LEARNING CHAN-VESE

Orhan Akal

Florida State University
Department of Mathematics
Florida, US, 32304

Adrian Barbu

Florida State University
Department of Statistics
Florida, US, 32304

ABSTRACT

Chan-Vese is a level set method that simultaneously evolves a level set surface and fits locally constant intensity models for the interior and exterior regions to minimize a Mumford-Shah integral. However, the length-based contour regularization in the Chan-Vese formulation is quite simple and too weak for many applications. In this paper we introduce a generalization of the Chan-Vese method to evolve a curve where the regularization is based on a Fully Convolutional Neural Network. We also show how to learn the curve model as a Recurrent Neural Network (RNN) using training examples. Our RNN differs from the standard ones because it has the Chan-Vese locally constant intensity model, which gives it better interpretability and flexibility.

Index Terms— level sets, Chan-Vese segmentation, convolutional neural networks, recurrent neural networks

1. INTRODUCTION

Image segmentation and related tasks, such as object segmentation and scene segmentation, have a wide range of applications, including (but not limited to) content-based image retrieval, medical imaging, autonomous driving, object detection, face recognition, etc.

While image segmentation is a generic problem, object segmentation is the problem of delineating object boundaries, such as a horse in an image or a liver in a CT scan. This problem is very important for medical imaging where it is used for delineating tumors or other pathologies, estimating the volume of a heart, a liver or a swollen lymph node, etc. Even though radiologists have been handling the aforementioned medical imaging tasks, an increasing amount of research indicates that computer vision techniques have the potential to outperform radiologists in terms of accuracy and speed.

Generic image segmentation is usually a low-level task that finds the boundary of a region purely based on the intensity difference with the neighboring regions. Object segmentation is a high-level task that aims at finding the boundary of a specific object and uses the shape of the object to eliminate distractors and to hallucinate where the boundary should be in places where it is not visible.

The Chan-Vese method [1] is a low level image segmentation method that uses a model with a constant region intensity cost and boundary length regularization in conjunction with a level set optimization algorithm to find the regions of interest.

This paper brings the following contributions:

- It introduces a generalization of the Chan-Vese formulation that allows to impose more complex shape priors in the level set framework. This formulation becomes a Recurrent Neural Network with hidden values for the intensity models inside and outside the segmented regions.

- It shows how to unwind the RNN and compute the gradients for learning the model using training examples and backpropagation and how to generate multiple initializations for training a model with better generalization.

- It presents 2D applications to liver segmentation and horse segmentation where it outperforms the original Chan-Vese, and also the RNN without the Chan-Vese formulation.

1.1. Related Work

Our work builds on the Chan and Vese [1] level set approach, which is a level set method that does not depend on edges and finds the region boundaries by alternatively minimizing an energy function and updating an intensity model. Further details will be given in Section 2.1. Our work improves this method by replacing the generic boundary length prior with a CNN that can learn a more specific object shape prior.

As in many other computer vision tasks, neural network (NN) based methods have been successfully employed to segment objects in images. For this purpose, NN based approaches essentially perform pixel-wise classification and try to predict for each pixel whether it is inside the object boundary or outside. Most of the NN based approaches mainly employ Fully Convolutional Networks [2], which were also described in Overfeat [3] for object detection and localization.

One popular NN based segmentation method is the U-net [4] which does not take a FCN approach, even though they use convolution layers. The U-net architecture has two paths, a contracting path in which convolution layers are employed to capture the context information, and a symmetrical expanding path, where deconvolution layers are used to pinpoint the exact locations of the boundary. The U-net takes a feed-forward approach while our method takes an iterative approach that refines the result in a number of iterations. In principle the

U-net model can also be used as the CNN in our model (see Figure 1), but that is subject to further exploration.

Some recent work [5]-[6] merges level sets with deep learning. Hu et al., [5] combines level sets with VGG16 [7] to segment out salient objects. The level set formulation of active contours is used in [6] along with optical flow for the task of moving object segmentation.

Recurrent Neural Networks (RNN) such as long short term memory (LSTM) networks [8] and Gated Recurrent Units (GRU) [9] are widely and mostly used for sequential data such as in speech recognition, time series predictions, etc. However, this phenomenon is changing.

