frame->track_feature_blob});

```

- recover_bbox

这个函数主要做的是一些与坐标转换相关的处理，upper_left/uppper_right中填的是相对于ROI（图片下面1920 x 768）中并且归一化到[0, 1]范围内的。比如xmin/ymin/xmax/ymax为(0.552336, 0.27967, 0.583794, 0.344488)。这个坐标会转换到ROI中的像素坐标。转换后x/y/w/h为(1060, 526, 60, 49)。之后还会和图像的大小做一个并，把那些超出图像的部分切掉。最后如果框的边界是在图像的边上的，会设置VisualObject的trunc_width/trunc_height成员。

最终可以看到detect中各项函数使用的时间如下所示。

name	time(ms)	describe
start	0.003	get input blob
GetImageBlob	0.014	from provider to input blob
ResizeGPU	0.014	resize
Infer	7.367	inference
get_objects_gpu	4.184	get objects
filter_bbox	0.042	filter not create box
Extract	0.115	Extract features
recover_bbox	0.014	

可以通过perception的tool的offline中的offline_obstacle_pipeline.cc进行离线的算法测试，最终生成测试图片如下所示。



最终的object检测的结果其实会保存在输入的frame.detected_objects.camera_supplement中，其中画出的box主要是在box中，这里的主要是2D的矩形，许在真实的系统中其实还进行了3D重建。

name	describe
xmin, ymin	矩形的左上角点
xmax, ymax	矩形的右上角的点

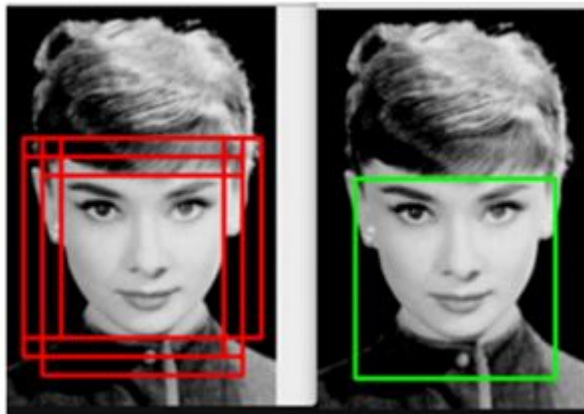
给出一该图片的检测输出

name	value
car	547 670 800 857 0.949245
car	1213 633 1789 988 0.934825
car	185 553 636 1024 0.89498
car	851 682 929 741 0.88613
car	905 685 948 727 0.605641
car	1010 685 1034 710 0.528782
pedestrian	1751 573 1852 875 0.951115

NMS算法

根据候选框的类别分类概率做排序: $A < B < C < D < E < F$

1. 先标记最大概率F
2. 将F与A-E进行IOU，设定阈值，要是小的舍掉，假设B，D舍掉
3. 在A C E中选取最大的E重复上述步骤



工具介绍

tensorRT

TensorRT是NVIDIA推出的一个高性能的深度学习推理框架，可以让深度学习模型在NVIDIA [GPU](#)上实现低延迟，高吞吐量的部署。TensorRT支持Caffe, TensorFlow, Mxnet, Pytorch等主流深度学习框架。TensorRT是一个C++库，并且提供了C++ API和Python API，主要在NVIDIA GPU进行高性能的推理(Inference)加速。

程序在运行的过程中会出现INT8 mode not support的字样，这是因为2070super不支持一个叫做 **DP4A** 的指令集优化，换上支持此指令集的GPU如2080TI会得到3-5倍的加速

tensorRT的相关知识点比较多，我主要参看了[这篇博客](#)，如果需要将默认的yolo网络修改为自己想要的网络，需要对tensorRT有一定的认识

nvinfer1

这是tensorRT中的一个namespace，他比较重要的地方是实现了网络中的不同层的GPU加速，在apollo中主要的优化就是争对apollo这个系统所特有的数据格式进行了数据处理，计算函数还是使用的tensorRT自带的计算函数，具体函数的使用可以查看[官方文档](#)，文档中给了具体函数的使用接口，以及功能介绍。

caffe

相对于工程性的项目，caffe的使用感觉是比tensorflow这些会方便一点，特别时融合了很多其他框架的时候，caffe的优势就体现出来了，caffe完全以配置信息的形势呈现，修改对应的网络模型只需要修改对应的配置文件，不需要重新对代码进行修改或者编译，我学习过程主要参考了这个人的[博客](#)，以及对源码进行了一定额阅读，其实在apollo的算法模块很多都是caffe的源码，需要的话可以进行阅读。

gtest

google开源测试工具，可以很方便的争对你开发的每一个函数进行快速的测试，具体可以查看[官方教程](#)

protobuf

google开源的类似json或者xml，是一种轻便高效的结构化数据存储格式，可以用于结构化数据串行化，或者说序列化。具体可查看[官方教程](#)，以及一些不错的[博客](#)。

cuda c

为了调用GPU进行运算，需要用到cuda c给出的API，Apollo也是使用cuda c这样的框架进行的异构设备的管理。

eigen

Eigen 是C++语言里的一个开源模版库，支持线性代数运算，矩阵和矢量运算，数值分析及其相关的算法。在Apollo中需要用这个库中的矩阵操作，包括一些矩阵乘法，放射变换等。在cameraframe类中有挺多。

opencv

会使用到cv中一些图像处理的函数，比如resize中的线性插值等。