Deep Active Learning for Interactive 3D Segmentation Of Medical Images

Vincent Groff

September 10, 2018

Table of contents

3D Image Segmentation

Medical Image Segmentation

3D Medical Image Segmentation

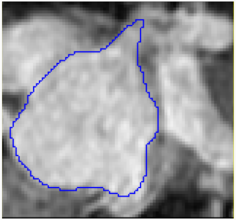

Figure 1: Segmentation of the left atrium

Main Problem: Laborious, particularly in 3D

Medical Image Segmentation

3D Medical Image Segmentation

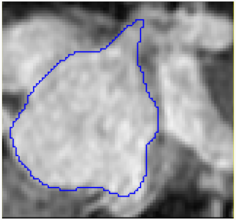

Figure 1: Segmentation of the left atrium

Main Problem: Laborious, particularly in 3D

How can we automate this?

Conditional Random Fields

Aim: Separate the set of *background* pixels (**B**) from the set of *object* pixels (**O**)

Define a cost function to be minimized over image X [1]

Conditional Random Fields

Aim: Separate the set of *background* pixels ($\mathbf{B}$) from the set of *object* pixels ($\mathbf{O}$)

Define a cost function to be minimized over image $\mathbf{X}$ [1]

$$E(\mathbf{X}) = \sum_i R_i + \lambda \sum_i \sum_j B_{ij} \tag{1}$$

- R_i - **Regional Penalty Term** - penalty for assigning pixel i to either the background ($\mathbf{B}$) or object ($\mathbf{O}$)
- B_{ij} - **Boundary Penalty Term** - penalty depending on properties of pixels i and j
- $\lambda > 0$ - a relative weighting

Conditional Random Fields

R_i - **Regional Penalty Term**

Map (0,1) probabilities to (∞, 0) penalties with negative log

R_i - **Regional Penalty Term**

Map (0,1) probabilities to (∞, 0) penalties with negative log

$$R_i = \begin{cases} -\log P(i \in O) & i \in \mathbf{O} \\ -\log P(i \in B) & i \in \mathbf{B} \end{cases} \tag{2}$$

Conditional Random Fields

B_{ij} - **Boundary Penalty Term**

Want to encourage boundaries at discontinuities in intensity

Penalize boundaries that are placed between similar pixels

Conditional Random Fields

B_{ij} - **Boundary Penalty Term**

Want to encourage boundaries at discontinuities in intensity

Penalize boundaries that are placed between similar pixels

$$B_{ij} = \begin{cases} 0 & y_i = y_j \\ \frac{1}{r_{ij}} e^{\frac{-(X_i - X_j)^2}{2\sigma^2}} & y_i \neq y_j \end{cases} \tag{3}$$

- X_i - grayscale value at pixel i
- y_i - binary label for pixel i (0 means $i \in \mathbf{B}$)
- r_{ij} - euclidean distance between pixels i and j

Graph Cuts

How do we get $P(i \in B)$ and $P(i \in O)$?

User interaction

- User gives seed points
- Probability distribution produced from seed points
- Probabilities of other pixels inferred from distribution

Works well, but...

Graph Cuts

How do we get $P(i \in B)$ and $P(i \in O)$?

User interaction

- User gives seed points
- Probability distribution produced from seed points
- Probabilities of other pixels inferred from distribution

Works well, but...

Requires object and background pixels to have distinct intensity values

Graph Cuts

Requires object and background pixels to have distinct intensity values

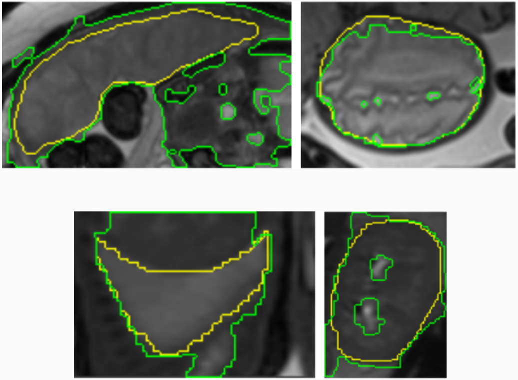

Figure 2: Grabcut algorithm in difficult cases. In *yellow* the ground truth, in *green* the algorithms guess [2]

Graph Cuts

Requires object and background pixels to have distinct intensity values

This is the case for most traditional segmentation methods

Convolutional Neural Networks

Convolutional Neural Networks

Advantages

- Spatial operations
- Learning
- Performance

Disadvantages

- Computationally expensive
- Poor generalisation to unseen types
- Need large datasets - expensive for medical imagery

