This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

End-to-End Learned Random Walker for Seeded Image Segmentation

Lorenzo Cerrone, Alexander Zeilmann, Fred A. Hamprecht
Heidelberg Collaboratory for Image Processing
IWR, Heidelberg University, Germany
{lorenzo.cerrone, alexander.zeilmann, fred.hamprecht}@iwr.uni-heidelberg.de

Abstract

We present an end-to-end learned algorithm for seeded segmentation. Our method is based on the Random Walker algorithm, where we predict the edge weights of the underlying graph using a convolutional neural network. This can be interpreted as learning context-dependent diffusivities for a linear diffusion process. Besides calculating the exact gradient for optimizing these diffusivities, we also propose simplifications that sparsely sample the gradient and still yield competitive results. The proposed method achieves the currently best results on a seeded version of the CREMI neuron segmentation challenge.

1. Introduction

Image segmentation is the task of partitioning an image into regions that are meaningful for a given task. Seeded segmentation is a popular semi-supervised variant, where an oracle annotates one or more pixels per region with an instance label, such that all seeds in a region share the same label.

Most seeded segmentation pipelines involve two parts: a first step that predicts, for each pair of adjacent pixels, if or not they are likely part of the same region; and a second step that ingests these predictions as well as the seed locations and then infers the precise extent of each region. The second step is often cast as inference in an undirected probabilistic graphical model, either discrete [6] or continuous [13].

If no prior information on the nature of the images to be segmented is available, the first step – boundary estimation – is of necessity generic. It is then possible to use or train some state of the art edge detection method (e.g. [19]) and choose the parameters of the second step – inference – by hand.

But since dense ground truth is required for the training of a powerful edge detection method in any case, one may also go further and train the two steps *jointly*. Such a *structured* learning is more general than the naive approach of training a CNN to predict good boundary indicators, oblivious to what these will be used for; and to subsequently and independently optimize the parameters of the inference step.

For the second, inference, step, we choose a Conditional Gaussian Markov Random Field with only seed fidelity as the unary term – also known as Random Walker algorithm [13] – and here show how to train that inference scheme jointly with a deep convolutional neural network (CNN).

Our choice of a Gaussian Markov Random Field is motivated by the following features:

- Less sensitive to noisy edge weights than the Watershed [11] which is fast and hence popular in seeded segmentation, but also susceptible to outliers based on its use of the (max, min)-semiring [4].
- More amenable to differentiation than the purely combinatorial graph cuts problem [6].
- More computationally efficient than more expressive generalizations involving higher-order potentials [22].

More specifically, we make the following contributions:

1. We demonstrate end-to-end structured learning of a pipeline consisting of a deep neural network that predicts edge weights, which are subsequently used in a seeded segmentation by linear diffusion, i.e., with the Random Walker algorithm.
2. We calculate the exact gradient of our objective function and propose a sparse sampling strategy of the gradient to reduce computation time.
3. We evaluate the algorithm on a seeded version of the MICCAI Challenge on Circuit Reconstruction from Electron Microscopy Images (CREMI) [10]. Here, the proposed method outperforms unstructured training of

a CNN combined with the Random Walker algorithm, and defines a new state of the art.

4. We provide the source code as PyTorch package on `https://github.com/hci-unihd/pytorch-LearnedRandomWalker`, allowing general use of the proposed Learned Random Walker.

2. Related Work

Diffusion processes are attractive because they engender nontrivial global behavior as a consequence of trivial local interactions. As such, they have a rich history in machine learning and computer vision.

Seeded segmentation could be seen as an instance of semi-supervised segmentation, which has been explored in conjunction with Gaussian Markov Random Fields (GMRF) on arbitrary graphs in [31], though without the supervised learning of edge weights as discussed here.

In computer vision, linear diffusion has been proposed for seeded segmentation in the seminal work of Grady [13]. In semantic segmentation and image-valued regression tasks, Tappen et al. [23] have shown how to learn context-dependent potentials for GMRFs. These authors did not explore seeded segmentation, and the number of parameters learned was in the hundreds, not hundreds of thousands as in our case. Jancsary et al. [14] have pioneered the use of flexible nonparametric classifiers to determine the potentials in a Conditional GMRF. However, they opted for ensembles of decision trees and had to estimate the impact of each possible split on the loss function, making a number of approximations necessary.

In recent work, Vernaza and Chandraker [27] have proposed a first-order approximation of the derivative for back-propagation. We show in section 4.5 that this approximation is not well-behaved for the kind of sparse seeds considered here.

Our algorithm can also be seen from the viewpoint of coupling a deep network with an undirected probabilistic graphical model. In contrast to [7, 30] we use a GMRF that makes inference easier. End-to-end learning of a GMRF was also proposed in [5], though these authors do not solve the inference problem exactly as we do; instead, they unroll a few steps of gradient descent into extra layers of their neural network. Our inference problem can be seen as convex quadratic program. Recent work proposes quadratic programming as a generic neural network layer [2]. However, the paper and implementation do not account for the kind of sparsity that characterizes our formulation, and scales only to a few hundred random variables, where we use tens of thousands. Indeed, the solution of huge sparse systems of the kind discussed here is possible in almost linear time [25].

