29 Dec 2016

SSD: Single Shot MultiBox Detector

Wei Liu[1], Dragomir Anguelov[2], Dumitru Erhan[3], Christian Szegedy[3],
Scott Reed[4], Cheng-Yang Fu[1], Alexander C. Berg[1]

[1]UNC Chapel Hill [2]Zoox Inc. [3]Google Inc. [4]University of Michigan, Ann-Arbor
[1]wliu@cs.unc.edu, [2]drago@zoox.com, [3]{dumitru,szegedy}@google.com,
[4]reedscot@umich.edu, [1]{cyfu,aberg}@cs.unc.edu

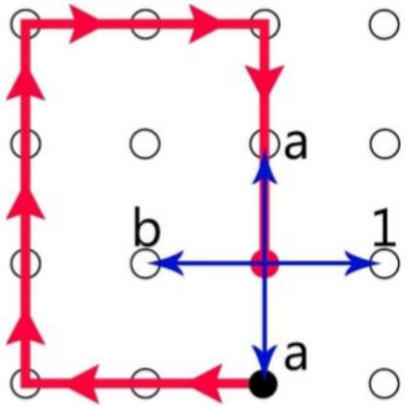
Ssd: **Single shot** multibox **detector**

W Liu, D Anguelov, D Erhan, C Szegedy... - European conference on ..., 2016 - Springer

We present a method for detecting objects in images using a **single** deep neural network. Our approach, named **SSD**, discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location. At prediction ...

Alexander Lobashev

Reinforced random walk



Обобщенная модель (2-D).

- $P(1) \rightarrow \frac{1}{1+b+2a}$
- $P(a) \rightarrow \frac{2a}{1+b+2a}$
- $P(b) \rightarrow \frac{b}{1+b+2a}$

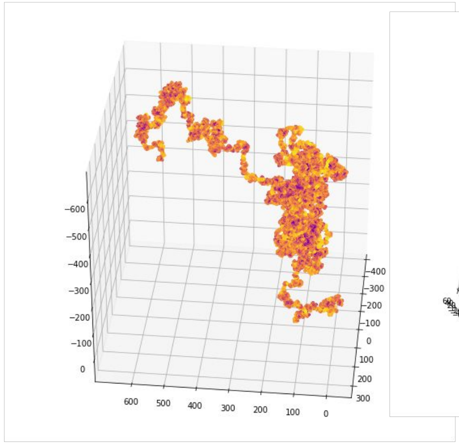
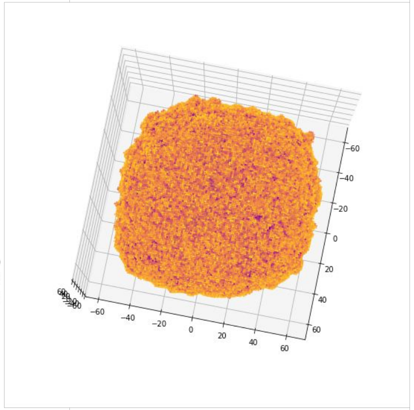
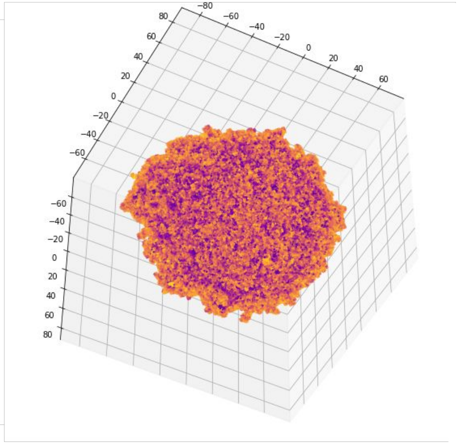
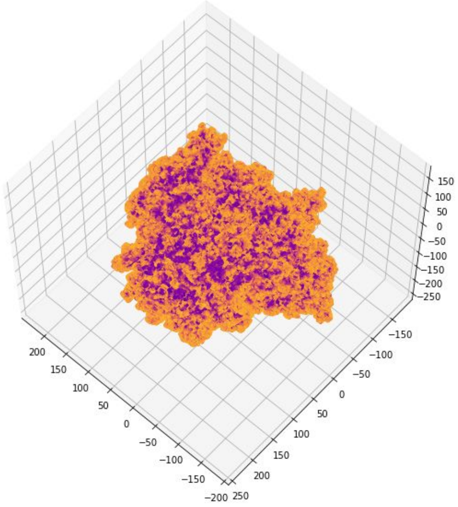
Model has two macroscopic parameters:

