The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks

Supplementary Material

Maxim Berman Amal Rannen Triki Matthew B. Blaschko
Dept. ESAT, Center for Processing Speech and Images
KU Leuven, Belgium
{maxim.berman,amal.rannen,matthew.blaschko}@esat.kuleuven.be

A. Detailed results for Section 4.2: binary segmentation on Pascal VOC

Figure A.1 shows segmentations obtained for binary foreground-background segmentation on Pascal VOC under different training losses, after finetuning a base multi-class classification network for a specific class. We see that the Lovász hinge for the Jaccard loss tends to fill gaps in segmentation, recover small objects, and lead to a more sensible segmentation globally.

Table A.1 presents detailed scores for this binary segmentation task. We notice a clear improvement of the per image-IoU by optimizing with the Jaccard loss. Moreover, the results are in agreement with the intuition that the best performance for a given loss on the validation set is achieved when training with that loss. In some limited cases (`boat`, `bottle`) the performance of the base multi-class network is actually higher than the fine-tuned versions. Our understanding of this phenomenon is that the context is particularly important for these specific classes, and the absence of label for the other classes during finetuning impedes the predictive ability of the network. Additionally, Figure A.2 presents an instance of convergence curves of this binary network, under the different losses considered.

Comparison to prior work [22] propose separately approximating the intersection

$$I \simeq \sum_{i=1}^{p} F_i \, [y_i^* = 1], \tag{A.1}$$

using the Iverson bracket notation, and the union

$$U \simeq \sum_{i=1}^{n} (p_i + [y_i^* = 1]) - I \tag{A.2}$$

for optimizing the binary IoU $\simeq I/U$. We compared the validation image mIoU under the loss of [22] and the binary Lovász hinge, for all the categories of binarized Pascal VOC, in the setting of section 4.2. We chose for [22] the best-scoring among 3 learning rates. As seen in Table A.2 the proxy loss in [22] does not reach the performance of our method. Since [22] uses the same approximation "batch–IoU $\simeq$ dataset–IoU", these observations extend to the binary dataset–IoU measure.

B. Supplementary experiment: IBSR brain segmentation

Data and Model In order to test the Lovász-Softmax loss on a different type of images, we consider the publicly available dataset provided by the Internet Brain Segmentation Repository (IBSR) [29]. This dataset is composed of Magnetic Resonance (MR) image data of 18 different patients annotated for segmentation. For this segmentation task, we used a model based on Deeplab [5] adapted to IBSR by Shakeri et al. [30]. Our evaluation follows the same procedure as in the cited paper: a subset of 8 subcortical structures is first selected: left and right thalamus, caudate, putamen, and pallidum, then 3 folds composed of respectively 11, 1, and 6 train, validation, and test volumes are used for training and testing. Table B.1 details the model architecture to which we add batch normalization layers between the convolutional layers and their ReLU activations.

Settings Similarly to [30], we consider the dataset composed of the 256 axial brain slices of each volume rather than using the 3D structure of the data. This dataset is composed of 256×128 grayscale images. Moreover, we discard the images that contain only the background class during training. For each fold, the training data is then limited to $\approx$ 800–900 slices. Training is done with stochastic gradient descent and a learning rate schedule to exponentially decrease from 10^{-1} to 10^{-3} over 35 epochs with either the cross-entropy loss as in the original model, or the Lovász-Softmax loss (the