Other important seeded segmentation strategies are based on discrete Markov Random Fields [6] that do however not admit the differentiability we crucially need; or on shortest paths similarities that can be computed even more efficiently than the diffusions we use [3]. However, the use of (single) shortest paths to measure the dissimilarity from each pixel to every seed is not as robust as the averaging over many paths that diffusion processes implicitly accomplish [28].

Such fragility is especially pronounced for watershed-type methods that base dissimilarities on a minimax criterion [9]. Even so, such methods have been used successfully in biomedical [18] and other applications. Recent work has sought to mitigate this limitation by learning the edge weights for watershed seeded segmentation in a structured fashion, just as we do, and moreover making the algorithm adaptive [29], the cost for the latter being a somewhat involved recurrent neural net formulation. The experiments in section 4.3 show that we can supersede the learned watershed in spite of our simpler architecture.

Remarkably, all of the above inference schemes for seeded segmentation – graph cuts, random walker / linear diffusion and geodesic distances – emerge as special cases of a unifying optimization problem [8].

3. Methods

3.1. Mathematical Background

To make this paper self-contained, we derive the linear system (1) which is solved in the Random Walker algorithm. While Grady [13] deduced the system by minimizing a discretized Dirichlet energy, we derive it directly from the discretized Laplace equation.

The Random Walker algorithm models seeded segmentation with $|\mathcal{L}|$ distinct categories of seeds as follows: each category or label $a \in \mathcal{L}$ is associated with an infinitely large reservoir. This reservoir is coupled to the pixels that have been marked with label a. From this reservoir, the label a can diffuse across the image. The pixels associated with all other labels $a' \in \mathcal{L} \setminus \{a\}$ act as sinks for label a. That is, the concentration of a label a is one at the pixels that have been marked with this label; it is zero at the locations of all other seeds; and its "concentration" in-between is given by the stationary state of the linear diffusion equation. Importantly, local diffusivities are informed by local image appearance: Random Walker assumes high diffusivity within ground truth segments and low diffusivity across segments.

In the continuous setting, diffusion is usually formulated as

$$u = f \text{ on } \partial\Omega,$$
$$\Delta u = 0 \text{ in } \Omega,$$

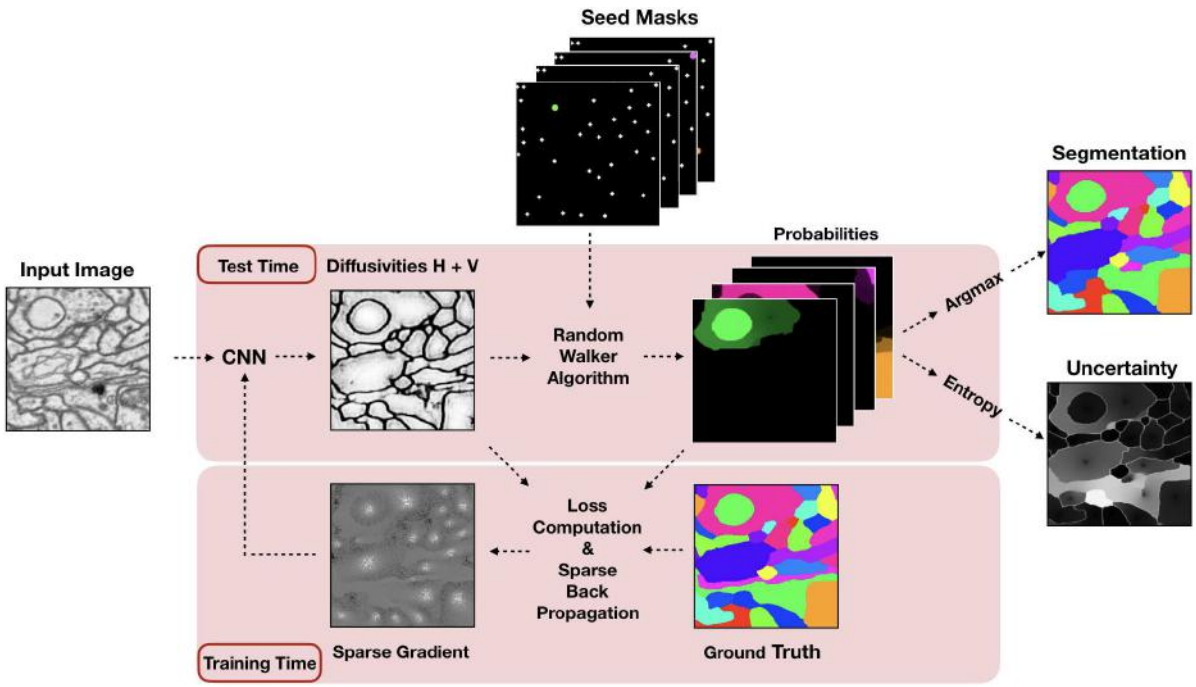

Figure 1. Illustration of the Learned Random Walker pipeline at training and test time. **Top:** forward pass, the input image is mapped by a CNN to an undirected edge-weighted graph. A probability map is computed for each object using the Random Walker algorithm. At test time, a segmentation and an uncertainty map are computed from the probabilities. **Bottom:** backward pass of the training. The dense derivative of the Random Walker algorithm is too expensive to compute, instead, a sampling scheme is used and only a very sparse Jacobian is passed to the CNN.

where we prescribe the value of a function u on the boundary of the domain Ω and require that the Laplace operator vanishes in the interior of the domain.