Recent work employs RNNs for object segmentation or scene labeling. A first approach was introduced in [10] where a CNN was used for scene labeling, and the model was trained using backpropagation through time (BPPT), similar to ours. However, they trained a loss function that evaluates the result at each iteration, while our loss only evaluates the last iteration. More importantly, our model has hidden parameters for the intensity models inside and outside the object, which is not present in any RNN based approach.

Reseg[11], has employed 4 RNNs to perform object segmentation in which the input image is divided into non overlapping patches. At each time step, vertical and horizontal filtering layers sweep through a patch, then the system yields a new projection then updates the RNN and iterates over the next patch. Unlike [11] we do not divide the image into subpatches and instead use the whole image in the recurrence.

2. PROPOSED METHOD

The problem that we are trying to address is how to impose better shape priors in the Chan-Vese formulation, and learn these shape priors using training examples instead of setting them by hand to a predefined form. Chen et al., [12] used NNs as implicit solvers of differential equations. We propose that we can learn the shape priors not by solving a PDE, but instead by using CNNs to encode the shape prior, transforming the Chan-Vese evolution into a Recurrent Neural Network (RNN). Training the shape prior is done in an end-to-end fashion by backpropagation, but difficulties arise since all iterations of the RNN share the same weights, so special attention must be given to compute the gradient.

2.1. Chan-Vese Overview

The Chan-Vese Active contour [1] is fitted by minimizing the following Mumford-Shah energy;

$$E(C) = \int_{C_i} (I(u)-\mu_i)^2 du + \int_{C_o} (I(u)-\mu_o)^2 du + \nu|C| \quad (1)$$

where I denotes the image intensity, C is the curve to be fitted, C_i, C_o are the regions inside and respectively outside the curve C, and μ_i and μ_o are the intensity averages of image I inside and outside the curve C respectively.

A direct approach would be to derive the Euler-Lagrange equation and obtain a curve evolution equation of the form

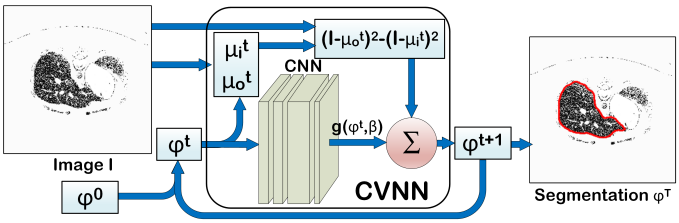

Fig. 1. Our RNN model merging CNN with Chan-Vese

$$\frac{\partial C}{\partial t} = f(\kappa)\vec{N} \quad (2)$$

where κ is the curvature of C, f is some function of the curvature, and $\vec{N}$ is the normal vector to the curve. However, such a direct approach is difficult to implement because of topological changes such as self intersection of the curve make managing the curve representation quite challenging.

The Chan-Vese approach takes a level set formulation instead. The curve C is represented as the 0 level set of a surface φ, i.e. $C = \{(x,y)|\varphi(x,y) = 0\}$. Usually $\varphi(x,y)$ is initialized as the signed distance transform of C i.e. inside the curve C $\varphi < 0$ and outside $\varphi > 0$ and the magnitude of $\varphi(x,y)$ is the distance of the point (x,y) to the closest point on curve C. Then the energy (1) is extended to an energy of the level set function φ:

$$E(\varphi) = \int (I(u)-\mu_o)^2(1-H_\epsilon(\varphi(u)))\,du + \int (I(u)-\mu_i)^2 H_\epsilon(\varphi(u))du + \nu\int \delta_\epsilon(\varphi(u))|\nabla\varphi(u)|du \quad (3)$$

where H_ϵ is the smoothed Heaviside function

$$H_\epsilon(z) = \begin{cases} 0 & \text{if } z < -\epsilon \\ 1 & \text{if } z > \epsilon \\ \frac{1}{2}[1 + \frac{z}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi z}{\epsilon})] & \text{if } |z| < \epsilon \end{cases} \quad (4)$$

and δ_ϵ is its derivative. The parameter ν controls the curve length regularization $\int |\nabla\varphi|$. When ν is small the curve (segmentation) C will have many small regions while when ν is large, the curve C will be smooth and the segmented regions will be large.

