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

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

Abstract

The Jaccard index, also referred to as the intersection-over-union score, is commonly employed in the evaluation of image segmentation results given its perceptual qualities, scale invariance – which lends appropriate relevance to small objects, and appropriate counting of false negatives, in comparison to per-pixel losses. We present a method for direct optimization of the mean intersection-over-union loss in neural networks, in the context of semantic image segmentation, based on the convex Lovász extension of submodular losses. The loss is shown to perform better with respect to the Jaccard index measure than the traditionally used cross-entropy loss. We show quantitative and qualitative differences between optimizing the Jaccard index per image versus optimizing the Jaccard index taken over an entire dataset. We evaluate the impact of our method in a semantic segmentation pipeline and show substantially improved intersection-over-union segmentation scores on the Pascal VOC and Cityscapes datasets using state-of-the-art deep learning segmentation architectures.

1. Introduction

We consider the task of semantic image segmentation, where each pixel i of a given image has to be classified into an object class $c \in \mathcal{C}$. Most of the deep network based segmentation methods rely on logistic regression, optimizing the cross-entropy loss [10]

$$\text{loss}(\boldsymbol{f}) = -\frac{1}{p}\sum_{i=1}^{p} \log f_i(y_i^*), \tag{1}$$

with p the number of pixels in the image or minibatch considered, $y_i^* \in \mathcal{C}$ the ground truth class of pixel i, $f_i(y_i^*)$ the network probability estimate of the ground truth probability of pixel i, and $\boldsymbol{f}$ a vector of all network outputs $f_i(c)$. This supposes that the unnormalized scores $F_i(c)$ of the network have been mapped to probabilities through a *softmax* unit

$$f_i(c) = \frac{e^{F_i(c)}}{\sum_{c' \in \mathcal{C}} e^{F_i(c')}} \quad \forall i \in [1, p], \forall c \in \mathcal{C}. \tag{2}$$

Loss (1) generalizes the logistic loss and leads to smooth optimization. During testing, the decision function commonly used consists in picking the class of maximum score: the predicted class for a given pixel i is $\tilde{y}_i = \arg\max_{c \in \mathcal{C}} F_i(c)$.

The measure of the cross-entropy loss on a validation set is often a poor indicator of the quality of the segmentation. A better performance measure commonly used for evaluating segmentation masks is the Jaccard index, also called the intersection-over-union (IoU) score. Given a vector of ground truth labels $\boldsymbol{y}^*$ and a vector of predicted labels $\tilde{\boldsymbol{y}}$, the Jaccard index of class c is defined as [14]

$$J_c(\boldsymbol{y}^*, \tilde{\boldsymbol{y}}) = \frac{|\{\boldsymbol{y}^* = c\} \cap \{\tilde{\boldsymbol{y}} = c\}|}{|\{\boldsymbol{y}^* = c\} \cup \{\tilde{\boldsymbol{y}} = c\}|}, \tag{3}$$

which gives the ratio in $[0, 1]$ of the intersection between the ground truth mask and the evaluated mask over their union, with the convention that $0/0 = 1$. A corresponding loss function to be employed in empirical risk minimization is

$$\Delta_{J_c}(\boldsymbol{y}^*, \tilde{\boldsymbol{y}}) = 1 - J_c(\boldsymbol{y}^*, \tilde{\boldsymbol{y}}). \tag{4}$$

For multilabel datasets, the Jaccard index is commonly averaged across classes, yielding the mean IoU (mIoU).

We develop here a method for optimizing the performance of a discriminatively trained segmentation system with respect to the Jaccard index. We show that a piecewise linear convex surrogate to the Jaccard loss based on the Lovász extension of submodular set functions yields a consistent improvement of predicted segmentation masks as measured by the Jaccard index.

Although the Jaccard index is often computed globally, over every pixel of the evaluated segmentation dataset [8], it can also be computed independently for each image. Using the per-image Jaccard index is known to have better perceptual accuracy by reducing the bias towards large instances of

the object classes in the dataset [6]. Due to these favorable properties, and the empirical risk minimization principle of optimizing the loss of interest at training time [25], optimization of the Jaccard loss during training has been frequently considered in the literature. However, in contrast to the present work, existing methods all have significant shortcomings that do not allow plug-and-play application to a wide range of learning architectures.

[20] provides a Bayesian framework for optimization of the Jaccard index. The author proposes an approximate algorithm using parametric linear programming to optimize a statistical approximation to the objective. [1] optimize IoU by selecting among a few candidate segmentations, instead of directly optimizing the model with respect to the loss. [3] optimize the Jaccard loss in a structured output SVM, but are only able to do so with a branch-and-bound optimization over bounding boxes and not full segmentations.