To carry these ideas over to the domain of discrete images, we consider an image as a connected undirected edge-weighted graph $G = (V, E)$ with adjacency relation $i \sim j \iff (i,j) \in E$ and edge weights $w_e \in \mathbb{R}_+$ for every edge $e \in E$. The set of vertices V is partitioned into unmarked vertices U and marked vertices M, the seed pixels. By L we denote the graph Laplacian which is defined as

$$L_{i,j} = \begin{cases} -w_{i,j} & \text{if } i \sim j \\ \sum_{k \sim i} w_{i,k} & \text{if } i = j \\ 0 & \text{else,} \end{cases}$$

i.e. $L = D - A$ with the degree matrix D and the adjacency matrix A of G. We consider the vertices ordered in such a way that the marked pixels appear above and left of the unmarked pixels in L:

$$L = \begin{pmatrix} L_M & B \\ B^T & L_U \end{pmatrix}.$$

We define a row stochastic matrix $Z \in [0,1]^{|V| \times |\mathcal{L}|}$ by

$$Z_{i,a} = \text{probability of vertex } i \text{ having label } a \in \mathcal{L}.$$

This matrix Z is called assignment matrix in recent literature [1]. We assume that the rows in Z are sorted in the same way as in L, i.e. the marked rows are above the unmarked rows. Thus, we can partition the matrix into a marked (known) part Z_M and an unmarked (wanted) part Z_U. With these notions, the continuous diffusion equation can be extended to the discrete setting in the following way:

$$\begin{pmatrix} L_M & B \\ B^T & L_U \end{pmatrix} \begin{pmatrix} Z_M \\ Z_U \end{pmatrix} = \begin{pmatrix} * \\ 0 \end{pmatrix},$$

i.e. the Laplace matrix multiplied by the assignment matrix is 0 on the unmarked vertices and not prescribed on the marked ones — indicated by the $*$ on the right-hand side. This is similar to Δu not being required to have a certain value on $\partial\Omega$. Z_M is set to user-specified values. Multiplying out yields the following extremely sparse linear system, which is at the heart of the Random Walker algorithm:

$$L_U Z_U = -B^T Z_M. \tag{1}$$

Since L_U is invertible, the solution Z_U of the linear system exists and is unique.

In summary, solving the above linear system gives the probability, for each pixel, to be associated with a given seed. These probabilities are typically fractional, which

turns out to be a great advantage: It allows the definition of meaningful spatially resolved uncertainty measures such as the entropy at each pixel,

$$H_v = -\sum_{a\in\mathcal{L}} Z_{v,a} \log\left(Z_{v,a}\right) \tag{2}$$

See Figure 2 for an illustration.

The same kind of measure would not be meaningful for graph cut type approaches, whose unimodularity guarantees integral all-or-nothing solutions. For graph cut results, the entropy would be constant across the entire image.

In the Random Walker, the final labeling is obtained from the resulting assignment matrix by a winner-take-all selection of the the label with maximal probability for each vertex $v \longmapsto \arg\max_{a\in\mathcal{L}} Z_{v,a}$.

3.2. Structured Learning of a CNN to Predict Edge Weights for the Random Walker

To find appropriate edge weights, early work [13] used simple parametric estimators of local image contrast, such as exponentiated intensity differences. Later work has used flexible nonlinear estimators, such as random forests, to estimate edge weights [14]. We want to use gradient descent to train weight estimators in a structured fashion in the sense of [26], only for seeded segmentation and for a more flexible and powerful class of estimators.

When solving equation (1) exactly, structured learning amounts to the following optimization problem: the minimization of loss l with respect to the neural network parameters Θ

$$\arg\min_{\Theta} l\left(Z_U^*, -L_U(I;\Theta)^{-1}\cdot B(I;\Theta)^T\cdot Z_M\right) \tag{3}$$

based on ground truth Z_U^*. Here, we have made explicit the dependence of L_U and B^T on the edge weights w, which in turn depend on image I and network parameters Θ.

To solve the optimization problem we use gradient descent, for which we compute the gradient

$$\frac{\partial l\left(Z_U^*, Z_U\right)}{\partial\Theta} = \frac{\partial l\left(Z_U^*, Z_U\right)}{\partial Z_U}\frac{\partial Z_U}{\partial w}\frac{\partial w}{\partial\Theta}. \tag{4}$$

The first and third term on the right hand side are standard: the partial derivative of the loss function with respect to a candidate solution, and of the neural network with respect to its parameters, respectively. The remaining tensor is cumbersome due to its extremely large dimensions: the partial derivative of the probability of (typically:) dozens of seeds $|\mathcal{L}|$ in millions of pixels $|U|$ with respect to millions of edges $|E|$ make for a formidable object $\partial Z_U/\partial w \in \mathbb{R}^{|U|\times|\mathcal{L}|\times|E|}$. One way to evaluate this expression is to take the derivative with respect to w of the linear system (1):

$$\frac{\partial L_U Z_U}{\partial w} = -\frac{\partial B^T Z_M}{\partial w}$$

Since the probabilities at the marked pixels do not depend on the edge weights we have $\partial Z_M/\partial w = 0$ and obtain with the product rule the following tensor-valued linear system, whose solution is $\partial Z_U/\partial w$:

$$L_U\frac{\partial Z_U}{\partial w} = -\frac{\partial L_U}{\partial w}Z_U - \frac{\partial B^T}{\partial w}Z_M \tag{5}$$

This equation is a combined representation of $|\mathcal{L}||E|$ usual linear systems with matrix L_U and changing right hand sides. These right hand sides are easy to calculate as $\partial L_U/\partial w$ and $\partial B^T/\partial w$ are very sparse and constant with respect to w, i.e. they do not change during the gradient descent.