This energy is minimized alternatively by updating μ_i, μ_o

$$\mu_i^t = \frac{\int I(u)H_\epsilon(\varphi^t(u))du}{\int H_\epsilon(\varphi^t(u))du}, \mu_o^t = \frac{\int I(u)[1-H_\epsilon(\varphi^t(u))]du}{\int[1-H_\epsilon(\varphi^t(u))]du}, \quad (5)$$

then assuming μ_i^t, μ_o^t fixed the solution φ needs to satisfy the Euler-Lagrange equation:

$$\delta_\epsilon(\varphi)[\nu\,\mathrm{div}(\frac{\nabla\varphi}{|\nabla\varphi|}) + (I-\mu_o^t)^2 - (I-\mu_i^t)^2] = 0 \quad (6)$$

2.2. Our Model

Denoting $\kappa(\varphi) = \mathrm{div}\,\frac{\nabla\varphi}{|\nabla\varphi|}$ in (6), this can be accomplished with the following iterative update

$$\varphi^{t+1} = \varphi^t + \eta\delta_\epsilon(\varphi^t)\big(\kappa(\varphi^t) + (I-\mu_o^t)^2 - (I-\mu_i^t)^2\big) \quad (7)$$

The curve C can always be obtained as the 0-level set of ϕ^t, which can be done using the marching cubes algorithm.

This way topological changes are naturally handled by the level set function φ.

We generalize the Chan-Vese formulation starting with the Euler-Lagrange equation (6) and replacing the divergence term $\kappa(\varphi) = \text{div}\,\frac{\nabla\varphi}{|\nabla\varphi|}$ responsible for the length regularization with a generic function $g(\varphi, \beta)$ with parameters β

$$\delta_\epsilon(\varphi)[g(\varphi,\beta) + (I-\mu_o^t)^2 - (I-\mu_i^t)^2] = 0 \qquad (8)$$

Minimization of the new model can be done with the following iterative algorithm, with or without the $\delta_\epsilon(\varphi^t)$ factor,

$$\varphi^{t+1} = \varphi^t + \eta\delta_\epsilon(\varphi^t)(g(\varphi^t,\beta) + (I-\mu_o)^2 - (I-\mu_i)^2) \quad (9)$$

Instead of using a predefined function, we learn $g(\varphi, \beta)$ as a Convolutional Neural Network (CNN). Because of the iterative update (9), the algorithm becomes a sort of Recurrent Neural Network that takes a preset number of steps T.

In Figure 1 we can see that in each iteration our model takes φ^t as input and passes that through the CNN portion of the model which gives the shape information $g(\varphi^t, \beta)$, which is passed to the Chan-Vese update (9), along with average image intensity inside and outside of the curve C, μ_i^t, μ_o^t from Eq. (5). This way φ^{t+1} is obtained and is fed back into the next iteration of the RNN. After T iterations φ^T is thresholded to obtain the segmentation result.

2.3. Training and Backpropagation

Our model iterates a number T of times of the same update equation (9), which means that our model has parameter sharing. Then the backpropagation becomes more challenging than the models that don't have parameter sharing. Following in the footsteps of Sun-Tappen [13], we used the chain rule to obtain the gradient of the loss function L with respect to the model parameters β

$$\frac{\partial L}{\partial \beta} = \sum_{k=1}^{T} \frac{\partial L}{\partial \varphi^k}\frac{\partial \varphi^k}{\partial \beta} = \frac{\partial L}{\partial \varphi^T}\cdot\sum_{k=1}^{T}\{\frac{\partial \varphi^k}{\partial \beta}\cdot\prod_{t=k}^{T-1}\frac{\partial \varphi^{t+1}}{\partial \varphi^t}\}. \quad (10)$$