a - interaction with **volume** of visited domain

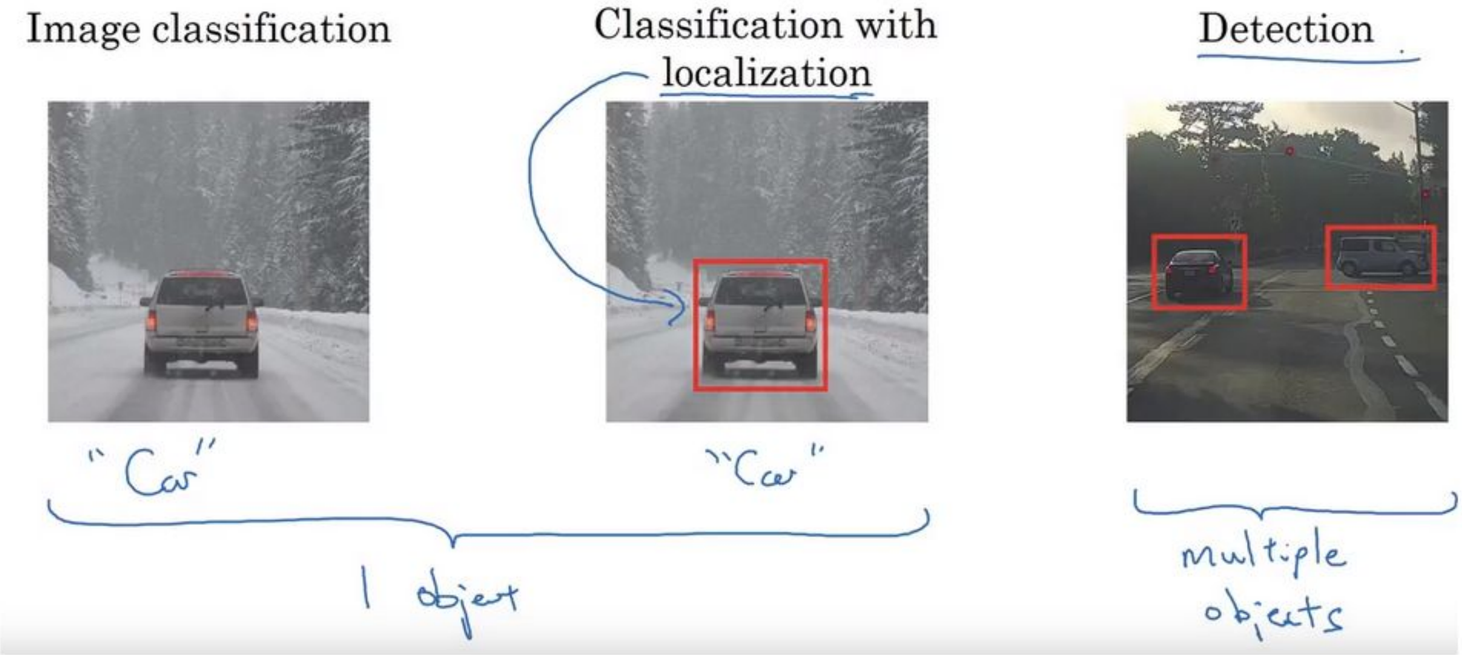
b - interaction with **surface** of visited domain

Reinforced random walk

Idea: Predict macroscopic parameters of the model from microstate represented as an image.

