CS590-DVC
Deep Visual Computing:
Deep Image Segmentation

Daniel G. Aliaga
Fall 2020
Sources:

• Image Segmentation Using Deep Learning: A Survey
  • Minaee, Boykov, Porikli, Plaza, Kehtarnavaz, and Terzopoulos [2020]

• Evolution of Image Segmentation using Deep Convolutional Neural Network: A Survey
  • Sultanaa, Sufian, Duttab [2020]

• A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D images
  • Ulku, Akagunduz [2020]
Deep Image Segmentation Timeline
Pre-Deep Learning

• Thresholding [1979]
• Histogram-based segmentation
• Region Growing
• Clustering
• Watershed
• Active contours
• Graph cuts
• Markov Random Fields
Histogram Based Segmentation

Raju and Neelima, 2012
Region Growing

• From a random set of seeds, grow regions of similar pixels, using for example 4-connected or 8-connected neighborhood
Clustering (K-means, EM)

Different cluster analysis results on "mouse" data set:

- Original Data
- k-Means Clustering
- EM Clustering

- Uses a closeness metric which could be spatial-distance, image color, or a combination for example

[Wikipedia, 2020]
Watershed Algorithms

• In some space, e.g. gradient map, find borders separating “bodies of water” and then fill-in each body of water.

Fig. 7. Example segmentations with more than two labels. (a) Seeds, (b, c) power watershed result ($q = 2$).

[Couprie et al., 2020]
Active Contours (or Snakes)

• The control points of a deformable curve (e.g., spline) is influenced by constraint and image forces that pull it towards object contours and internal forces

[Debakla et al., 2011]
Graph Cuts

- Pixels are arranged in a graph connected to source/sink for foreground/background.

- Image-dependent metric used for similarity (here edge weight implies membership value).

- Then a “min cut” is made through edges so that nodes remain joined with their maximum weights.

- Often scribbles are used to seed the regions.

[Debakla et al., 2011]

Figure 2: Example segmentation of a very simple 3-by-3 image. Edge thickness corresponds to the associated edge weight. (Image courtesy of Yuri Boykov.)
[Debakla et al., 2011]
Markov Random Fields (MRFs)

• Define a probability measure on set of all possible labelings per pixel
• Maximize $P(\omega|f)$ (probability of labeling $\omega$ given feature(s) $f$):
  $\omega^{MAP} = \arg\min_{\omega} P(\omega|f) \leftarrow$ MAP estimate
• Exploits regions are often homogenous; neighboring pixels usually have similar properties (intensity, color, texture, ...)
• Allows MCMC sampling of the underlying hidden/complex structure

[Grinias et al., 2016]
What is MCMC?

• Markov Chain Monte-Carlo
  Stochastic optimization with many parallel threads, different ‘temperatures’
  and varying types of proposed state changes

• Take a peek at:
Conditional Random Field (CRF)

• Proposed by Lafferty et al. 2001

• Is a particular form of MRF that specifies the probabilities of possible label sequences given an observation sequence

• Effectively, each random variable may also be conditioned upon a set of observations

• More suitable for segmenting time-series data, for example.
Deep Learning Based Image Segmentation

1) Fully convolutional networks
2) Deep Lab and upsampling
3) Deconvolution networks (U-Net and V-Net in separate slides)
4) GAN-based segmentation (in separate slides)
5) Recurrent neural network based models
6) Instance Segmentation: Deep Mask and Region proposal networks
7) Combined Semantic and Instance Segmentation
• Convolutionalize proven classification architectures: AlexNet, VGG, and GoogLeNet

• Why?
  Trained end-to-end, pixels-to-pixels, improves previous best result and supports arbitrary input size
Convolutionalization

- Remove classification layer and convert all fully connected layers to convolutions
- Append 1x1 convolution with channel dimensions and predict scores at each of the coarse output locations
- Upsample via deconvolution to produce pixel-to-pixel input -> output mapping
- Add skip connections to improve level of detail
What is a 1x1 convolution?

- It is one way to keep resolution but reduce number of channels:
How to upsample?