Alternative approaches train binary classifiers, but on data that are sampled to capture high Jaccard index. [4, 12] use IoU and related overlap measures to define training sets for binary classifiers in a complex multi-stage training. Such sampling-based approaches clearly induce suboptimality in the empirical risk approximation and do not lend themselves to convenient modular application in a deep learning setting.

Still other recent high-impact research has highlighted the need for optimization of the Jaccard index, but resort to binary training as a proxy, presumably for lack of a convenient and flexible method of directly optimizing the loss of interest. [18] train with logistic loss and test with the Jaccard index. The paper introducing the highly influential OverFeat network specifically addresses the shortcoming in the discussion section [23]: "We are using ℓ_2 loss, rather than directly optimizing the intersection-over-union (IoU) criterion on which performance is measured. Swapping the loss to this should be possible...." However, this is left to future work. In this paper, we develop the necessary plug-and-play loss layer to enable flexible direct minimization of the Jaccard loss in a deep learning setting, while demonstrating its applicability for training state-of-the-art image segmentation networks.

Our approach is based on the recent development of general strategies for generating convex surrogates to submodular loss functions, including the Lovász hinge [26]. Based on the result that the Jaccard loss is submodular, this strategy is directly applicable. We moreover generalize this approach to a multiclass setting by considering a regression-based variant, using a softmax activation layer to naturally map network probability estimates to the Lovász extension of the Jaccard loss. In this work, we (i) apply the Lovász hinge with Jaccard loss to the problem of binary image segmentation (Sec. 2.1), (ii) propose a surrogate for the multi-class setting, the Lovász-Softmax loss (Sec. 2.2), (iii) design a batch-based IoU surrogate that acts as an efficient proxy to the dataset IoU measure (Sec. 3.1), (iv) analyze and compare the properties of different IoU-based measures, and (v) demonstrate a substantial and consistent improvement in performance measured by the Jaccard index in state-of-the-art deep learning based segmentation systems.

2. Optimization surrogates for submodular loss functions

In order to optimize the Jaccard index in a continuous optimization framework, we consider smooth extensions of this discrete loss. The extensions are based on submodular analysis of set functions, where the set function maps from a set of mispredictions to the set of real numbers [26, Equation (6)].

For a segmentation output $\tilde{\boldsymbol{y}}$ and ground truth $\boldsymbol{y}^*$, we define the set of mispredicted pixels for class c as

$$\mathbf{M}_c(\boldsymbol{y}^*, \tilde{\boldsymbol{y}}) = \{\boldsymbol{y}^* = c, \tilde{\boldsymbol{y}} \neq c\} \cup \{\boldsymbol{y}^* \neq c, \tilde{\boldsymbol{y}} = c\}. \quad (5)$$

For a fixed ground truth $\boldsymbol{y}^*$, the Jaccard loss in Eq. (4) can be rewritten as a function of the set of mispredictions

$$\Delta_{J_c} : \mathbf{M}_c \in \{0,1\}^p \mapsto \frac{|\mathbf{M}_c|}{|\{\boldsymbol{y}^* = c\} \cup \mathbf{M}_c|}. \quad (6)$$

Note that for ease of notation, we naturally identify subsets of pixels with their indicator vector in the discrete hypercube $\{0,1\}^p$.

In a continuous optimization setting, we want to assign a loss to any vector of errors $\boldsymbol{m} \in \mathbb{R}^p_+$, and not only to discrete vectors of mispredictions in $\{0,1\}^p$. A natural candidate for this loss is the convex closure of function (6) in $\mathbb{R}^p$. In general, computing the convex closure of set functions is NP-hard. However, the Jaccard set function (6) has been shown to be submodular [27, Proposition 11].

Definition 1 [9]. *A set function $\Delta : \{0,1\}^p \to \mathbb{R}$ is* submodular *if for all $\mathbf{A}, \mathbf{B} \in \{0,1\}^p$*

$$\Delta(\mathbf{A}) + \Delta(\mathbf{B}) \geq \Delta(\mathbf{A} \cup \mathbf{B}) + \Delta(\mathbf{A} \cap \mathbf{B}). \quad (7)$$

The convex closure of submodular set functions is *tight* and *computable in polynomial time* [19]; it corresponds to its Lovász extension.

Definition 2 [2, Def. 3.1]. *The* Lovász extension *of a set function $\Delta : \{0,1\}^p \to \mathbb{R}$ such that $\Delta(\mathbf{0}) = 0$ is defined by*

$$\overline{\Delta} : \boldsymbol{m} \in \mathbb{R}^p \mapsto \sum_{i=1}^{p} m_i \, g_i(\boldsymbol{m}) \quad (8)$$

$$\textit{with } g_i(\boldsymbol{m}) = \Delta(\{\pi_1, \ldots, \pi_i\}) - \Delta(\{\pi_1, \ldots, \pi_{i-1}\}), \quad (9)$$

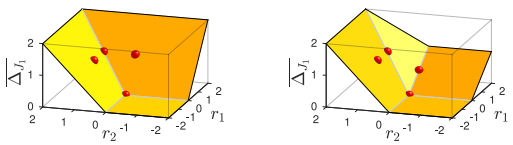
$\boldsymbol{\pi}$ being a permutation ordering the components of $\boldsymbol{m}$ in decreasing order, i.e. $x_{\pi_1} \geq x_{\pi_2} \ldots \geq x_{\pi_p}$.