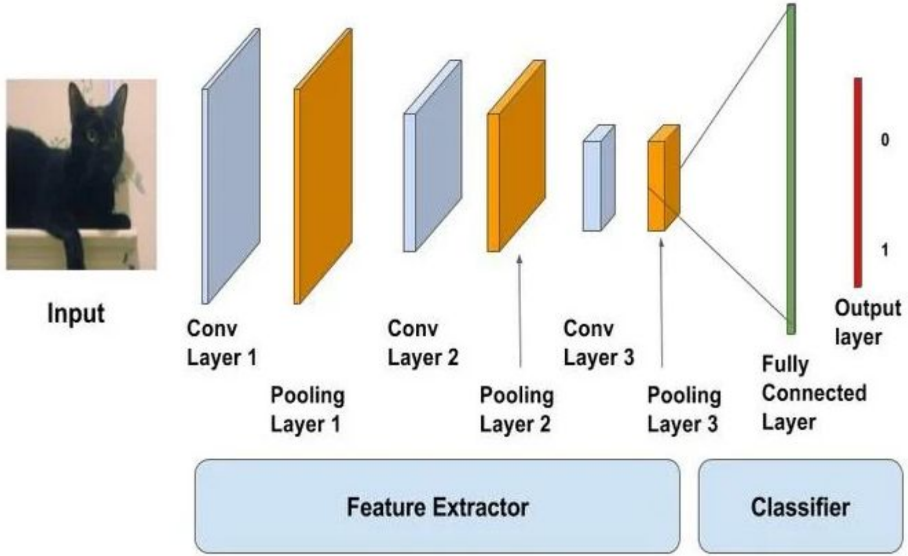





Different computer vision tasks



Terminologies example. Source:deeplearning.ai C4W3L01

Classical CNN architecture for image classification



1) Convolution, max pooling, dropout
2) Fully connected layers
3) One hot encoded labels

Problem: overfitting because of too large number of parameters

Classical CNN architecture for object classification

1) Convolution, max pooling, dropout
2) Fully connected layers
3) One hot encoded labels