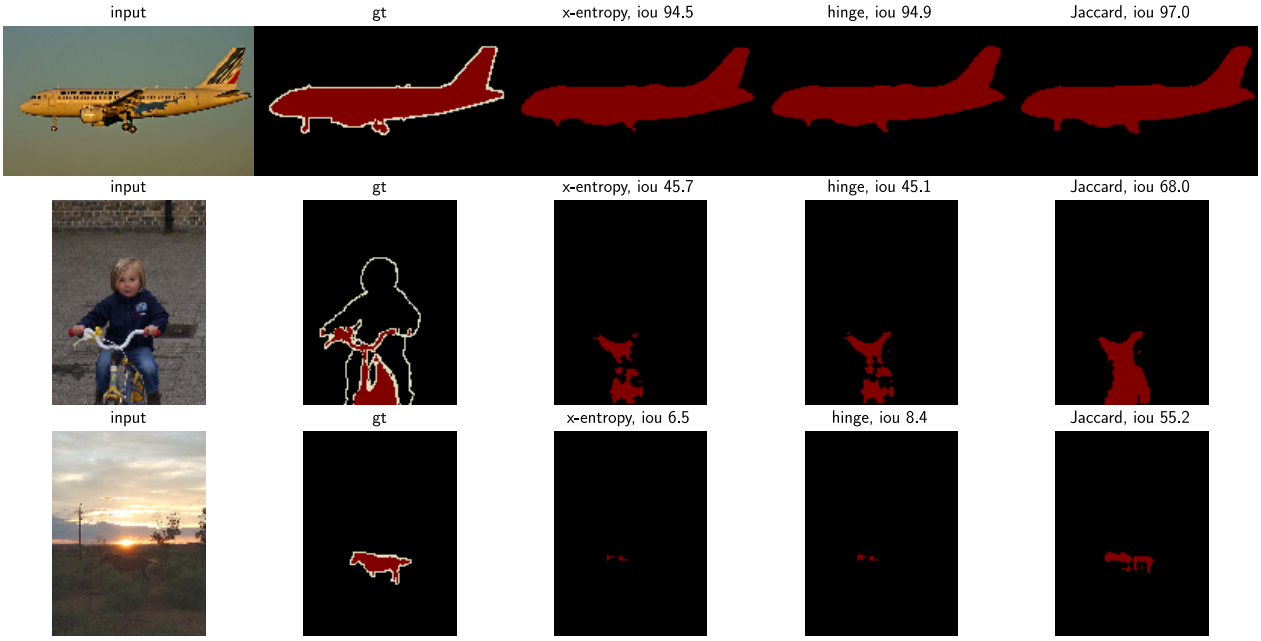


Figure A.1: Example binary segmentations trained with different losses and associated IoU scores on Pascal VOC.

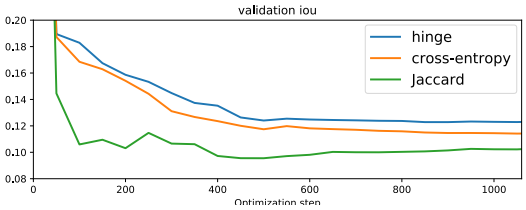


Figure A.2: Evolution of the validation IoU during the course of the optimization with the different losses considered.

batch-mIoU for present classes variant). As we are interested on showing the effect of the loss, we do not apply the CRF post-processing proposed in [30].

Results The mean Jaccard index and DICE over the 3 folds for each of the four classes (right + left) of interest along with the mean scores across all classes are given in Table B.2, showing an improvement when using the Lovász-Softmax loss. Some qualitative results are shown in Figure B.1, highlighting the improvements in detecting some fine subcortical structures when the Lovász-Softmax loss is used.

C. Proximal gradient algorithm

We have developed a specialized optimization procedure for the Lovász Hinge for binary classification with the Jaccard loss, based on a computation of the proximal operator of the Lovász Hinge. We include this algorithm here for completeness but have not used it for the main results of the paper, instead relying on standard stochastic gradient descent with momentum. The *proximal gradient algorithm* we propose here has been independently proposed by Frerix et al. [28].

Our motivation for the proximal gradient algorithm stems from the piecewise-linearity of our loss function, which might destabilize stochastic gradient descent. Instead we would like to exploit the geometry of the Lovász Hinge. We therefore analyze the applicability of (variants of) the proximal gradient algorithm for optimization of a risk functional based on the Lovász hinge.

Definition C.1 (Proximal operator)**.** *The proximal operator of a function f with a regularization parameter λ is*

$$\text{prox}_{f,\lambda}(x) = \arg\min_u f(u) + \frac{\lambda}{2}\|u - x\|^2 \qquad \text{(C.1)}$$

We consider the problem of minimizing a (sub)differentiable function f. Iterative application of the proximal operator with an appropriately decreasing schedule of $\{\lambda_t\}_{0\leq t\leq\infty}$ leads to convergence to a local minimum analogously to gradient descent. Furthermore, it is straightforward to show that, given an appropriately chosen schedule of λ parameters, the proximal gradient algorithm will converge at least as fast as gradient descent.