• One option is to use deconvolution, which really should be called *transposed convolution*

• Recall convolution:
How to upsample?

• One option is to use deconvolution, which really should be called *transposed convolution*.

[Dumoulin & Visin 2016]
Why skip connections?

• Once augmented with skips, the network makes and fuses predictions from several streams that are learned jointly and end-to-end
Results

Fig. 4. Refining fully convolutional networks by fusing information from layers with different strides improves spatial detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).
Results

https://www.youtube.com/watch?v=xr_2dwZDH6U
Beyond CNN

• CNNs successful in part because of their invariance to local transformations
  • Remember why?
    Answer: Pooling...

• While good for high-level tasks like classification, not good for mid-level tasks like segmentation...

• What can we do?
- CNNs successful in part because of their invariance to local transformations
  - Remember why?
    Answer: Pooling...

- While good for high-level tasks like classification, not good for mid-level tasks like segmentation...

- What can we do?
Instead, let's do:

1. Upsampling:
   - Use the 'atrous' (with holes) algorithm originally developed for efficiently computing the undecimated discrete wavelet transform (Mallat, 1999)
   - Use bilinear interpolation

2. Boost model's ability to capture fine details by employing a fully-connected Conditional Random Field (CRF)
Atrous Algorithm

- *trous* means “holes” in French
- In wavelet transform, scheme known as “algorithme a trous” – they define the term *atrous convolution* as a shorthand for convolution with upsampled filters
- Upsampling is done by inserting holes between nonzero filter taps
- 4x upsampling performed
Bilinear Interpolation

• Further upsampling done using bilinear interpolation (8x factor)

• Works ok because function is relatively smooth

• Is cheaper and simpler than the deconvolution (aka transposed convolution) of FCNs
• Instead:
  • 1. Upsampling:
    • Use the ‘atrous’ (with holes) algorithm originally developed for efficiently computing the undecimated discrete wavelet transform (Mallat, 1999)
    • Use bilinear interpolation
  • 2. Boost model’s ability to capture fine details by employing a fully-connected Conditional Random Field (CRF)
Fully Connected CRF

• What is it?
  • A graph is formed by connecting all pairs of pixels
  • Conditioned dependencies

• Two feature spaces:
  • Bilateral: forces pixels with similar color and position to have similar labels
  • Smoothness: only considers spatial proximity to enforce smoothness

• Method of Krahenbuhl and Koltun [2011] used to perform an efficient inference algorithm in which the edge potentials (between pairs of pixels) are defined by a linear combination of Gaussian kernels in the feature space
Fully Connected CRF

Fig. 6: PASCAL VOC 2012 val results. Input image and our DeepLab results before/after CRF.
DeepLab Successors:

• DeepLabV2
  • uses ResNet and Atrous Spatial Pyramid Pooling (ASPP)
    [PAMI 2018]

• DeepLabV3
  • removes dense CRF and does Atrous Convolution “on steroids”
    [2017/2018]

https://www.youtube.com/watch?v=culrijsu9GY
DeepLab V2:
Atrous Spatial Pyramid Pooling (ASPP)

• First, what is spatial pyramid pooling?
Spatial Pyramid Pooling (SPP)

- An additional set of layers in between conv-layers and FC layers in a typical CNN network:

  - Provides multiple pooling layers with different scales
Spatial Pyramid Pooling (SPP)

- Often combined with multi-size training (e.g., 224x224 and 180x180)
- SPP together with multi-size training has proven to be beneficial overall
Atrous Spatial Pyramid Pooling (ASPP)

- Basically, SPP with the Atrous Convolution Algorithm
DeepLab V3

- Essentially, investigates how much and where to perform atrous convolution, and also does ASPP
- They omit using dense CRF yet obtain superior performance (85.7% on PASCAL VOC dataset)
Learning Deconvolution Network for Semantic Segmentation
Noh, Hong, Han, ICCV 2015

