Capturing Spatial Dependencies with Deep Neural Networks II

Yao-Yi Chiang
Computer Science and Engineering
University of Minnesota
yaoyi@umn.edu
Semantic Segmentation and Object Detection

A short introduction
Recall: Overhead Imagery Understanding Tasks

CNN so far - Image Classification

• Take an input image, predict an object class

Semantic Segmentation
Semantic Segmentation

• Output a class map for each pixel (here: dog vs background)

• Also, **instance segmentation**: specify each object instance as well (two dogs have different instances)
  • This can be done through **object detection** + **segmentation**
Sliding Window

- Every window provides some context for classifying the center pixel
- Inefficient, why?
Fully Convolutional Network (FCN)

- Predict / backpropagate for every output pixel
- Aggregate maps from several convolutions at different scales
- Upsampling/unpooling back to dense pixels for classification

Learnable Upsampling

• 1D Example
Learnable Upsampling: Transposed Convolution

\[
\begin{align*}
\text{Input} & \quad = \quad 0 & 0 & \quad + \quad 0 & 1 & \quad + \quad 0 & 2 & \quad + \quad 0 & 3 \\
& \quad \quad 0 & 0 & \quad \quad 0 & 1 & \quad \quad 0 & 2 & \quad \quad 0 & 3 \\
& \quad \quad 2 & 3 & \quad \quad 2 & 3 & \quad \quad 4 & 6 & \quad \quad 6 & 9 \\
\text{Kernel} & \quad = \quad 0 & 1 & \quad \quad 0 & 1 & \quad \quad 0 & 3 & \quad \quad 0 & 4 & \\
& \quad \quad 2 & 3 & \quad \quad 2 & 3 & \quad \quad 6 & 9 & \quad \quad 6 & 9 \\
\text{Output} & \quad = \quad 0 & 0 & 1 & \quad \quad 0 & 4 & 6 & \quad \quad 0 & 4 & 12 & 9
\end{align*}
\]

https://d2l.ai/chapter_computer-vision/transposed-conv.html
U-Net

- **Skip connections** between corresponding convolution and deconvolution layers
- **Sharper masks** by using precise spatial information (early layers)
- **Better object detection** by using semantic information (late layers)
- Weighted loss for clear boundaries

**Fig. 3.** HeLa cells on glass recorded with DIC (differential interference contrast) microscopy. (a) raw image. (b) overlay with ground truth segmentation. Different colors indicate different instances of the HeLa cells. (c) generated segmentation mask (white: foreground, black: background). (d) map with a pixel-wise loss weight to force the network to learn the border pixels.
Hourglass Network

- U-Net like architectures repeated sequentially
- Each block refines the segmentation for the following
- Each block has a segmentation loss
DeepLab V3

- Atrous convolutions, or dilated convolutions
  - Arbitrarily enlarge the field-of-view of filters
  - Compute the responses of any layer at any desirable resolution
- Atrous Spatial Pyramid Pooling (ASPP)
  - To classify the center pixel (orange), ASPP exploits multi-scale features by employing multiple parallel filters with different rates
  - The effective Field-Of-Views are shown in different colors

https://arxiv.org/abs/1606.00915v2
Semantic Segmentation Summary

Fei-Fei Li, Ranjay Krishna, Danfei Xu. Lecture 15
Semantic Segmentation on PASCAL VOC 2012 test

https://paperswithcode.com/sota/object-detection-on-coco
Object Detection
Localization

• Single object per image
• Predict coordinates of a bounding box \((x, y, w, h)\)
• Evaluate via Intersection over Union (IoU) (Jaccard Similarity)
Localization as regression
Localization as regression

The sum of the all squared differences between the true value and the predicted value.
Classification + Localization

CNN

7x7 x 2048
conv feature map

class scores
• Use a pre-trained CNN on ImageNet (e.g., ResNet)
• The "localization head" is trained separately with regression (i.e., generate coordinates in a continuous space)
• Possible end-to-end finetuning of both tasks
• At test time, use both heads
Classification + Localization

- C classes, 4 output dimensions (1 box)
- **Predict exactly N objects**: predict (N×4) coordinates and (N×K) class scores
Object Detection

• We don't know in advance the number of objects in the image.
• Object detection relies on *object proposal* and *object classification*
• **Object proposal**: find regions of interest (RoIs) in the image
• **Object classification**: classify the object in these regions
• Two main families:
  • Single-Stage: A grid in the image where each cell is a proposal (SSD, YOLO, RetinaNet)
  • Two-Stage: Region proposal then classification (Faster-RCNN)
YOLO – You Only Look Once

• Slice the image into cells
• Every cell predicts for
  • Locations of bounding boxes and confidence scores
  • Class probabilities

YOLO – You Only Look Once

- Every cell predicts a number of B bounding boxes: $[P_c, B_x, B_y, B_w, B_h]$
  - $P_c$: $\text{Pr}(\text{obj}) \times \text{IOU}$, the probability of an object in the cell $\times \text{IOU}$ between the predicted box and any ground truth box
  - $B_x, B_y, B_w, B_h$, object bounding box location (relative to the cell) and size (relative to the image)