Proposition C.1. *Given a gradient descent parameter η, $x_{t+1} = x_t - \eta\nabla f(x_t)$, there exists a set of descent parameters $\{\lambda_t\}_{0\leq t\leq\infty}$ such that (i) the step size of the proximal operator is equivalent to gradient descent and (ii)* $\text{prox}_{f,\lambda_t}(x_t) \leq x_t - \eta\nabla f(x_t)$.

Table A.1: Losses measured on our validation set of the 20 Pascal VOC categories, after a training with cross-entropy loss (**x**), hinge-loss (**h**), and Lovász-hinge (**j**). **b** indicates the performance of the base network, trained for all categories.

	aeroplane				bicycle				bird				boat			
training	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**
x-entropy, $\cdot 10^{-2}$		**2.8**	3.3	4.1		12.3	**11.0**	11.3		**3.4**	4.0	4.6		**6.4**	**6.4**	6.9
hinge, $\cdot 10^{-2}$		2.9	**2.6**	2.8		14.8	12.1	**11.5**		3.6	3.3	**3.1**		7.4	**6.6**	**6.6**
Jacc-Hinge, $\cdot 10^{-1}$		3.8	3.6	**2.8**		13.8	12.0	**9.2**		6.2	5.8	**4.1**		7.4	7.4	**5.2**
Image-IoU, %	86.2	88.6	87.7	**89.6**	63.2	61.2	58.7	**66.3**	84.5	82.1	81.3	**86.9**	**80.3**	75.8	73.2	79.9
	bottle				bus				car				cat			
training	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**
x-entropy, $\cdot 10^{-2}$		**5.8**	**5.9**	7.3		**3.7**	4.3	5.1		**4.0**	4.4	5.6		**4.9**	5.2	5.9
hinge, $\cdot 10^{-2}$		6.6	5.6	**4.5**		3.9	**3.4**	3.9		4.4	4.0	**3.5**		5.4	**4.9**	5.1
Jacc-Hinge, $\cdot 10^{-1}$		14.8	11.8	**8.0**		3.6	3.1	**2.4**		9.8	8.9	**5.4**		4.8	4.4	**3.3**
Image-IoU, %	**71.9**	70.1	68.0	70.5	90.7	90.2	90.4	**91.2**	76.3	77.0	75.5	**80.5**	88.7	86.0	86.5	**89.8**
	chair				cow				diningtable				dog			
training	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**
x-entropy, $\cdot 10^{-2}$		11.4	**11.1**	13.1		**6.1**	6.5	7.7		14.1	**12.7**	12.9		**5.7**	6.0	6.3
hinge, $\cdot 10^{-2}$		13.3	11.8	**11.0**		6.9	**6.2**	7.6		16.7	14.5	**13.7**		6.3	**5.8**	**5.8**
Jacc-Hinge, $\cdot 10^{-1}$		16.6	14.4	**9.8**		5.6	5.1	**4.1**		12.5	10.7	**7.9**		5.6	5.0	**3.4**
Image-IoU, %	59.3	54.0	51.2	**59.6**	83.4	84.0	82.6	**86.3**	66.7	70.6	70.0	**73.8**	83.8	82.1	81.7	**87.6**
	horse				motorbike				person				potted-plant			
training	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**
x-entropy, $\cdot 10^{-2}$		**5.2**	6.2	6.5		**6.2**	6.6	7.2		**5.8**	**5.9**	8.1		**6.1**	6.5	7.9
hinge, $\cdot 10^{-2}$		5.7	**5.3**	5.8		7.0	**6.4**	6.8		6.5	6.0	**5.4**		6.9	**6.1**	**6.1**
Jacc-Hinge, $\cdot 10^{-1}$		6.0	5.7	**4.6**		5.1	4.8	**3.7**		8.1	7.4	**4.9**		12.4	10.4	**8.2**
Image-IoU, %	82.4	82.1	79.1	**84.8**	83.8	82.6	82.8	**85.4**	78.2	79.1	77.1	**82.0**	66.1	65.6	65.3	**68.0**
	sheep				sofa				train				tvmonitor			
training	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**	**b**	**x**	**h**	**j**
x-entropy, $\cdot 10^{-2}$		**6.4**	**6.5**	7.8		13.8	**13.4**	14.9		**7.0**	7.2	8.8		**5.6**	6.0	6.2
hinge, $\cdot 10^{-2}$		7.2	**6.4**	7.9		16.4	**15.2**	17.2		7.9	**7.3**	9.2		6.3	5.5	**4.7**
Jacc-Hinge, $\cdot 10^{-1}$		6.3	5.8	**4.6**		10.5	9.9	**8.2**		5.2	5.2	**3.0**		9.3	7.6	**5.9**
Image-IoU, %	83.7	80.3	78.1	**85.3**	69.7	69.6	67.7	**72.1**	88.8	83.9	81.3	**89.7**	78.1	77.8	77.8	**80.6**