Segmentation CNNs

Classification CNNs - gradually extract larger features from smaller ones

Classification CNNs - gradually extract larger features from smaller ones

U-Net CNNs - gradually re-combine those features to produce a segmentation map

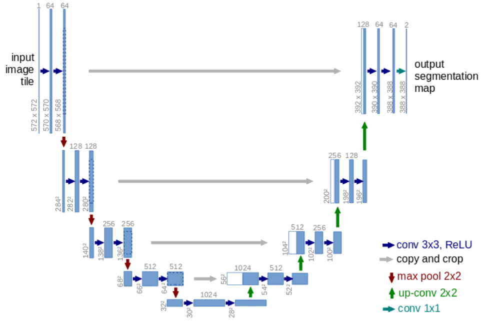

Figure 3: The U-Net architecture [3]

Segmentation CNNs

Improvements since original U-Net:

- Residual blocks
- Spatial Dropout
- Segmentation Layers

Segmentation CNNs

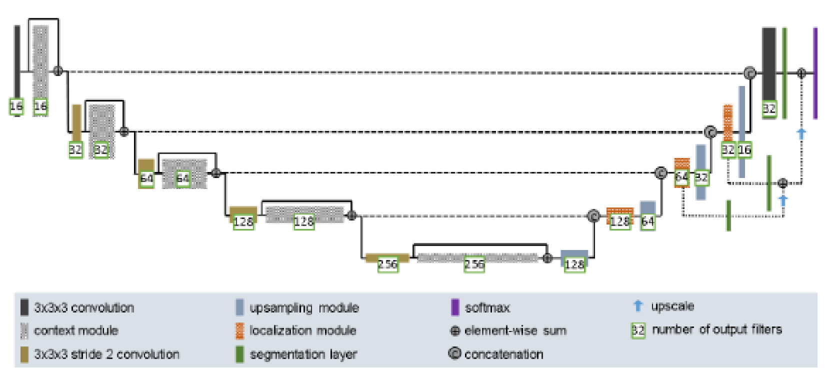

Figure 4: The Isensee et al. architecture [4]

Segmentation Layers encourage earlier layers to produce segmentations

Aims and Design

Aims

Aims:

Minimize the disadvantages of CNNs so that they can be used for 3D image segmentation

Disadvantages

1. Computationally expensive
2. Poor generalisation to unseen types
3. Need large datasets - expensive for medical imagery

BIFSeg

Bounding-box Image-specific Fine-Tuning Segmentations (BIFSeg) [2]

Combines CNN and CRF

Leverages user corrections

1. Draw bounding box around area of interest
2. CNN and CRF give initial guess
3. User provides correction scribbles
4. CNN fine-tunes using corrections and gives corrected result

BIFSeg

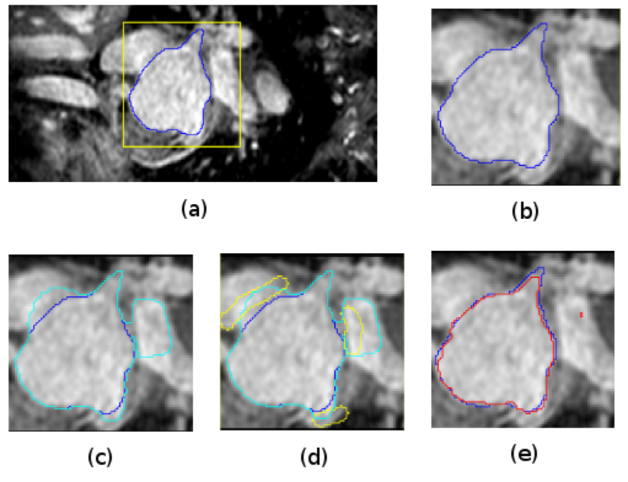

Figure 5: The steps in the BIFSeg framework. In *blue* is the ground truth, in *cyan* the initial guess, in *yellow* pixels relabelled as background and in *red* is the final result

BIFSeg

Two things going on:

- **Combining CNN and CRF**
  CNN softmax output used as probability values for calculating R_i

BIFSeg

Two things going on:

- **Combining CNN and CRF**
  CNN softmax output used as probability values for calculating R_i
- **Fine-Tuning CNN using CRF**
  1. CNN, CRF and scribbles produce segmentation
  2. CNN trained to match segmentation
  3. Repeat steps 1 and 2 n times

Fine-Tuning CNN using CRF

CNN loss function is **weighted** on a per-pixel basis during fine-tuning, such that the weight for pixel i is

$$w_i = \begin{cases} 0 & i \in \mathbf{U} \\ w & i \in \mathbf{S} \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

- **U** - set of pixels for which the current segmentation is uncertain
- **S** - set of user scribbles