This is also illustrated in Figure 2. Using the the update (9) without the $\delta_\epsilon(\varphi^t)$, we get

$$\frac{\partial \varphi^{t+1}}{\partial \varphi^t} = 1 + \eta\Big(\frac{\partial g(\varphi^t,\beta)}{\partial \varphi^t} - 2(I-\mu_o)\cdot\frac{\partial \mu_o(\varphi^t)}{\partial \varphi^t} + 2(I-\mu_i)\cdot\frac{\partial \mu_i(\varphi^t)}{\partial \varphi^t}\Big), \text{ where} \quad (11)$$

$$\frac{\partial \mu_i(\varphi^t)}{\partial \varphi^t} = \frac{\delta_\epsilon(\varphi^t)\cdot(I-\mu_i)}{\int H_\epsilon(\varphi^t(x))dx}, \frac{\partial \mu_o(\varphi^t)}{\partial \varphi^t} = \frac{\delta_\epsilon(\varphi^t)\cdot(\mu_o - I)}{\int(1-H_\epsilon(\varphi^t(x)))dx} \quad (12)$$

and H_ϵ has been defined in Eq. (4) and δ_ϵ is its derivative. We also get

$$\frac{\partial \varphi^{t+1}}{\partial \beta} = \frac{\partial \varphi^{t+1}}{\partial g}\cdot\frac{\partial g}{\partial \beta} = \eta\cdot\frac{\partial g}{\partial \beta} \quad (13)$$

Consequently

$$\frac{\partial L}{\partial \beta} = \frac{\partial L}{\partial \varphi^T}\cdot\eta\cdot\sum_{k=1}^{T}\{\frac{\partial g(\varphi^{k-1},\beta)}{\partial \beta}\cdot\prod_{t=k}^{T-1}\frac{\partial \varphi^{t+1}}{\partial \varphi^t}\} \quad (14)$$

where $\frac{\partial \varphi^{t+1}}{\partial \varphi^t}$ is given in Eq. (11).

Training. Training is done using triplets (I_i, Y_i, M_i) containing full input images I_i with corresponding desired segmentation maps Y_i and initialization binary maps M_i. The initial level set surface φ^0 was obtained from each binary map M_i as the signed distance transform.

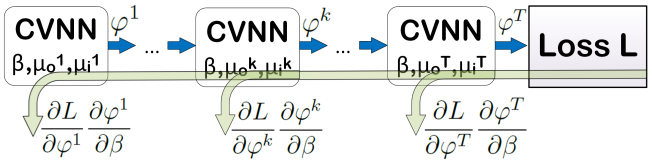

Fig. 2. For backpropagation, we unwind the model T times and compute the gradient using Eq. (14).

As loss function we used the Lorenz loss [14] for its robustness to outliers and ease of optimization

$$\ell(u) = \log(1 + \max(0, 1-u)^2) \qquad (15)$$

The overall Lorenz loss for all training examples (I_i, Y_i, M_i), $i = 1, ..., N$ is

$$L(\beta) = \sum_{i=1}^{N}\sum_{x\in Y_i}\ell(R_i(x,\beta)Y_i(x)) \qquad (16)$$

where $R_i(\cdot, \beta)$ is the output φ^T obtained by our RNN with parameters β on image I_i with initialization M_i.

When the number of iterations T is large, the loss (16) could have many local optima. To avoid getting stuck in a shallow optimum, we followed [15] and used the result of a trained RNN with fewer iterations as initialization. We therefore started with training a 1 iteration RNN, then used it as initialization for the 2-iteration one, and so on.

To be able to better handle local minima, we dynamically changed the learning rate by using an enlarged cosine wave, which is a modification of Huang et. al.'s [17] cosine annealing wave. However, unlike their formulation, the amplitude of our cosine wave is gradually enlarged.

$$\alpha_i = \frac{\alpha_1}{2}\beta^{\lfloor\frac{i\cdot r}{n}\rfloor}\left(\cos\left(\frac{\pi \mod (i-1, n/r)}{n - \lfloor\frac{i\cdot r}{n}\rfloor}\right) + 1\right) \quad (17)$$

where α_i is the learning rate for the i^{th} epoch, α_1 is the largest possible learning rate for which the gradient does not explode, β is augmentation factor for which we took $\beta = 1.25$, n and r are number of epochs to be run and number of waves respectively.