3.3. Simplifications for Calculating the Gradient

On the other hand, computing the huge rank 3 tensor $\partial Z_U/\partial w$ requires, in each gradient descent step, solving the tensor-valued linear system (5), which is computationally expensive. As the tensor is only used in the tensor multiplication

$$\frac{\partial l(Z_U^*, Z_U)}{\partial w} = \frac{\partial l(Z_U^*, Z_U)}{\partial Z_U}\frac{\partial Z_U}{\partial w} \in \mathbb{R}^{|\mathcal{L}|}$$

we can make a few simplifying approximations:

Sparse Gradient Instead of calculating the entire gradient tensor $\partial l(Z_U^*, Z_U)/\partial w$ we randomly select $n \ll |E|$ edges for which we solve the corresponding linear systems and set the entries corresponding to other edges to zero. This approach can be seen as a stochastic gradient descent algorithm.

We have tried more sophisticated strategies, including ones based on the label entropy or ones that concentrate on misclassified pixels. However, none of them performed significantly better than the simple and parameter-free uniform sampling of edges, so this was used in all experiments shown.

Gradient Pruning In equation (4), the entries in the huge 3-tensor are multiplied, and hence modulated, with entries from $\partial l(Z_U^*, Z_U)/\partial Z_U$. Inspired by [16], for a given edge (ij) we only compute the contributions from label

$$\arg\max_a \left|\left(\frac{\partial l(Z_U^*, Z_U)}{\partial Z_U}\right)_{i,a}\right|.$$

Taken together, these two approximations reduce the size of the tensor from $|U|\times|\mathcal{L}|\times|E|$ to $|U|\times 1\times n$, i.e instead of solving $|\mathcal{L}||E|$ linear systems of size $|U|$ we only solve n linear systems of size $|U|$. As an example, we choose a 4-connected graph with 128×128 vertices and 10 labels. Fig 3 shows average backpropagation run-times. We found $n = 1024$ to be the best compromise between ($\approx$ 15-fold) speed-up and accuracy.

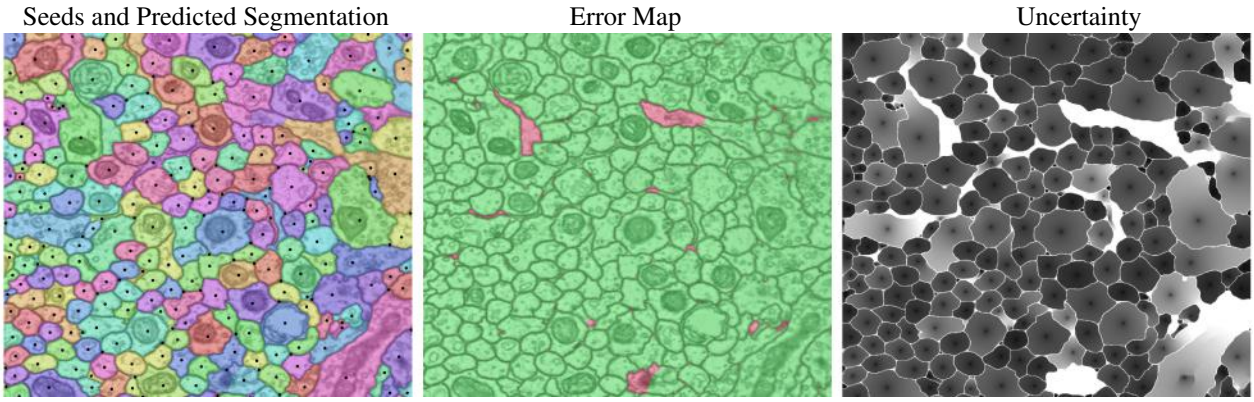

Figure 2. The assignment matrix produced by the Random Walker algorithm can be converted to a segmentation (left picture, including the seeds we used in the Random Walker algorithm). The center image illustrates the wrongly labeled pixels (marked red). The uncertainty of the labeling is illustrated via the entropy (white indicates high, black indicates small uncertainty).
We found that wrongly segmented regions usually have high uncertainty. However the converse is not true in general: High uncertainty does not necessarily indicate a wrong segmentation.

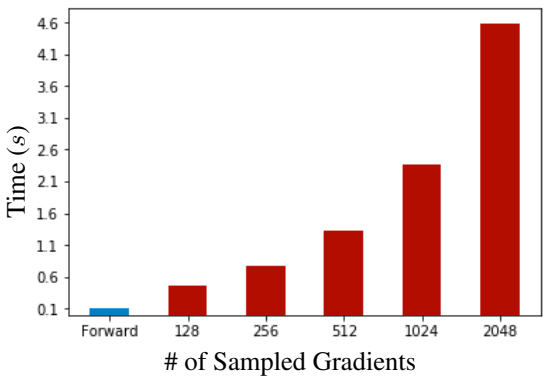

Figure 3. Run time comparison for different number of gradients sampled. For comparison, a complete backpropagation step ($n = 16384$) takes $37s$. All results reported are for a (i7-6700, 3.4 GHz) CPU machine.