BIFSeg

Fine-Tuning CNN using CRF

CNN loss function is **weighted** on a per-pixel basis during fine-tuning, such that the weight for pixel i is

$$w_i = \begin{cases} 0 & i \in \mathbf{U} \\ w & i \in \mathbf{S} \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

- **U** - set of pixels for which the current segmentation is uncertain
- **S** - set of user scribbles

Pixels in **U** are either **geodesically** near a scribble of opposite label or have P_i near 0.5

Design

BIFSeg works on unseen object types - though it needs a CNN that has learned **generalisable features**

Performs better and with less user interaction on seen object types

Design

BIFSeg works on unseen object types - though it needs a CNN that has learned **generalisable features**

Performs better and with less user interaction on seen object types

Unseen Types - Transfer learning

1. Start by using a CNN trained in generalised segmentation
2. After *n* images have been segmented, add them to a dataset
3. Load a new CNN with the generalised CNN weights and train it on the dataset
4. Repeat steps 2 and 3 until all images have been segmented

Design

Disadvantages

1. Computationally expensive
2. Poor generalisation to unseen types
3. Need large datasets

Design

Disadvantages

1. Computationally expensive ✓(Web Server)
2. Poor generalisation to unseen types
3. Need large datasets

Design

Disadvantages

1. Computationally expensive ✓(Web Server)
2. Poor generalisation to unseen types ✓(BIFSeg)
3. Need large datasets

Design

Disadvantages

1. Computationally expensive ✓(Web Server)
2. Poor generalisation to unseen types ✓(BIFSeg)
3. Need large datasets ✓(Transfer Learning)

Results

Seen object types

Evaluate performance on **seen** objects

Train left atrium images

Organ	Dice Score
Left Atrium (Initial Prediction)	0.872 ± 0.025
Left Atrium (With 6 scribbles)	0.912 ± 0.018
Liver and Spleen (Initial Prediction)	0.093 ± 0.043

Table 1: Validation scores for the generalized segmentation CNN on the organs it was trained on

Unseen Object Types

Evaluate performance on **unseen** objects

First train a generalised segmentation CNN on:

- Left Atrium
- Prostate
- Hippocampus

Unseen Object Types

Organ	Dice Score (CNN)	Dice Score (CNN+CRF)
Liver	0.516 ± 0.019	0.532 ± 0.022
Spleen	0.417 ± 0.033	0.484 ± 0.037

Table 2: Dice scores by the generalized segmentation CNN on unseen object types with and without the CRF

Organ	Dice Score (10 scribbles)	Dice Score (15 scribbles)
Liver	0.873 ± 0.029	0.891 ± 0.021
Spleen	0.892 ± 0.018	0.918 ± 0.016

Table 3: Dice scores by the generalized segmentation CNN on unseen object types, after fine-tuning with user interaction

Unseen Object Types

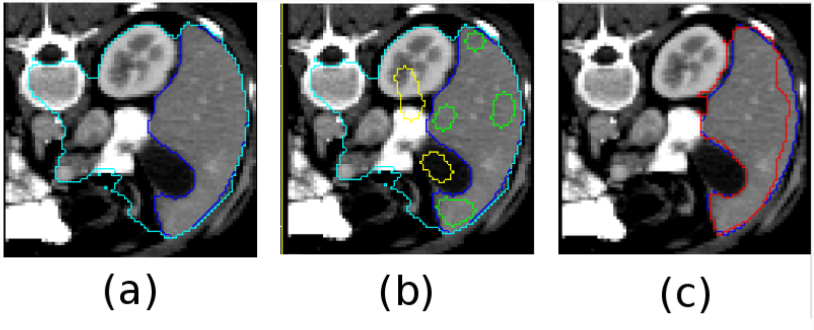

Figure 6: Segmentation of unseen object type (liver) using BIFSeg. In *blue* is the ground truth, in *yellow* are background scribbles, in *green* foreground scribbles and in *red* the final segmentation. Final Dice score over the 3D image was 0.884

Transfer Learning

Evaluate performance of **transfer learning**

Train 2 CNNs on a **small dataset** (4 training, 6 validation)

- Train one starting from scratch
- Train one starting from generalised CNN (transfer learning)

Transfer Learning - Liver

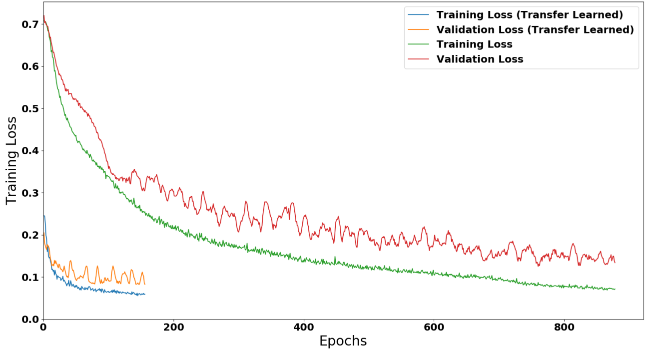

Figure 7: Comparison of transfer learning and learning from scratch on the liver with 4 training images and 6 validation images