3. EXPERIMENTS

We implemented our model in MatConvNet [18] where we created new layers for the computation of μ_i, μ_o and for the Chan-Vese update (9).

CNN architecture. For CNN we used a network with two convolutional layers with 7 filters of size 7×7 with padding followed by a convolutional layer with 7 filters of size $1 \times 1 \times 7$ followed by ReLU and finally another convolutional layer with one filter of size $1 \times 1 \times 7$. The 7×7 filters used padding such that the size of the output was kept the same as the size of the input.

Multiple Initializations. In order for our model to generalize better we added for each input image I multiple initializations. One initialization was obtained the same way as it will be obtained at test time. Other 10 initializations were obtained from the ground truth Y by the following distortions:

	CV-1it	CV-2it	CV-3it	CV-10it	CV-100it	CVNN-1it	CVNN-2it	CVNN-3it
Liver Dataset	90.85	91.07	91.20	91.72	92.69	92.48	93.56	**94.04**
Weizmann horse dataset [16]	50.61	51.01	51.42	53.83	67.66	**68.72**	68.36	68.10

Table 1. Dice coefficients on the test set obtained with 4-fold cross validation. 'CV - $n$it' stands for standard Chan-Vese with n iterations, and 'CVNN - $n$it' stands for our proposed method with n iterations.

Fig. 3. Ground truth based initializations. Left: ground truth. Middle: distorted by added or punched semicircles at random border locations. Right: The middle image is corrupted by adding Gaussian noise and used as initialization for training.

First, semicircles were added or holes were punched at random locations on the boundary of Y, then Gaussian noise was added to the distorted map around the edge. This process is illustrated in Figure 3.

We trained various number of different initializations of the same image and observed that more initializations resulted in better generalization.

Dataset. We used the Weizmann horse dataset [16] and the multi-organ dataset [19], from which we used all CT volumes that were provided with a liver segmentation, for a total of 17 volumes. A total of 11 slices have been used from each volume, at location $z = 100, 105, ..., 150$, for a total of 187 images. The data was preprocessed by obtaining a rough liver segmentation with a CNN, an intensity histogram from inside the rough segmentation, finally obtaining a liver likelihood map using the histogram only. The likelihood map and the initial CNN segmentation were used as input for the methods evaluated, and results are shown in Figure 4.

Results. We report the results based on Dice coefficients [20]. Since the liver dataset is rather small, thus the results are shown with 4-fold cross validation. For the horse data we used 128 images for training and the remaining 200 images for testing. In Table 1 are showed various number of iterations of the Chan-Vese algorithm and our model CVNN results on the Liver and Horse datasets. From the results it is obvious that even one iteration of our CVNN can perform about as good as 100 iterations of Chan-Vese. Considering computation time, CV-100 it takes 0.5 seconds per image, while CVNN-3it takes 0.1 seconds per image on the liver dataset, so CVNN-3it is 5 times faster and obtains better results.

Ablation study. We performed an ablation study to see the benefit of including the intensity term $(I - \mu_o^t)^2 - (I - \mu_i^t)^2$ in Eq. (9), and what would happen if we added the image intensity as another channel in the CNN, obtaining $g(\varphi, I, \beta)$. The results for the liver data are shown in Table 2.

It is clear that without the intensity model the accuracy that CVNN is no better than 1 iteration of CV. We can obtain a regular RNN by removing the intensity model and adding the image I as a channel for the CNN. We see that this RNN does better than without the intensity model, but no better than a 10-iteration Chan-Vese. Finally, using the image as a channel for CNN and adding back the intensity model gives another rise of 2% in accuracy, but still not as good as our proposed method that only uses the CNN for the shape.

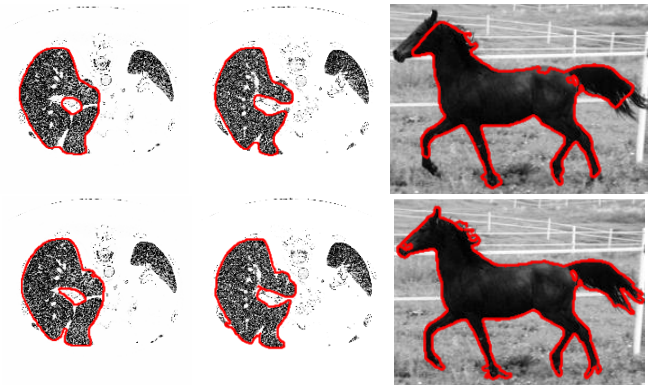

Fig. 4. Segmentation results. Top: Chan-Vese with 100 iteration. Bottom: CVNN with 3 iterations.