4. Experiments and Results

4.1. Pipeline Overview

The proposed method is agnostic to CNN architecture; we employ a Convolutional-Deconvolutional network inspired by [17]. The segmentation is conducted in 2D, but the network accepts a 3D volume as input (3 slices along z) and predicts edge weights for the central slice only. In our implementation we used the 4-connected graph, so on average we only need two edge weights per pixel, one horizontal and one vertical. The network details are shown in the supplementary material. The network outputs edge weight images at half the original resolution. This speeds up both inference and backpropagation and reduces the memory footprint without impacting the final result. The assignments are then scaled to the original size using bilinear interpolation before computing the loss.

As loss function we choose the Cross Entropy loss (CE) on the assignment matrix, defined as

$$\mathrm{CE}(Z^*, Z) = -\frac{1}{|V|} \sum_{i \in V} \sum_{a \in \mathcal{L}} Z^*_{i,a} \log\left(Z_{i,a}\right) \tag{6}$$

where Z is our calculated assignment matrix and Z^* the ground truth. We also tried employing the Mean Squared Loss and the Dice Loss. While results are comparable, CE performed slightly better. In addition, we used an unstructured CE loss on the weights and for regularization an ℓ^2 weight decay on the network parameters Θ. In summary, the loss reads

$$l\left(Z^*, Z(\Theta)\right) = \mathrm{CE}(Z^*, Z(\Theta)) + \alpha \mathrm{CE}\left(w^*, w(\Theta)\right) + \frac{\gamma}{2} \|\Theta\|_2^2 . \tag{7}$$

where w^* are the ground truth edges obtained from Z^*, $\alpha = 10^{-2}$ and $\gamma = 10^{-5}$.

The network is trained on patches of size 256×256. We use mini-batches of size 10 and train for a total of 10 epochs. Before the structured training, the CNN is pre-trained using only the side loss on the same data, with mini-batches of size 10 and for 2 epochs. As optimizer we use Adam [15].

4.2. Seeded CREMI 2D Segmentation

In our experiments we work on the data from the MICCAI Challenge on Circuit Reconstruction from Electron Microscopy Images (CREMI) [10]. The challenge has been designed to measure automated segmentation accuracy in connectomics. There, the aim is to trace a myriad neural processes in serial section electron microscopy images to ultimately produce a wiring diagram of the brain. While

the accuracy of automated methods is improving continuously [12], they do not yet reach a precision that would allow immediate downstream processing. Also, automated methods need much training data, and so seeded segmentation of connectomics data is of interest both for ground truth generation and for proofreading of automated predictions.

The CREMI dataset is composed of three volumes from adult Drosophila melanogaster (common fruit fly) brain tissue, each of size $1250 \times 1250 \times 125$ pixels. The three volumes are very different in appearence: While the neurites in CREMI A are mostly homogeneous in size and shape, the two other volumes (CREMI B and C) are more challenging, with cells that have jagged boundaries and large variations in size and shape. We use the first 50 slices in z for testing and the last 75 for training.

For our seeded segmentation, we assume that an oracle provides precisely one seed per segment. Here we implement an oracle by automatically computing the seeds from the ground truth. For each segment we place one seed at a random location, but with a reasonable distance to any boundary. Because using seeds from an oracle is a strong assumption we cannot directly compare our approach to the unseeded algorithms competing in the CREMI challenge. Therefore, we evaluate the performance of our end-to-end algorithm by comparing it to the following pipelines, which are also seeded approaches:

Standard Random Walker: We slightly modified our network to directly predict boundary probability maps. For this, we trained the CNN on the same dataset for a total of 10 epochs, while using the Dice loss and mini-batches of size 10. Subsequently, we compute the segmentation using the standard Random Walker algorithm given in [13]. The algorithm has a single hyperparameter β, which we tune optimally. As for the Learned Random Walker we downsampled the graph by a factor of 2 to reduce computational footprint.

Watershed: For the Watershed algorithm, we used the same methodology as for the standard Random Walker algorithm to predict boundary probability maps. The only difference is in the output size. Indeed, the Watershed algorithm is very efficient to compute and thus we do not downsample.

Learned Watershed: Lastly, we compared our results with the numbers published in [29].

All segmentations are evaluated using the official CREMI metrics: Variation of Information (VOI) and Adapted Rand Error (ARAND). VOI [20] is the conditional entropy of the predicted and the ground truth segmentation: $\text{VOI} = \text{VOI}_{\text{split}} + \text{VOI}_{\text{merge}} = H(Z|Z^*) + H(Z^*|Z)$ where H is the conditional entropy. ARAND is the complement of the Adjusted Rand Index [21]: $\text{ARAND} = 1 - \text{AdjRAND}$. According to the challenge, we used a tolerance of two pixels around the boundary.