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 14, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 7, 7, 64)          36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 3136)              0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                200768
_________________________________________________________________
activation_1 (Activation)    (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 257,162
Trainable params: 257,162
Non-trainable params: 0
```

Problem: overfitting because of too large number of parameters
Fully connected layers contribute most to total number of parameters

Previous models for object detection

Benchmark datasets:

1) PASCAL VOC
2) COCO

Target metric:

mAP (mean average precision)

R-CNN

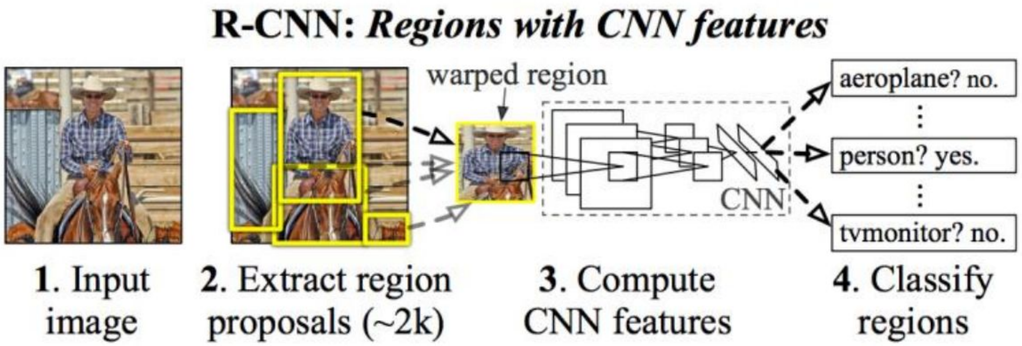
[Girshick R. Fast r-cnn //Proceedings of the IEEE international conference on computer vision. – 2015. – C. 1440-1448.]

Previous models for object detection

R-CNN (Regions with Convolutional Neural Networks)

[Girshick R. Fast r-cnn //Proceedings of the IEEE international conference on computer vision. – 2015. – C. 1440-1448.]

Steps of object detection by R-CNN
1) Region proposals
2) Rescale image
3) Feed image to image classifier
4) Repeat many many times



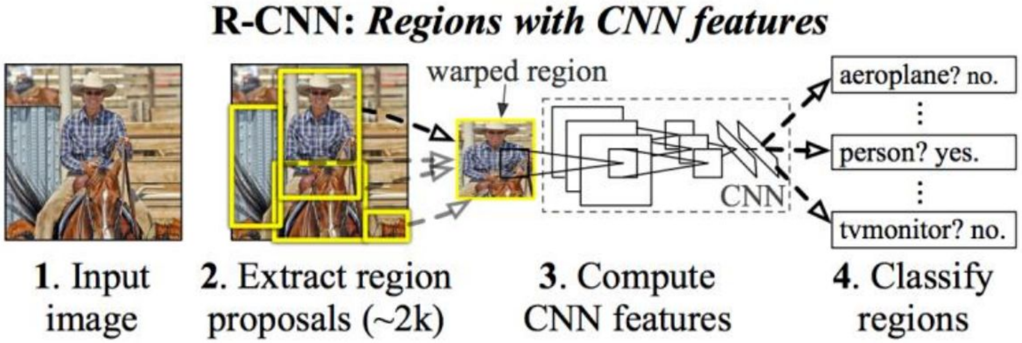
Previous models for object detection

R-CNN (Regions with Convolutional Neural Networks)

[Girshick R. Fast r-cnn //Proceedings of the IEEE international conference on computer vision. – 2015. – C. 1440-1448.]

Steps of object detection by R-CNN

1) Region proposals
2) Rescale image
3) Feed image to image classifier
4) Repeat many many times



Problem: too slow prediction, need to evaluate classifier many times

Previous models for object detection

YOLO (You Only Look Once)

[Redmon J. et al. You only look once: Unified, real-time object detection //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – С. 779-788.]

Steps of object detection by R-CNN

1) Region proposals
2) Rescale image
3) Feed image to image classifier
4) Repeat many many times

YOLO: reformulate object detection as regression problem
As result in one evaluation it is possible to predict probabilities and bounding boxes for multiple objects at one evaluation

Previous models for object detection

YOLO (You Only Look Once)

[Redmon J. et al. You only look once: Unified, real-time object detection //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – C. 779-788.]

YOLO: reformulate object detection as regression problem
As result in one evaluation it is possible to predict probabilities and bounding boxes at one network evaluation

More complex labels format is used

Previous models for object detection

More about PASCAL VOL labels format

Figure 1: Example datapoint in PascalVOC

(a) Image: 2008_000089.jpg

(b) Annotation: 2008_000089.xml