- Called “DeConvNet”; is an encoder-decoder style network

Figure 2. Overall architecture of the proposed network. On top of the convolution network based on VGG 16-layer net, we put a multi-layer deconvolution network to generate the accurate segmentation map of an input proposal. Given a feature representation obtained from the convolution network, dense pixel-wise class prediction map is constructed through multiple series of unpooling, deconvolution and rectification operations.
Convolution Part

• Basic on standard multiple convolution and max-pooling layer pairs
Deconvolution Part

• Typical max-pooling is uninvertible
• How to make max-pooling invertible?
Switches

- Record max locations during pooling
- During unpooling uses switches to place the reconstructions from the layer above into appropriate locations

Figure 1. Top: A deconvnet layer (left) attached to a convnet layer (right). The deconvnet will reconstruct an approximate version of the convnet features from the layer beneath. Bottom: An illustration of the unpooling operation in the deconvnet, using switches which record the location of the local max in each pooling region (colored zones) during pooling in the convnet.

[Courtesy Zeller and Fergus, 2013]
Figure 2. Overall architecture of the proposed network. On top of the convolution network based on VGG 16-layer net, we put a multi-layer deconvolution network to generate the accurate segmentation map of an input proposal. Given a feature representation obtained from the convolution network, dense pixel-wise class prediction map is constructed through multiple series of unpooling, deconvolution and rectification operations.
Deconvolution Example

Figure 4. Visualization of activations in our deconvolution network. The activation maps from top left to bottom right correspond to the output maps from lower to higher layers in the deconvolution network. We select the most representative activation in each layer for effective visualization. The image in (a) is an input, and the rest are the outputs from (b) the last $14 \times 14$ deconvolutional layer, (c) the $28 \times 28$ unpooling layer, (d) the last $28 \times 28$ deconvolutional layer, (e) the $56 \times 56$ unpooling layer, (f) the last $56 \times 56$ deconvolutional layer, (g) the $112 \times 112$ unpooling layer, (h) the last $112 \times 112$ deconvolutional layer, (i) the $224 \times 224$ unpooling layer and (j) the last $224 \times 224$ deconvolutional layer. The finer details of the object are revealed, as the features are forward-propagated through the layers in the deconvolution network. Note that noisy activations from background are suppressed through propagation while the activations closely related to the target classes are amplified. It shows that the learned filters in higher deconvolutional layers tend to capture class-specific shape information.
Results

- Comparison:

(and many more in their paper)
SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation
Badrinarayanan, Kendall, Cipolla, PAMI 2017

- Similar to DeconvNet but more compact memory wise and other improvements


- [https://www.youtube.com/watch?v=CxanE_W46ts](https://www.youtube.com/watch?v=CxanE_W46ts)
DeepMask: Learning to Segment Object Candidates [2017]

• Top branch: determines segmentation mask for centered object
• Bottom branch: estimates object score for the input patch

Figure 1: (Top) Model architecture: the network is split into two branches after the shared feature extraction layers. The top branch predicts a segmentation mask for the the object located at the center while the bottom branch predicts an object score for the input patch. (Bottom) Examples of training triplets: input patch $x$, mask $m$ and label $y$. Green patches contain objects that satisfy the specified constraints and therefore are assigned the label $y = 1$. Note that masks for negative examples (shown in red) are not used and are shown for illustrative purposes only.
DeepMask: Learning to Segment Object Candidates [2017]

• Score
  • Specifically, a patch is given high score if it satisfies the following constraints:
    • (i) the patch contains an object roughly centered in the input patch
    • (ii) the object is fully contained in the patch and in a given scale range

• Each sample $k$ in the training set contains
  • (1) the RGB input patch $x_k$,
  • (2) the binary mask corresponding to the input patch $m_k$ (with $m_{ijk} \in \{\pm 1\}$, where $(i, j)$ corresponds to a pixel location on the input patch), and
  • (3) a label $y_k \in \{\pm 1\}$ which specifies whether the patch contains an object.