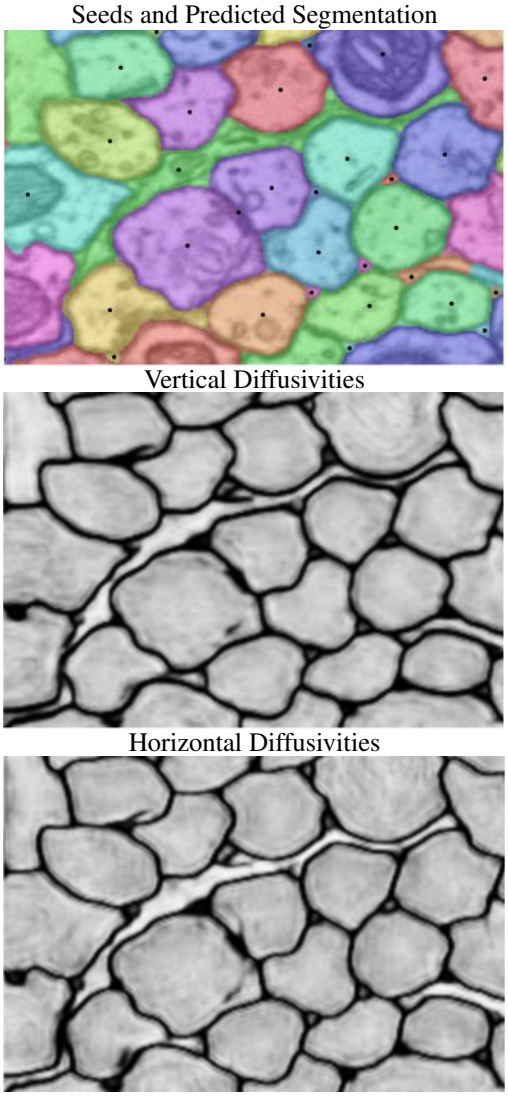

Figure 4. Learned Random Walker qualitative behavior in narrow corridors. Top: seeds and the resulting Learned Random Walker segmentation. Middle and bottom: The learned horizontal and vertical diffusivities are much stronger inside a corridor than near its perimeter. The horizontal diffusivity (bottom image) is a little larger because the corridor itself is horizontal.

4.3. Seeded CREMI 2D Segmentation Results

We show all results in Table 1. Quantitatively, the Learned Random Walker with its structured training outperformed the Random Walker algorithm with unstructured training in every experiment. Furthermore, the Learned Random Walker gave the best results in two of the three volumes, CREMI A and B, whereas Watershed obtained the

VOI	WS	LWS	RW	LRW
Cremi A	0.075 ± 0.024	—	0.177 ± 0.015	$\mathbf{0.062 \pm 0.021}$
Cremi B	0.211 ± 0.080	—	0.362 ± 0.086	$\mathbf{0.193 \pm 0.089}$
Cremi C	$\mathbf{0.209 \pm 0.074}$	—	0.421 ± 0.091	0.232 ± 0.081
Total	0.165 ± 0.091	0.376 ± 0.034	0.320 ± 0.127	$\mathbf{0.162 \pm 0.102}$
ARAND	WS	LWS	RW	LRW
Cremi A	0.016 ± 0.010	—	0.042 ± 0.008	$\mathbf{0.011 \pm 0.009}$
Cremi B	0.049 ± 0.044	—	0.153 ± 0.078	$\mathbf{0.045 \pm 0.044}$
Cremi C	$\mathbf{0.053 \pm 0.045}$	—	0.163 ± 0.066	0.061 ± 0.038
Total	$\mathbf{0.039 \pm 0.037}$	0.082 ± 0.001	0.239 ± 0.146	$\mathbf{0.039 \pm 0.040}$

Table 1. Quantitative comparison of Seeded Watershed on a good boundary probability map, Learned Watershed [29], Random Walker on a good boundary probability map and Learned Random Walker on the seeded CREMI challenge. Lower is better.

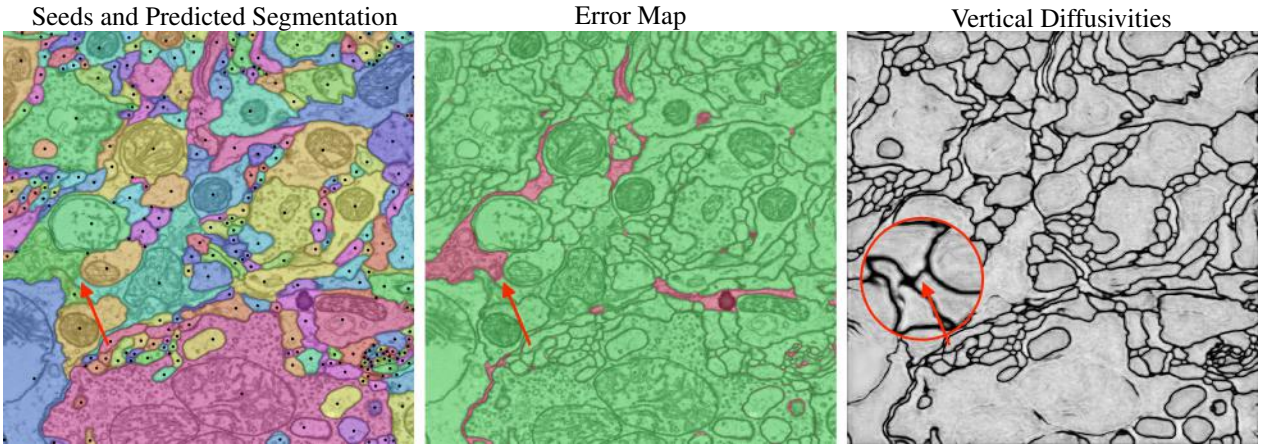

Figure 5. Typical Learned Random Walker fail on CREMI C. The red circle in the right image indicates a false positive boundary detection. In this part of the data, the Learned Random Walker tends to hallucinate boundaries.

best results on CREMI C.