```
<annotation>
  <folder>VOC2012</folder>
  <filename>2008_000089.jpg</filename>
  <source>
    <database>The VOC2008 Database</database>
    <annotation>PASCAL VOC2008</annotation>
    <image>flickr</image>
  </source>
  <size>
    <width>376</width>
    <height>500</height>
    <depth>3</depth>
  </size>
  <segmented>1</segmented>
  <object>
    <name>chair</name>
    <pose>Frontal</pose>
    <truncated>0</truncated>
    <occluded>0</occluded>
    <bndbox>
      <xmin>71</xmin>
      <ymin>18</ymin>
      <xmax>307</xmax>
      <ymax>494</ymax>
    </bndbox>
    <difficult>0</difficult>
  </object>
</annotation>
```


YOLO v1 input:

448x448x3 tensor (RGB image)

YOLO v1 output:

7x7x30 tensor

For YOLO training PASCAL VOC label should be converted to 7x7x30 tensor encoding objects position

Previous models for object detection

Instead of predicting the center of the bounding box normalized by the width and height of the image, Yolo predicts xy-offsets relative to a cell in a 7×7 grid.

YOLO v1 input:

448x448x3 tensor (RGB image)

YOLO v1 output:

7x7x30 tensor

For YOLO training PASCAL VOC label should be converted to 7x7x30 tensor encoding objects position

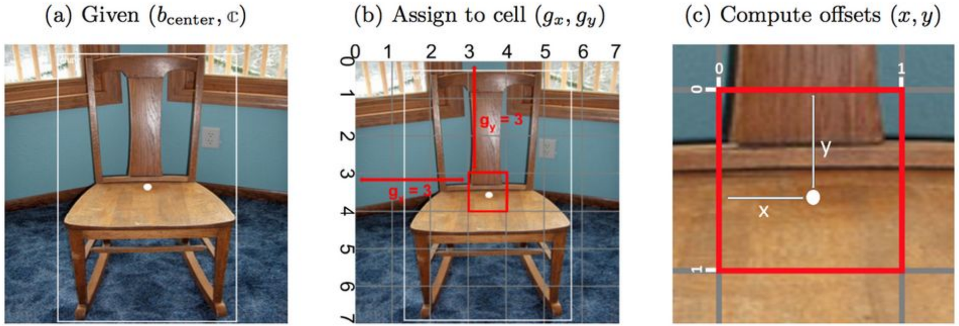
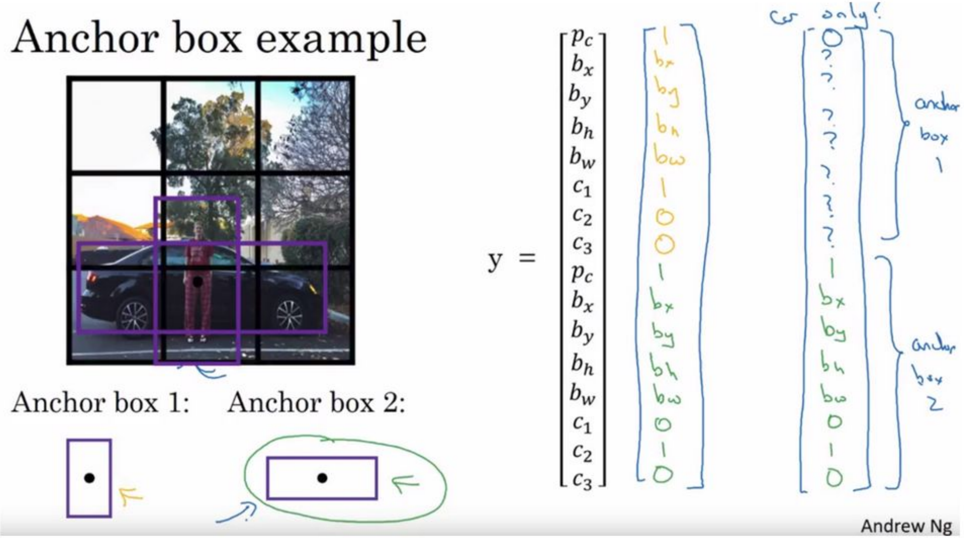


Figure 3: Visualizing how an object is assigned to a grid cell

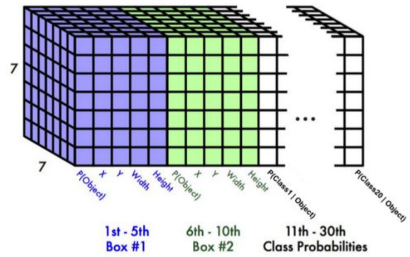
Previous models for object detection



Anchor box example. Source: deeplearning.ai C4W3L08

Figure 2: Yolo CNN, output interpretation and loss (images modified from [5] or [1])

(a) Yolo CNN output interpretation [5]



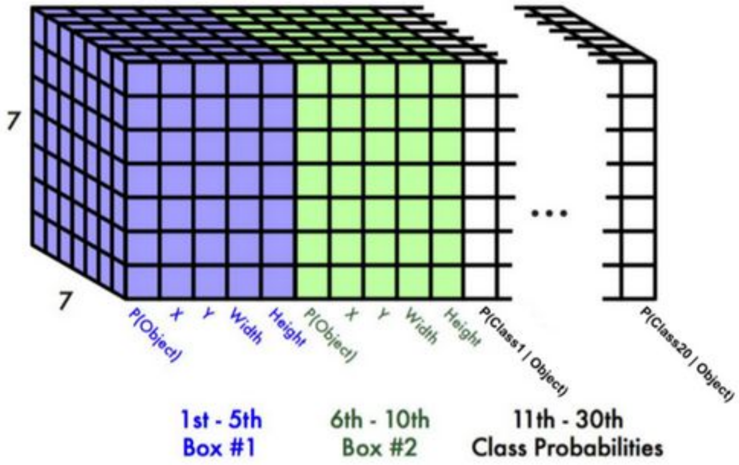
(b) Yolo loss function in [1].

$$\lambda_{\text{coord}}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{obj}}\left[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2\right]$$
$$+\lambda_{\text{coord}}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{obj}}\left[\left(\sqrt{w_i}-\sqrt{\hat{w}_i}\right)^2+\left(\sqrt{h_i}-\sqrt{\hat{h}_i}\right)^2\right]$$
$$+\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{obj}}\left(C_i-\hat{C}_i\right)^2$$