	Update term in Eq. (9)	Dice
Proposed	$g(\varphi,\beta)+(I-\mu_o^t)^2-(I-\mu_i^t)^2$	**94.04**
No intensity model	$g(\varphi,\beta)$	89.74
Regular RNN	$g(\varphi,I,\beta)$	91.46
Image I channel in CNN	$g(\varphi,I,\beta)+(I-\mu_o^t)^2-(I-\mu_i^t)^2$	93.42

Table 2. Ablation study of the importance of the intensity term $(I - \mu_o^t)^2 - (I - \mu_i^t)^2$ in Eq. (9) and whether using the image I as a channel in the CNN is useful or not.

4. CONCLUSION

In this paper we proposed a generalization of the Chan-Vese method that allows to learn a more complex shape model using a CNN instead of the generic boundary length prior from the original formulation. For this purpose we replaced the curvature term from the Chan-Vese update with a CNN and showed how to train it by back-propagation through time. We also showed how to generate multiple initializations for training in order to improve generalization of the obtained model.

Experiments showed that one iteration of the proposed approach can obtain comparable results with 100 iterations of Chan-Vese, and more iterations further improve accuracy.

In the future, we plan to extend the proposed approach to 3D data, where the Chan-Vese method is very computationally intensive. We also plan to extend the approach to multi-organ segmentation.

5. REFERENCES

[1] T Chan and L Vese, "Active contours without edges," *IEEE Transactions on Image Processing*, vol. 10, pp. 266–277, 2001.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 3431–3440.

[3] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*. Springer, 2015, pp. 234–241.

[5] Ping Hu, Bing Shuai, Jun Liu, and Gang Wang, "Deep level sets for salient object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2300–2309.

[6] Ping Hu, Gang Wang, Xiangfei Kong, Jason Kuen, and Yap-Peng Tan, "Motion-guided cascaded refinement network for video object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1400–1409.

[7] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[8] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[10] Pedro HO Pinheiro and Ronan Collobert, "Recurrent convolutional neural networks for scene labeling," in *ICML*, 2014, number EPFL-CONF-199822.

[11] Francesco Visin, Marco Ciccone, Adriana Romero, Kyle Kastner, Kyunghyun Cho, Yoshua Bengio, Matteo Matteucci, and Aaron Courville, "Reseg: A recurrent neural network-based model for semantic segmentation," in *CVPR Workshops*, 2016, pp. 41–48.

[12] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems*, 2018, pp. 6571–6583.

[13] Jian Sun and MF Tappen, "Learning non-local range markov random field for image restoration," in *CVPR*. IEEE Computer Society, 2011, pp. 2745–2752.

[14] Adrian Barbu, Yiyuan She, Liangjing Ding, and Gary Gramajo, "Feature selection with annealing for computer vision and big data learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 2, pp. 272–286, 2017.

[15] Adrian Barbu, "Training an active random field for real-time image denoising," *IEEE Transactions on Image Processing*, vol. 18, no. 11, pp. 2451–2462, 2009.

[16] Eran Borenstein and Shimon Ullman, "Class-specific, top-down segmentation," in *ECCV*, 2002, pp. 109–122.

[17] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger, "Snapshot ensembles: Train 1, get m for free," *arXiv preprint arXiv:1704.00109*, 2017.

[18] Andrea Vedaldi and Karel Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 689–692.

[19] Holger R Roth, Le Lu, Amal Farag, Hoo-Chang Shin, Jiamin Liu, Evrim B Turkbey, and Ronald M Summers, "Deeporgan: Multi-level deep convolutional networks for automated pancreas segmentation," in *MICCAI*, 2015, pp. 556–564.

[20] Lee R Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.