- Every cell also predicts C conditional class probabilities for all boxes:
  - $C_1, C_2, \ldots, C_n$: $\text{Pr}(\text{Class}_i \mid \text{obj} = \text{true})$ Give the cell contains an object, what is the probability of the object

https://www.coursera.org/learn/convolutional-neural-networks
https://polakowo.io/datadocs/docs/deep-learning/object-detection
YOLO – You Only Look Once

• **Non-maximum suppression** to merge detected boxes with high confidence

• For each object class:
  • discard bounding boxes where probability of object being present is **below some threshold**, say 0.6
  • take the bounding box with the highest score,
  • discard any remaining bounding boxes with IOU value above some threshold (0.5)

YOLO – You Only Look Once

- After ImageNet pretraining, the whole network is trained end-to-end
- The loss is a weighted sum of different regressions

\[
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2]
\]
\[
+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right]
\]
\[
+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left( C_i - \hat{C}_i \right)^2
\]
\[
+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left( C_i - \hat{C}_i \right)^2
\]
\[
+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_{t}(c) - \hat{p}_{t}(c))^2
\] (3)

YOLO9000 / YOLOv2

- Better, Faster, Stronger
- Instead of directly predict any bounding box sizes from a cell, use anchor boxes as prior knowledge and then predict to adjust the boxes
- Anchor boxes
  - Predefined dimensions of bounding boxes
  - Calculated from training data using K-Means with IOU as the distance measure
  - When K=5, every cell predicts to adjust 5 anchor boxes
  - Help detect multiple objects centered at a cell

RetinaNet

- One-stage object detection
- Feature Pyramid Networks (FPN)
  - Effectively handle multiscale objects
- FocalLoss
  - Help the networks to focus on hard classification

Figure 1. (a) Using an image pyramid to build a feature pyramid. Features are computed on each of the image scales independently, which is slow. (b) Recent detection systems have opted to use only single scale features for faster detection. (c) An alternative is to reuse the pyramidal feature hierarchy computed by a ConvNet as if it were a featurized image pyramid. (d) Our proposed Feature Pyramid Network (FPN) is fast like (b) and (c), but more accurate. In this figure, feature maps are indicated by blue outlines and thicker outlines denote semantically stronger features.

https://arxiv.org/abs/1612.03144
FocalLoss

• Class Imbalance Problem
  • Information related to one class in a dataset used in training is over-represented than the other classes
  • Make the network biased towards learning more representations of the data-dominated class and other classes will be underlooked

• Focal Loss
  • Reduce the loss for “well-classified examples” (e.g., p > 0.5 and it's correct)
  • Increase loss for “hard-to-classify examples” (e.g., p < 0.5)
  • Help the models focus on the rare class in case of class imbalance.

See also Dice Loss for semantic segmentation: https://www.youtube.com/watch?v=NqDBvUPD9jg
https://amaarora.github.io/2020/06/29/FocalLoss.html
R-CNN

- Selective search to find boxes of object candidates (slow, CPU time, 47 sec. per image) – recursively combine similar regions
- ConvNet trained on individual regions
- Predict a class label for each box using a support vector machine (SVM)
- Predict offsets of the bounding boxes to improve box precision

https://www.geeksforgeeks.org/selective-search-for-object-detection-r-cnn/
Fast R-CNN

• Find regions of interest from the feature map (selective search, but instead of from the input image)
• ConvNet trained on the entire image and use ROI pooling to generate fixed feature vector for each region for prediction
• Much faster than R-CNN
ROI pooling

• Use ROI pooling to generate a fixed-size feature vector for each proposal for final classification
• Allows to propagate gradient only on interesting regions, and efficient computation
Faster-RCNN

• Region proposal network (RPN): Take an image (of any size) as input and outputs a set of rectangular object proposals, each with an objectness score
  • Use anchor boxes
  • Lots of proposals (high recall, low precision)

• Classification: decide if a proposed region has an object

Object Detection Summary

• YOLO family
  • One stage
  • Very fast, better than real-time (45 – 150 frames per second)
  • Might have lower recall compared to Faster RCNN
  • Might not work well for small objects

• Faster RCNN family
  • Two stage
  • The second stage can focus on ROI proposals
  • Might be slower than YOLO
Object Detection on COCO test-dev

Leaderboard

Dataset

View box AP by Date for All models

https://paperswithcode.com/sota/object-detection-on-coco
Object Detection + Semantic Segmentation
Instance Segmentation: Mask R-CNN

Fei-Fei Li, Ranjay Krishna, Danfei Xu. He et al, “Mask R-CNN”, arXiv 2017
Mask R-CNN Results
Overhead Imagery
Understanding
Overhead Imagery Understanding Tasks

• Typically customize existing CV models with pretrained weights
  • YOLO or Faster R-CNN for object detection
  • Unet or similar architecture for semantic segmentation