$$+\lambda_{\text{noobj}}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{noobj}}\left(C_i-\hat{C}_i\right)^2$$
$$+\sum_{i=0}^{S^2}\mathbb{1}_{i}^{\text{obj}}\sum_{c\in\text{classes}}(p_i(c)-\hat{p}_i(c))^2$$

Previous models for object detection

Figure 2: Yolo CNN, output interpretation and loss (images modified from [5] or [1])

(a) Yolo CNN output interpretation [5]

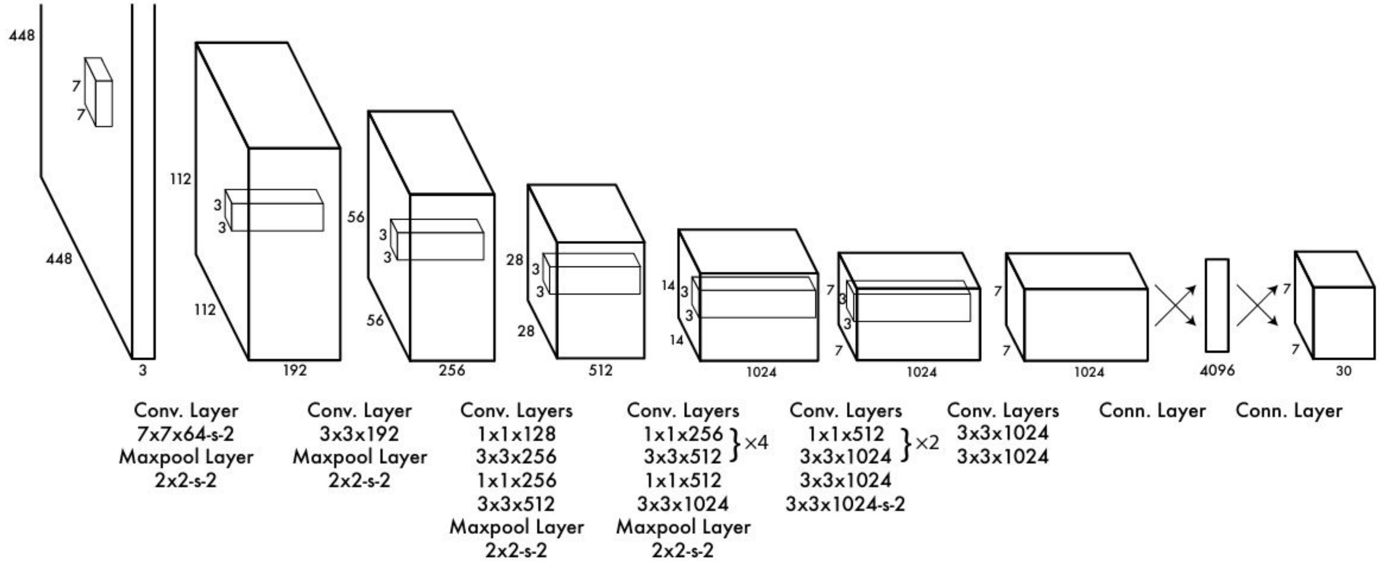


(b) Yolo loss function in [1].

$$
\begin{aligned}
&\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2
\end{aligned}
$$

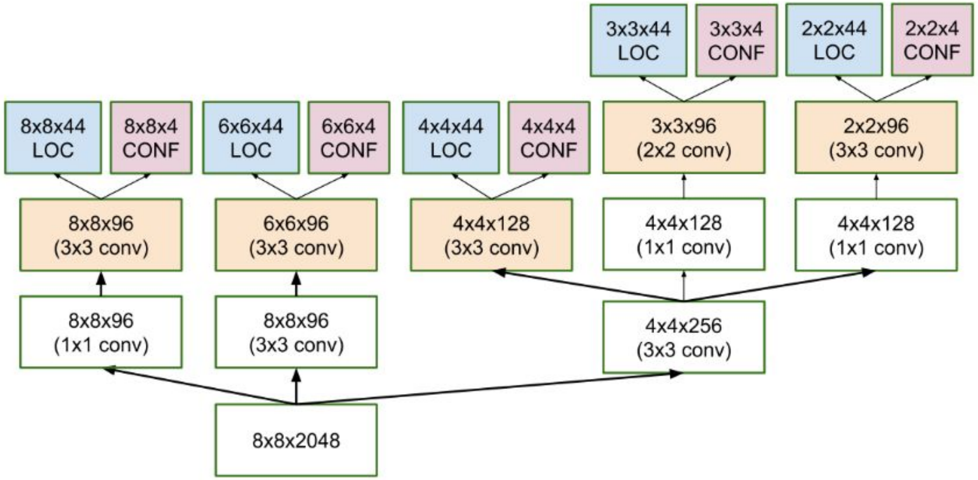
Previous models for object detection

YOLO model **still contains fully connected layers** which leads to large number of parameters and risk of overfitting if applied to small custom datasets



Difference between SSD and YOLO architectures

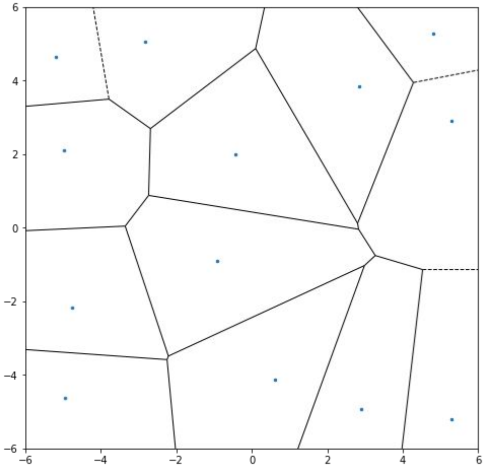
SSD also takes in final prediction results information from different image scales

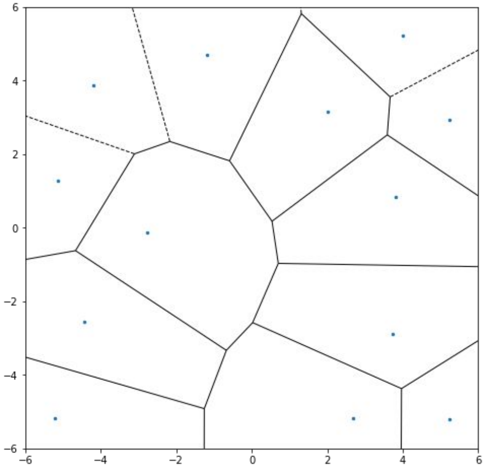


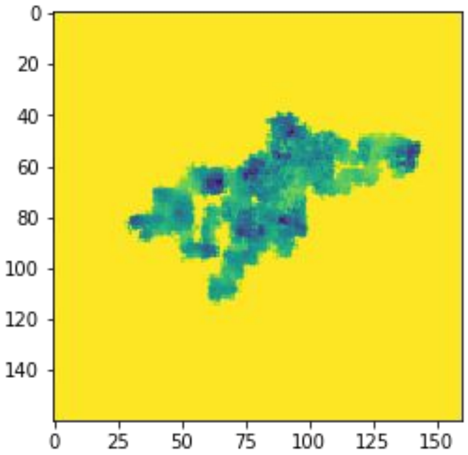