Let Δ be a set function encoding a submodular loss such as the Jaccard loss defined in Equation (6). By submodularity $\overline{\Delta}$ is the tight convex closure of Δ [19]. $\overline{\Delta}$ is piecewise linear and interpolates the values of Δ in $\mathbb{R}^p \setminus \{0,1\}^p$, while having the same values as Δ on $\{0,1\}^p$, i.e. on any set of mispredictions (Equation (5)). Intuitively, if $\boldsymbol{m}$ is a vector of all pixel errors, $\overline{\Delta}(\boldsymbol{m})$ is a sum weighting these errors according to the interpolated discrete loss. By its convexity and continuity, $\overline{\Delta}$ is a natural surrogate for the minimization of Δ with first-order continuous optimization, such as in neural networks. The elementary operations involved to compute $\overline{\Delta}$ (*sort*, *dot product*, ...) are differentiable and implemented on GPU in current deep learning frameworks. The vector $\boldsymbol{g}(\boldsymbol{m})$ of which the components are defined in Equation (9) directly corresponds to the derivative of $\overline{\Delta}$ with respect to $\boldsymbol{m}$.

In the following, we consider two different settings in which we construct surrogate losses by using the Lovász extension and specifying the vector of errors $\boldsymbol{m}$ that we use:

1. The foreground-background segmentation problem, which leads to the Lovász hinge, as described in [27];
2. The multiclass segmentation problem, which leads to the Lovász-Softmax loss, incorporating the softmax operation in the Lovász extension.

2.1. Foreground-background segmentation



(a) GT $= [-1,-1]$ (b) GT $= [-1,1]$

(c) GT $= [1,-1]$ (d) GT $= [1,1]$

Figure 1: Lovász hinge in the case of two pixel predictions for the four possible ground truths GT, as a function of the relative margins $r_i = 1 - F_i(\boldsymbol{x})\, y_i^*$ for $i = 1, 2$. The red dots indicate the values of the discrete Jaccard index.

In the binary case, we consider the optimization of the Jaccard index for the foreground class Δ_{J_1}. We use a max-margin classifier: for an image $\mathbf{x}$, we define

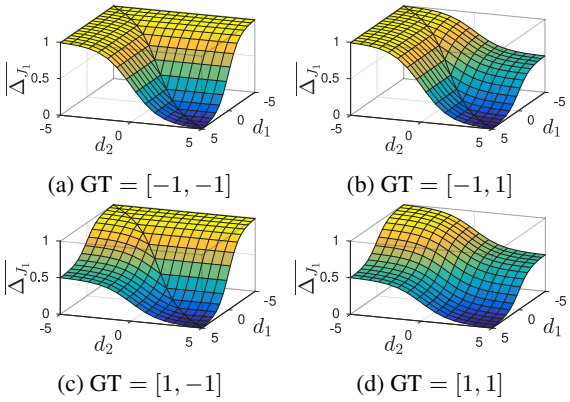
- $y_i^* \in \{-1, 1\}$ the ground truth label of pixel i,
- $F_i(\mathbf{x})$ the i-th element of the output scores $\boldsymbol{F}$ of the model, such that the predicted label $\tilde{y}_i = \text{sign}(F_i(\boldsymbol{x}))$,
- $m_i = \max(1 - F_i(\boldsymbol{x})\, y_i^*, 0)$ the hinge loss associated with the prediction of pixel i.

In this setting, the vector of hinge losses $\boldsymbol{m} \in \mathbf{R}^+$ is the vectors of errors discussed before. With $\overline{\Delta_{J_1}}$ the Lovász extension to Δ_{J_1}, the resulting loss surrogate

$$\text{loss}(\boldsymbol{F}) = \overline{\Delta_{J_1}}(\boldsymbol{m}(\boldsymbol{F})) \tag{10}$$

is the Lovász hinge applied to the Jaccard loss, as described in [26]. It is piecewise linear in the output scores $\boldsymbol{F}$ as a composition of piecewise linear functions. Moreover, by choice of the hinge loss for the vector $\boldsymbol{m}$, the Lovász hinge reduces to the standard hinge loss [24] in the case of a single prediction, or when using the Hamming distance instead of the Jaccard loss as a basis for the construction. Figure 1 illustrates the extension of the Jaccard loss in the case of the prediction of two pixels, illustrating the convexity and the tightness of the surrogate.