Transfer Learning - Spleen

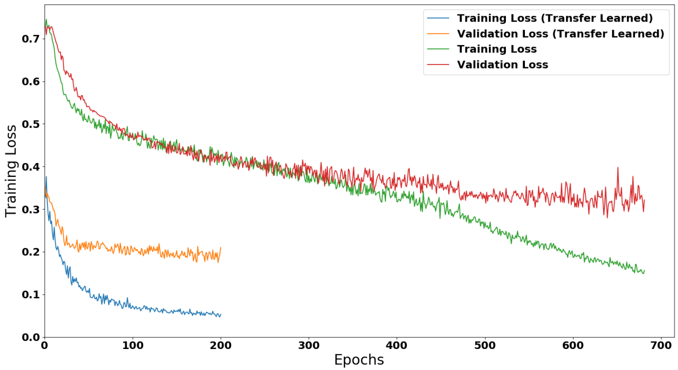

Figure 8: Comparison of transfer learning and learning from scratch on the spleen with 4 training images and 6 validation images

Further Work

Further Work

- Timed tests with users comparing different frameworks
- Temperature scaling for smoother probability distribution
- TLS
- Multiple servers with GPUs

Conclusion

Conclusion

Minimized disadvantages of CNNs

Disadvantages

1. Computationally expensive ✓(Web Server)
2. Poor generalisation to unseen types ✓(BIFSeg)
3. Need large datasets ✓(Transfer Learning)

References I

Yuri Boykov and Marrie-Pierre Jolly.
Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images.
Proceedings of International Conference on Computer Vision, I:105–112, 2001.

Guotai Wang, Wenqi Li, Maria A. Zuluaga, Rosalind Pratt, Premal A. Patel, Michael Aertsen, Tom Doel, Anna L. David, Jan Deprest, Sbastien Ourselin, and Tom Vercauteren.
Interactive medical image segmentation using deep learning with image-specific fine-tuning.
IEEE, 2018.

References II

Olaf Ronneberger, Philipp Fischer, and Thomas Brox.
U-net: Convolutional networks for biomedical image segmentation.
arXiv:1505.04597, 2015.

Fabian Isensee, Wolfgang Wick Philipp Kickingereder, Martin Bendszus, and Klaus H. Maier-Hein.
Brain tumor segmentation and radiomics survival prediction: Contribution to the brats 2017 challenge.
MICCAI BraTS 2017 proceedings, pages 100–107, 2017.

Conditional Random Fields

How do we solve it?

Represent the image as a graph with

- One node per pixel
- One source node (object)
- One sink (background)

Conditional Random Fields

How do we solve it?

Represent the image as a graph with

- One node per pixel
- One source node (object)
- One sink (background)

and

- Edges between pixels have a cost $\lambda \frac{1}{r_{ij}} e^{\frac{-(x_i - x_j)^2}{2\sigma^2}}$
- Edges between pixels and the source terminal cost $-\log P(i \in B)$
- Edges between pixels and the sink terminal cost $-\log P(i \in O)$

A minimum cost cut which severs the source from the sink will now minimize the cost function

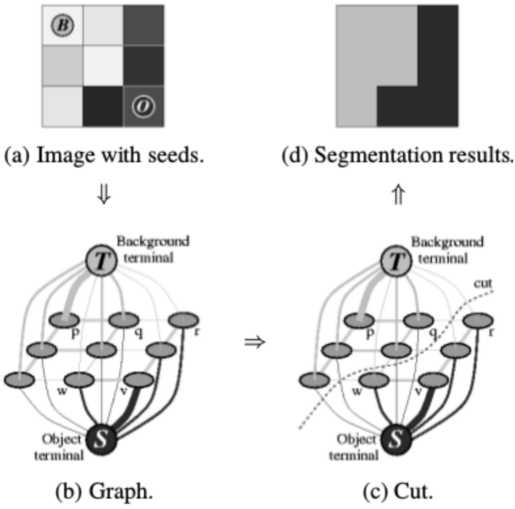

Figure 9: Minimum cost cut

Maxflow: Finding the maximal flow from source to sink

The edges that are saturated in maxflow are those in the minimal cost cut.

Conditional Random Fields

Maxflow: Finding the maximal flow from source to sink

The edges that are saturated in maxflow are those in the minimal cost cut.

Solve with **Breadth First Search**, by gradually saturating edges