Table A.2: Per-class test IoU (%) corresponding to the results by the best learning rate for [22] compared to the results of the Lovász hinge.

	airplane	cycle	bird	boat	bottle	bus	car	cat	chair	cow	d. table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
[22]	79.9	54.7	75.5	72.5	68.7	86.2	73.3	78.4	56.6	75.4	72.2	76.9	68.8	79.4	71.7	62.1	76.5	69.9	77.8	77.1
Lovász-Hinge	89.6	66.3	86.9	79.9	70.5	91.2	80.5	89.8	59.6	86.3	73.8	87.6	84.8	85.4	82.0	68.0	85.3	72.1	89.7	80.6

Proof. Starting with claim (i), we note that the proximal operator is the Lagrangian of the constrained optimization problem $\arg\min_u f(u)$ s.t. $\|x-u\|^2 \leq R$ for some $R > 0$, and we may therefore consider λ_t such that $R_t = \|\eta \nabla f(x_t)\|^2$, where $\{x_t\}_{0 \leq t \leq \infty}$ is the sequence of values visited in gradient descent.

Claim (ii) follows directly from the definition of the proximal operator as the minimization of $f(u)$ within a ball of radius R_t around x_t must be at least as small as the value at the gradient descent direction. □

It is straightforward to convert a gradient descent step size schedule to an equivalent proximal gradient schedule of λ_t values such that, were the objective linear, the two

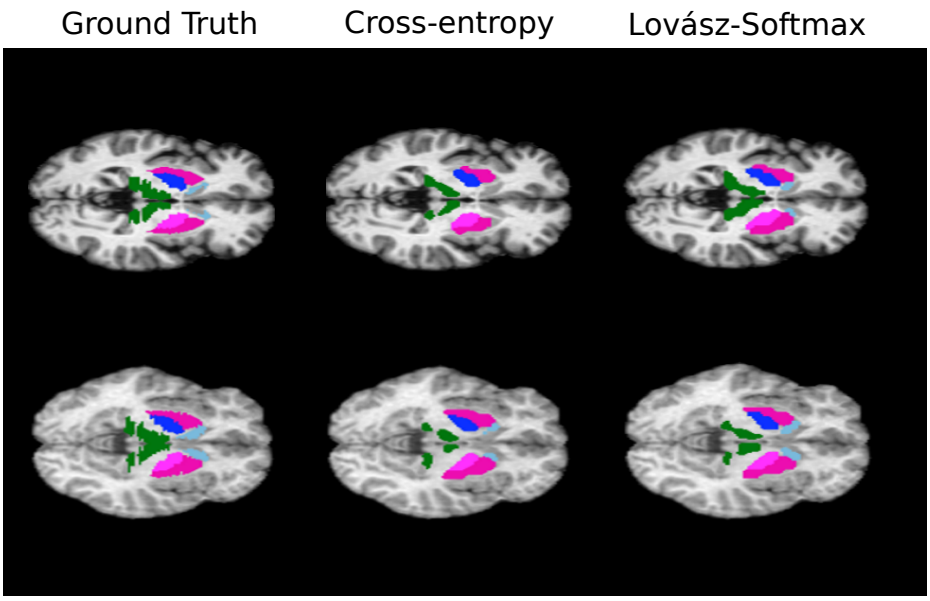


Figure B.1: Some examples of segmentation on the ISBR dataset. These examples are taken from two different patients and two different folds, and show an improvement in the segmentation of some fine structures when the Lovász-Softmax loss is used.