Indeed, volumes B and C are less well suited for the Random Walker algorithm, which has intrinsic shrinkage bias, whereas watershed does not [24]. We observe that the structured learning seems to overcome this handicap: While the standard Random Walker algorithm tends to suffer in narrow corridors or near narrow funnels, the structured training helped the network predict stronger diffusivities inside this kind of regions, see Figure 4. We can assume the network learns a strong shape prior on how these kind of regions look like.

On the other hand, we observed a tendency to false positive boundary hallucination with the pure structured training of the network. An example is shown in Figure 5. Moreover, in our experiments, we observed that a CNN trained directly on a boundary detection task generally performs better on more semantic tasks, like distinguishing inner structures or mitochondria from true cell boundaries.

4.4. Qualitative Experiments on Arabidopsis Thaliana Ovules

In Figure 6 we show a qualitative comparison between LRW and Watershed in a different imaging domain. The 3D dataset (courtesy of Schneitz Lab, TU Munich, Germany) consists of several Arabidopsis thaliana ovules, acquired using confocal laser scanning microscopy. The ovules are the major female reproductive organ of the plant. The Arabidopsis ovule serves as a prominent model for quantitative morphogenetic studies which require highly accurate segmentation of cells. The experimental setup used is identical to sec. 4. As shown in Figure 6, the results agree qualitatively with our experiments on CREMI.

4.5. Sampling Strategy vs. Approximate Back-Propagation

Different approaches have been proposed for backpropagating gradient in Laplacian systems. Very recently, a first order approximation of the true derivative has been used

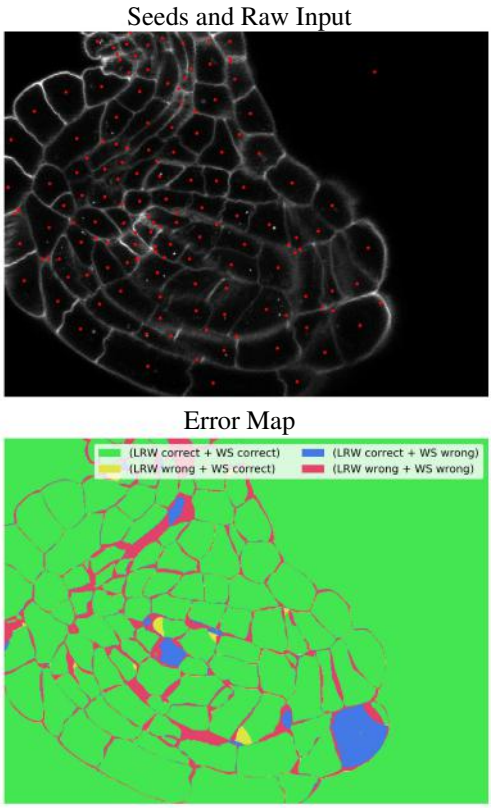

Figure 6. Qualitative comparison between Learned Random Walker and Watershed applied on confocal microscopy data. Blue and yellow areas represent the regions where the Learned Random Walker outperformed Watershed and vice versa, respectively.

in [27] for semantic segmentation. Their approach has the conspicuous advantage that it requires solving only one system of linear equations. We set up a simple experiment to show the differences between the Learned Random Walker backpropagation and the one presented in [27].

We tried to perfectly regress a single image labelling in two different scenarios: Extended seeding, where we use large brush strokes on the image as seeds; and sparse seeds, where each region was seeded in a single pixel only. Instead of using a neural network to predict edge weights, we over-parametrize the system by using one parameter per edge; and we use gradient descent to update all these parameters until convergence.

For this example we used a sample image from the CREMI dataset [10] and the same methodology as in section 4. The quantitative results are presented in Table 2.

We find that the Learned Random Walker can fit the ground truth both under extended and sparse seeds; whereas the first-order approximation to backpropagation gives satisfactory results only with extended but not with sparse seeds. Qualitatively, we observed that the first-order approximation breaks down far from any given seeds Figure 7. Moreover, we find that the sampling strategy introduced in sec. 3.3 has little effect on the reconstruction accuracy.

ARAND	Extended Seeding	Sparse Seeding
First order approx. [27]	0.04	0.32
LRW, 250 samples (ours)	0.01	0.03
LRW, no sampling	0.01	0.01

Table 2. Quantitative comparison between Learned Random Walker presented here and first order approximation from [27]. The ARAND metric is defined in Section 4.2, and lower is better.

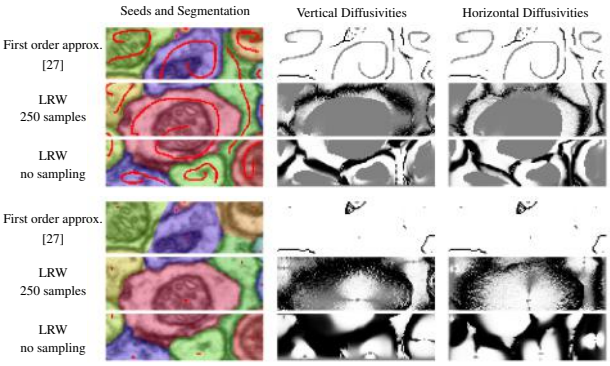

Figure 7. Qualitative comparison between Learned Random Walker presented here and first order approximation from [27]. We can observe that neither the sparse seeding nor our sampling strategy (sec. 3.3) affect the reconstruction capability of the Learned Random Walker.