• A patch $x_k$ is given label $y_k = 1$ if it satisfies the following constraints:
  • (i) the patch contains an object roughly centered in the input patch
  • (ii) the object is fully contained in the patch and in a given scale range

• Score and mask are learned simultaneously
ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks
Visin, Kastner, Cho, Matteucci, Courville, Bengio

• Uses RNNs
• What are RNNs?
Recurrent Neural Network (RNN)

• They are a class of neural networks good for processing temporal sequences (or any inputs that depend on prior inputs); the network encodes hidden states and dependencies

• How do they work?
Recurrent Neural Network (RNN)

• The base case:

\[ N_x = N_y = 1 \]

Standard Perceptron/Neural Network:

- \( x \) = input
- \( \hat{y} \) = output
- \( a^{<0>} \) = bias
- “blue box” = the neuron

[Images by Afshine Amidi and Shervine Amidi]
Recurrent Neural Network (RNN)

• From a sequence of inputs generate an output:
Recurrent Neural Network (RNN)

• From a sequence of inputs generate an output:

• $N_x > 1; N_y = 1$
Recurrent Neural Network (RNN)

• From an input generate a sequence of outputs:
Recurrent Neural Network (RNN)

• From an input generate a sequence of outputs:

\[ N_x = 1; \ N_y > 1 \]
Recurrent Neural Network (RNN)

• From an input sequence generate an output sequence of same size:

\[ N_x = N_y > 1 \]
Recurrent Neural Network (RNN)

• From an input sequence generate an output sequence:

\[ N_x \geq 1, \; N_y \geq 1 \]
Recurrent Neural Network (RNN)

- They are a class of neural networks good for processing temporal sequences (or any inputs that depend on prior inputs); the network encodes hidden states and dependencies.
Recurrent Neural Network (RNN)

For each timestep $t$, the activation $a^{<t>}$ and the output $y^{<t>}$ are expressed as follows:

$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$

and

$$y^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$$

where $W_{ax}, W_{aa}, W_{ya}, b_a, b_y$ are coefficients that are shared temporally and $g_1, g_2$ activation functions.
ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks
Visin, Kastner, Cho, Matteucci, Courville, Bengio

• How to apply RNNs to do image segmentation?
ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks
Visin, Kastner, Cho, Matteucci, Courville, Bengio

• Lets assume only a 1D RNN (just like the prior examples)
• What is the input $x$? What is the output $x$?
• Suggestion:
  • Flatten pixels in “X” (to 1D array)
  • And produce output “Y” (1D array)
  • And “Y” is the “X” of the next cell...
  • e.g.:
ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks
Visin, Kastner, Cho, Matteucci, Courville, Bengio

• Lets assume only a 1D RNN (just like the prior examples)
• What is the input \( x \)? What is the output \( x \)?
• Suggestion:
  • Flatten pixels in “X” (to 1D array)
  • And produce output “Y” (1D array)
  • And “Y” is the “X” of the next cell...
  • e.g.:

![Diagram showing the flow of input and output vectors in a recurrent neural network.]
ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks
Visin, Kastner, Cho, Matteucci, Courville, Bengio

• Lets assume only a 1D RNN (just like the prior examples)
• What is the input x? What is the output x?
• Suggestion:
  • Flatten pixels in “X” (to 1D array)
  • And produce output “Y” (1D array)
  • And “Y” is the “X” of the next cell...
  • e.g.: And also bidirectional for columns...
ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks
Visin, Kastner, Cho, Matteucci, Courville, Bengio

Figure 1: A one-layer ReNet

Figure 2: The ReNet network used for SVHN classification
Rich feature hierarchies for accurate object detection and semantic segmentation (R-CNN)
Girshick, Donahue, Darrell, Malik

• This paper is the first to show that a CNN can lead to dramatically higher object detection (not quite segmentation)
  • Exploits “Recognition using Regions”, by Gu et al. CVPR 2009
    • This paper computes features of regions for recognition
      • Exploits “From Contours to Regions: An Empirical Evaluation”, by Arbelaez et al. CVPR 2009
        • This paper computes contours and then uses Oriented Watershed Transform (OWT) and Ultrametric Contour Map (UCM) to define a OWT-UCM algorithm to determine a hierarchical set of contiguous pixel regions
From Contours to Regions: An Empirical Evaluation
Arbelaez et al. CVPR 2009