Block	convolution			pooling		batch norm.
	kernel	# filters	dilation	kernel	stride	
1	7×7	64	1	3×3	2	yes
2	5×5	128	1	3×3	2	yes
3	3×3	256	2	3×3	1	yes
4	3×3	512	2	3×3	1	yes
5	3×3	512	2	3×3	1	yes
6	4×4	1024	4	none		yes
7	1×1	9	1	none		no

Table B.1: Layers used for the brain image segmentation.

algorithms would be equivalent. Indeed, the proximal gradient algorithm applied to a piecewise linear objective only differs from gradient descent at the boundaries between linear pieces, in which case it converges in a strictly smaller number of steps than gradient descent.

We optimize a deep neural network architecture by a modified backpropagation algorithm in which the gradient direction with respect to the loss layer is given by the direction of the empirical difference $x_t - \text{prox}_f(x_t)$. We note that this modification to the standard gradient computation is compatible with popular optimization strategies such as Adam [16]. In initial experiments using the true gradient rather than that based on the proximal operator, we found that the use of momentum led to faster empirical convergence than Adam, and we therefore have based our subsequent comparison and empirical results on optimization with momentum.

We show here that these momentum terms still do not lead in practice to as efficient update directions as those defined by the proximal operator.

Definition C.2 (Momentum [31])**.** *Gradient descent with momentum is achieved with the following update rules*

$$v_{t+1} = \alpha v_t + \nabla f(x_t) \tag{C.2}$$
$$x_{t+1} = x_t - \eta v_{t+1}, \tag{C.3}$$

where η is the gradient descent parameter and $\alpha \in [0, 1]$ is the momentum coefficient.

Unrolling this recursion shows that momentum gives an exponentially decaying weighted average of previous gradient values, and setting $\alpha = 0$ recovers classical gradient descent.

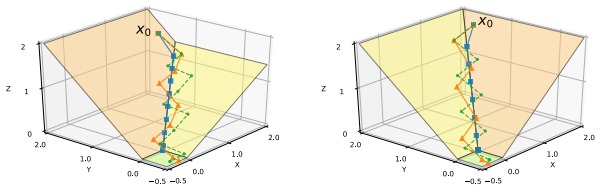
Figure C.1 shows the behavior of gradient descent with momentum on the problem

$$\min_{x \in \mathbb{R}^2} \max \left(0, \left\langle x, \begin{pmatrix} \nu \\ 0 \end{pmatrix} \right\rangle, \left\langle x, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\rangle \right), \tag{C.4}$$

where ν is a positive scalar that allows us to adjust the relative scale of the gradients on either side of the boundary between the pieces. In all cases, the momentum oscillates around piecewise-linear edges, and in Figure C.1c, we see that traversing to a piece of the loss surface with very different slope can lead to multiple steps away from the boundary before returning to a steeper descent direction. By contrast, the proximal algorithm immediately determines the optimal descent direction.

Table B.2: Test results on IBSR brain segmentation task - Average on 3 folds

		Thalamus Proper	Caudate	Putamen	Pallidum	Mean
Cross Entropy	Jaccard	72.74	52.31	61.55	54.04	60.16
	DICE	84.17	68.33	76.07	70.02	74.65
Lovász Softmax	Jaccard	73.56	54.44	62.57	55.74	61.55
	DICE	84.74	70.25	76.89	71.50	75.84



(a) $\nu = 0.7$ (b) $\nu = 1.0$

(c) $\nu = 1.3$

Figure C.1: Optimization behavior of the piecewise-linear surface defined in Equation C.4: gradient descent (green, dashed) and momentum (orange, plain) oscillate around the edge, while the proximal algorithm (green) finds the optimal descent direction.

Optimization study We specialize the proximal gradient algorithm to our proposed Jaccard Hinge loss. We compute an approximate value of the proximal point to any initial point on the loss surface by following a greedy minimization path to the proximal objective C.1. This computation is detailed in Algorithm C.1.