Architecture of multi-scale convolutional prediction of the location and confidences of multibox

SSD-500 (the highest resolution variant using 512x512 input images) achieves best mAP on Pascal VOC2007 at 76.8%, but at the expense of speed, where its frame rate drops to 22 fps. SSD-300 is thus a much better trade-off with 74.3 mAP at 59 fps.

Application to reinforced random walk





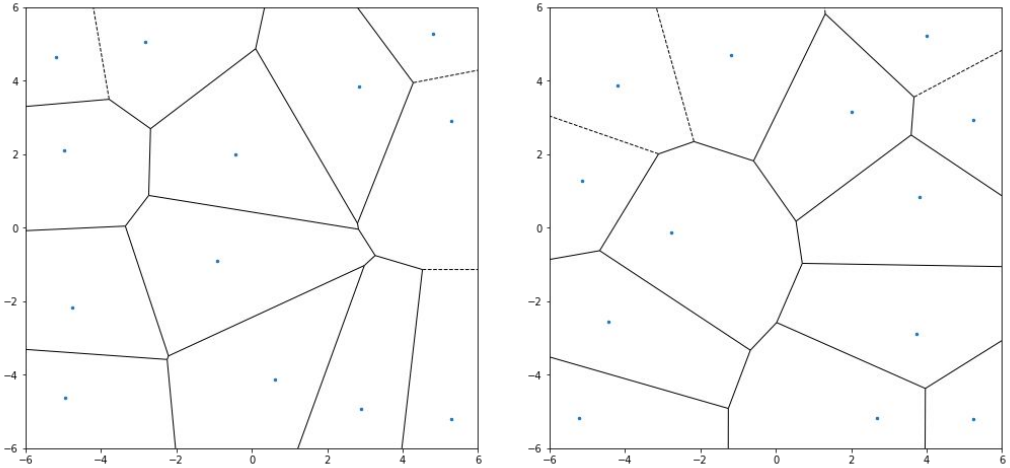


Input tensor representing single microstate

Input shape: 160x160 tensor
Output shape: 13x128 tensor
Tessellation of phase diagram on 13 domains, and we have 128 such tessellations.

log(a) in [-6, 6]
log(b) in [-6, 6]

We use SSD architecture to estimate model parameters



We encode one point on phase diagram as follows:

for every tessellation we encode domain in which lies given point as (0, 0, ..., 0, 1, 0, …, 0)

In total we have 128 such vectors of length 13.

Input shape: 160x160 tensor
Output shape: 13x128 tensor
Tessellation of phase diagram on 13 domains, and we have 128 such tessellations.

log(a) in [-6, 6]
log(b) in [-6, 6]

Results. Estimated phase diagram