5. Acknowledgments

We are grateful to Kay Schneitz, Rachele Tofanelli and Athul Vijayan for sharing data and helpful discussions. This work was supported by the Deutsche Forschungsgemeinschaft (DFG) research unit FOR2581 Quantitative Plant Morphodynamics.

6. Conclusion

We have proposed an end-to-end learned pipeline for seeded segmentation. We successfully trained a CNN jointly with the Random Walker algorithm obtaining very competitive results and outperforming the standard Random Walker algorithm. Furthermore, we propose and implemented an efficient sparse-backpropagation training and experimentally proved that our method is able to train a network with very sparse seeds. In our experiments we always used a dense ground truth, but the proposed approach also allows for training a network from sparse ground truth. We plan to further explore this regime in future work.

References

[1] Freddie Åström, Stefania Petra, Bernhard Schmitzer, and Christoph Schnörr. A Geometric Approach to Image Labeling. In *Proc. ECCV*, 2016.

[2] Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 136–145, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[3] Xue Bai and Guillermo Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct 2007.

[4] John Baras and George Theodorakopoulos. *Path Problems in Networks*. Synthesis Lectures on Communication Networks. Morgan & Claypool Publishers, 2010.

[5] Gedas Bertasius, Lorenzo Torresani, Stella X. Yu, and Jianbo Shi. Convolutional random walk networks for semantic image segmentation. In *CVPR*, pages 6137–6145. IEEE Computer Society, 2017.

[6] Yuri Y. Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *Proceedings Eighth IEEE International Conference on Computer Vision*, volume 1, pages 105–112, 2001.

[7] Liang-Chieh Chen, Alexander Schwing, Alan Yuille, and Raquel Urtasun. Learning deep structured models. In *International Conference on Machine Learning*, pages 1785–1794, 2015.

[8] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. Power watershed: A unifying graph-based optimization framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1384–1399, July 2011.

[9] Jean Cousty, Gilles Bertrand, Laurent Najman, and Michel Couprie. Watershed cuts: Thinnings, shortest path forests, and topological watersheds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):925–939, May 2010.

[10] CREMI. Miccai challenge on circuit reconstruction from electron microscopy images, 2017. `https://cremi.org`.

[11] A. X. Falcao, J. Stolfi, and R. de Alencar Lotufo. The image foresting transform: theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, Jan 2004.

[12] Jan Funke, Fabian David Tschopp, William Grisaitis, Arlo Sheridan, Chandan Singh, Stephan Saalfeld, and Srinivas C Turaga. Large scale image segmentation with structured loss based deep learning for connectome reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[13] Leo Grady. Random walks for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(11):1768–1783, 2006.

[14] Jeremy Jancsary, Sebastian Nowozin, and Carsten Rother. Loss-specific training of non-parametric image restoration models: A new state of the art. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VII*, ECCV'12, pages 112–125, Berlin, Heidelberg, 2012. Springer-Verlag.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[16] Nikos Komodakis and Georgios Tziritas. Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *IEEE Transactions on Image Processing*, 16(11):2649–2661, Nov 2007.

[17] Kisuk Lee, Jonathan Zung, Peter Li, Viren Jain, and H. Sebastian Seung. Superhuman accuracy on the snemi3d connectomics challenge. *CoRR*, abs/1706.00120, 2017.

[18] Filip Malmberg, Robin Strand, Joel Kullberg, Richard Nordenskjöld, and Ewert Bengtsson. Smart paint - a new interactive segmentation method applied to mr prostate segmentation. MICCAI workshop, 2012.

[19] Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and L. Van Gool. Convolutional oriented boundaries: From image segmentation to high-level tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):819–833, April 2018.

[20] Marina Meilă. Comparing clusterings: an axiomatic view. In *In ICML 2005: Proceedings of the 22nd international conference on Machine learning*, pages 577–584. ACM Press, 2005.

[21] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[22] Stefan Roth and Michael J. Black. Fields of experts: a framework for learning image priors. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 860–867 vol. 2, June 2005.

[23] Kegan G. G. Samuel and Marshall F. Tappen. Learning optimized map estimates in continuously-valued mrf models. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 477–484, June 2009.

[24] Ali Kemal Sinop and Leo Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct 2007.

[25] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 81–90, New York, NY, USA, 2004. ACM.

[26] Marshall F. Tappen, Ce Liu, Edward H. Adelson, and William T. Freeman. Learning gaussian conditional random fields for low-level vision. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[27] Paul Vernaza and Manmohan Chandraker. Learning random-walk label propagation for weakly-supervised semantic segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 2953–2961, July 2017.

[28] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, Dec 2007.

[29] Steffen Wolf, Lukas Schott, Ullrich Köthe, and Fred A. Hamprecht. Learned watershed: End-to-end learning of seeded segmentation. *ICCV*, pages 2030–2038, 2017.

[30] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.

[31] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, pages 912–919. AAAI Press, 2003.