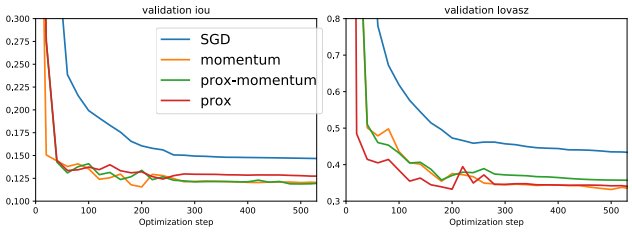


Figure C.2: Jaccard loss optimization with different optimization methods.

Algorithm C.1 Computation of $\text{prox}_{\overline{\Delta_{J_1}},\lambda}(\boldsymbol{m})$

Input: Current $\boldsymbol{m}$, $\overline{\Delta_{J_1}}$, λ
Output: $\boldsymbol{m}^* = \text{prox}_{\overline{\Delta_{J_1}},\lambda}(\boldsymbol{m})$

$\boldsymbol{v}^0$, $\boldsymbol{\pi} \leftarrow$ decreasing ordering of $\boldsymbol{m}$ and permutation
$\boldsymbol{v} \leftarrow \boldsymbol{v}^0$
$\mathbf{g} \leftarrow \mathbf{grad}_{\boldsymbol{v}}\, \overline{\Delta_{J_1}}$ (as a function of the sorted margins)
$\boldsymbol{E} \leftarrow \{$constraint $g_i = g_{i+1} = \ldots = g_{i+p}$
for each equality $v_i = v_{i+1} = \ldots = v_{i+p}\}$
$\boldsymbol{c}_z \leftarrow$ constraint $g_{z+1} = \ldots = g_d$
for z minimal index such that $v_z < 0$
finished $\leftarrow$ False
while not finished **do**
  if $\mathbf{g} = \mathbf{0}$: **break**
  $\mathbf{g} \leftarrow \mathbf{proj}_{\boldsymbol{E} \cup \{\boldsymbol{c}_z\}}\, \mathbf{g}$
  $\boldsymbol{v}_{\text{next}} \leftarrow$ projection of $\boldsymbol{v}$ on the closest edge of $\overline{\Delta_{J_1}}$ in the direction $\mathbf{g}$
  stop $\leftarrow 1/\lambda + \langle \boldsymbol{v} - \boldsymbol{v}^0, \mathbf{g} \rangle / \langle \mathbf{g}, \mathbf{g} \rangle$
  if stop $< \|\boldsymbol{v}_{\text{next}} - \boldsymbol{v}\|$ **then**
    $\boldsymbol{v} \leftarrow \boldsymbol{v} + \text{stop} \cdot \mathbf{g}$
    finished $\leftarrow$ True
  else
    $\boldsymbol{v} \leftarrow \boldsymbol{v}_{\text{next}}$
    Add the new constraint to $\boldsymbol{E}$ or update $\boldsymbol{c}_z$
  end if
end while
return $\boldsymbol{m}^* = \boldsymbol{v}_{\pi^{-1}}$

We investigate the choice of the optimization in terms of empirical convergence rates on the validation data. We evaluate the use of varying optimization strategies for the last layer of the network in Figure C.2. Experimentally, we find that the proximal gradient algorithm converges better than stochastic gradient descent alone, and has similar or better performance to stochastic gradient descent with momentum, which it can easily be combined with.

References

[5] C. Liang Chieh, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *Transactions on Pattern Analysis and Machine Intelligence*, 2017. 1

[16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014. 4

[22] M. A. Rahman and Y. Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International Symposium on Visual Computing*, pages 234–244. Springer, 2016. 1, 3

[28] T. Frerix, T. Möllenhoff, M. Moeller, and D. Cremers. Proximal backpropagation. In *International Conference on Learning Representations*, 2018. 2

[29] T. Rohlfing. Image similarity and tissue overlaps as surrogates for image registration accuracy: Widely used but unreliable. *IEEE Transactions on Medical Imaging*, 31(2):153–163, 2012. 1

[30] M. Shakeri, S. Tsogkas, E. Ferrante, S. Lippe, S. Kadoury, N. Paragios, and I. Kokkinos. Sub-cortical brain structure segmentation using F-CNN's. In *International Symposium on Biomedical Imaging*, pages 269–272. IEEE, 2016. 1, 2

[31] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. on the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on International Conference on Machine Learning*, pages III–1139–1147, 2013. 4