• They use a contour detector yielding image boundary probabilities and boundary orientations

• Build a sequence of contours using a watershed algorithm (OWT) and a method that prefers a hierarchy of closed contours (UCM)
1. Define a “bag of regions”

2. Compute 4 region features: contour shape, edge shape, color, texture

3. Learn discriminative weights for the features:
Rich feature hierarchies for accurate object detection and semantic segmentation (R-CNN)
Girshick, Donahue, Darrell, Malik

• Usage:
  • Given input image,
  • use selective search (or some other method) to generate region proposals,
  • warp each region proposal to a same-size image that is given to a CNN to obtain features,
  • use a SVM to determine which object type is being seen and also to refine the bounding box, and
  • output the highest probability label (and box).
Rich feature hierarchies for accurate object detection and semantic segmentation (R-CNN)
Girshick, Donahue, Darrell, Malik

• Training:
  • CNN pretrained
  • Use similar region proposals but with at least a threshold overlap with ground truth (e.g., 30%, 50%)

• Con
  • takes 40-50 seconds at usage-time to obtain a result (because need to process typically 2000 regions)

• Improvements?
Fast R-CNN
Girshick, ICCV 2015

• How to make R-CNN faster?
  • Instead of feeding 2000 region proposals to the CNN, feed one image and then feed 2000 region proposals on the reduced feature map
  • Then train a small fully-connected network and softmax layer to obtain the best proposal and a bounding box refinement

• About 10x faster
• How to go even faster?
Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
Ren et al. 2016

• How to go even faster?
  • Tack on a few more layers to the conv-net
  • Train it to produce regions (RPN) and also a set of features
  • Then, a similar region pooling and classification is done as with Fast R-CNN

• About 100x faster than the original R-CNN
You Only Look Once: Unified, Real-Time Object Detection (YOLO)  
Redmon et al. 2016

- Joint region proposal generation and classification into a single network

- Not as accurate as others, but 1000x faster than R-CNN


**Figure 1: The YOLO Detection System.** Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to $448 \times 448$, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model’s confidence.
Mask R-CNN
He et al. 2018

• Does instance segmentation by extending Faster R-CNN to also predict a segmentation mask via an additional branch
• The mask branch is a small FCN applied to each region-of-interest
Figure 2. **Mask R-CNN** results on the COCO test set. These results are based on ResNet-101 [19], achieving a mask AP of 35.7 and running at 5 fps. Masks are shown in color, and bounding box, category, and confidences are also shown.
UPSNet: A Unified Panoptic Segmentation Network
Xiong et al. 2019

• Joins semantic segmentation (often CNN based) with instance segmentation (often region-proposal based)

• Predicts “things” (with instance head) and “stuff” (with semantic head)

  * things = “people, car, bicycle, ...” and stuff = “sky, ground, ...”
UPSNet: A Unified Panoptic Segmentation Network
Xiong et al. 2019

Figure 4: Visual examples of panoptic segmentation. 1-st and 2-nd rows are from COCO. 3-rd and 4-th rows are from Cityscapes. 5-th and 6-th rows are from our internal dataset.
More Deep Image Segmentation Works

• SharpMask (successor to DeepMask, kinda)

• More Deep Image Segmentation
  • U-Net: What does it work? Include derivative works
    • Ronneberger et al. 2015 and more
  • GAN Based Segmentation
  • RNN Methods
    • RNN
      • Includes Long Short Term Memory (LSTM) methods
  • Attention Based Networks (?)
  • CNN with Active Contour

• Deep Instance Segmentation:
  • SD5: Simultaneous Detection and Segmentation (SDS), 2014
  • Regional CNN (R-CNN) based methods:
    • What is R-CNN
      • Mask R-CNN (instance segmentation)
    • Multi-task Network Cascades (MNC):

• UPSNet: A Panoptic Segmentation

• Video Segmentation: show some as well that consider temporal aspect