Open Challenges

• The need of customization of network architectures come from open challenges in dealing with overhead imagery
  • Small objects
  • Imbalanced data, inconsistent training data domain (e.g., scenic photos vs overhead imagery)
  • Large search space/large image
  • Multi-scale
  • Multi-orientation
  • Multi-bands
  • Occlusion (camera angle or vertical occlusion)
  • Varying imagery quality (shadows, cloud covers, over/under-exposition)
You Only Look Twice: Rapid Multi-Scale Object Detection In Satellite Imagery

Figure 3: Limitations of the YOLO framework (left column, quotes from [10]), along with YOLT contributions to address these limitations (right column).

https://arxiv.org/abs/1805.09512
Generating Location Specific Training Data

• **Location** is the key to link various types of data
  • e.g., can provide context-rich annotated (training) data (if we do it correctly)

  “By using the data in OSM, we were able to collect more than 100 million labeled examples to add to our training data set.”

  “However, using OSM data for labels presented several challenges that required novel approaches to overcome.”

Microsoft Building Footprints

Conflation: Generating your own training data

A short introduction
Conflation

- Integrating multiple geo-spatial datasets by establishing the correspondence between matched entities (control point pairs) and transforming the other objects accordingly.
Conflation
Challenges

• Different projections, accuracy levels, resolutions
• Result in spatial inconsistencies
AMS-Conflation to Align Vector and Imagery

Control Point Detection

Filtering Technique

Triangulation and Rubber-Sheeting

Localized Template Matching (LTM)

Bayes Classifier

Matching by Correlation

road width and road directions

Filtering Control Point Pairs Using Vector Median Filter (VMF)

Triangulation & Rubbersheeting

• Delaunay triangulation on the CP pairs (maximize the minimum angle of all the angles of the triangles)
• Use the 3 CP pairs for each triangle to calculate a transformation matrix
• Transform data within each triangle according to the matrix

MO-DOT+ High-resolution USGS Color Image

Red lines: Original MO-DOT
Blue lines: Conflated lines
NAVSTREETS+ High-resolution USGS Color Image

Red lines: Original NAVSTREETS roads
Yellow lines: Conflated lines
Conflation Results: Parcel data
Matching Point Sets for Conflation

- Sometimes one data source does not provide the exact geocordinates
- Find salient features from both datasets for matching
  - e.g., road intersections
- A point pattern matching problem

https://doi.org/10.1145/1032222.1032231
Point Pattern Matching

• Pick a pair of points in first dataset
Point Pattern Matching

- Pick a pair of points in first dataset
- Pick two random points in the second dataset
Point Pattern Matching

• Pick a pair of points in first dataset
• Pick two random points in the second dataset
• Apply the same transformation all points
• Calculate the number of matches

Repeat until every point has a match

• Complexity is very high!
GeoPPM: Exploit Connectivity

Light Green: Impossible candidates for $P_1$
Orange: Possible candidates for $P_1$
GeoPPM: Exploit Orientation

Orange: Possible candidates for $P_1$
Light Green: Impossible candidates for $P_1$
GeoPPM: Exploit Angles between Points

Some of the possible candidate pairs for $P_1, P_2$

Some of the impossible candidate pairs for $P_1, P_2$
GeoPPM: Exploit Density between Points

Some of the possible Candidate pairs for $P_1, P_2$

Some of the impossible Candidate pairs for $P_1, P_2$
Conflation Summery

- We can integrate geographic data from different sources by establishing correspondence between datasets.
- Conflation helps:
  - generate accurate training data for imagery understanding (and other machine learning tasks)
  - integrate the extracted data from imagery to existing data

https://ai.facebook.com/blog/mapping-roads-through-deep-learning-and-weakly-supervised-training/
One more thing: EarthScan

- How to effectively exploit prior knowledge of things in space for
  - Detecting objects from overhead imagery using reduced numbers and varieties of annotated visual samples, and
  - Parsing overhead imagery into complete geographic layers

Resources

• Datasets: https://github.com/chrieke/awesome-satellite-imagery-datasets
• Methods/Code: https://github.com/robmarkcole/satellite-image-deep-learning
Acknowledgements

• Deep learning slides adapted from https://m2dsupsdlclass.github.io/lectures-labs/ by Olivier Grisel and Charles Ollion (CC-By 4.0 license)

• Gil, Yolanda (Ed.) Introduction to Computational Thinking and Data Science. Available from http://www.datascience4all.org
These materials are released under a CC-BY License

https://creativecommons.org/licenses/by/2.0/

You are free to:

Share — copy and redistribute the material in any medium or format
Adapt — remix, transform, and build upon the material
for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

Please credit as: Chiang, Yao-Yi Introduction to Spatial Artificial Intelligence. Available from https://yaoyichi.github.io/spatial-ai.html

If you use an individual slide, please place the following at the bottom: “Credit: https://yaoyichi.github.io/spatial-ai.html

We welcome your feedback and contributions.

Credit: http://www.datascience4all.org/