2.2. Multiclass semantic segmentation



(a) GT $= [-1,-1]$ (b) GT $= [-1,1]$

(c) GT $= [1,-1]$ (d) GT $= [1,1]$

Figure 2: Lovász-Softmax for the foreground class, with two classes $\{-1, 1\}$ and two pixels, for each ground truth labeling GT. The loss is plotted against the difference of unnormalized scores $d_i = F_i(y_i^*) - F_i(1 - y_i^*)$ for $i = 1, 2$.

In a segmentation setting with more than two classes, we propose a surrogate based on a logistic output instead of using a max-margin setting. Specifically we map the output scores of the model to probability distributions using a softmax unit as is done traditionally in the case of the cross-entropy loss.

We use the class probabilities $f_i(c) \in [0, 1]$ defined in Equation (2) to construct a vector of pixel errors $\boldsymbol{m}(c)$ for class $c \in \mathcal{C}$ defined by

$$m_i(c) = \begin{cases} 1 - f_i(c) & \text{if } c = y_i^*, \\ f_i(c) & \text{otherwise.} \end{cases} \tag{11}$$

We use the vector of errors $\boldsymbol{m}(c) \in [0, 1]^p$ to construct the loss surrogate to Δ_{J_c}, the Jaccard index for class c:

$$\text{loss}(\boldsymbol{f}(c)) = \overline{\Delta_{J_c}}(\boldsymbol{m}(c)) \tag{12}$$

When considering the class-averaged mIoU metric, common in semantic segmentation, we average the class-specific surrogates; hence we define the Lovász-Softmax loss as

$$\text{loss}(\boldsymbol{f}) = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \overline{\Delta_{J_c}}(\boldsymbol{m}(c)) \tag{13}$$

which is piecewise linear in $\boldsymbol{f}$, the normalized network outputs. Figure 2 show this loss as a function of the unnormalized vector outputs $\boldsymbol{F}$ for a prediction of two pixels. In the limit of large scores (confident outputs), the probability vectors at each pixel $(f_i(c))_{c \in \mathcal{C}}$ are close to an indicator vector, and we recover the values of the discrete Jaccard index for the corresponding discrete labeling with respect to the ground truth, as seen on the figure.

3. Optimization of intersection over union

Naïve computation of the Lovász extension (Equation (8)) applied to Δ_{J_c} can be achieved by sorting the elements of $\boldsymbol{m}$ in $\mathcal{O}(p \log p)$ time and doing $\mathcal{O}(p)$ calls to Δ_{J_c}. However, if we compute Δ_{J_c} by Equation (3), each call will cost $\mathcal{O}(n)$. As π is known in advance, we may simply keep track of the cumulative number of false positives and negatives in $\{\pi_1, \dots, \pi_i\}$ for increasing i yielding an amortized $\mathcal{O}(1)$ cost per evaluation of Δ_{J_c} (cf. [27, Equation (43)]). This computation also yields the gradient $\boldsymbol{g}(\boldsymbol{m})$ at the same computational cost. This is a powerful result implying that a tight surrogate function for the Jaccard loss is available and computable in time $\mathcal{O}(p \log p)$. The algorithm for computing the gradient of the loss surface resulting from this procedure is summarized in Algorithm 1.

Algorithm 1 Gradient of the Jaccard loss extension $\overline{\Delta_{J_c}}$

Inputs: vector of errors $\boldsymbol{m}(c) \in \mathbb{R}_+^p$
class foreground pixels $\boldsymbol{\delta} = \{\boldsymbol{y}^* = c\} \in \{0, 1\}^p$
Output: $\boldsymbol{g}(\boldsymbol{m})$ gradient of $\overline{\Delta_{J_c}}$ (Equation (9))

$\boldsymbol{\pi} \leftarrow$ decreasing sort permutation for $\boldsymbol{m}$
$\boldsymbol{\delta_\pi} \leftarrow (\delta_{\pi_i})_{i \in [1,p]}$
intersection $\leftarrow \text{sum}(\boldsymbol{\delta}) - \textbf{cumulative\_sum}(\boldsymbol{\delta_\pi})$
union $\leftarrow \text{sum}(\boldsymbol{\delta}) + \textbf{cumulative\_sum}(1 - \boldsymbol{\delta_\pi})$
$\boldsymbol{g} \leftarrow 1 - \textbf{intersection}/\textbf{union}$
if $p > 1$ **then**
  $\boldsymbol{g}[2:p] \leftarrow \boldsymbol{g}[2:p] - \boldsymbol{g}[1:p-1]$
end if
return $\boldsymbol{g}_{\boldsymbol{\pi}^{-1}}$

3.1. Image–mIoU vs. dataset–mIoU

The official metric of the semantic segmentation task in Pascal VOC [7] and numerous other popular competitions is the dataset–mIoU,

$$\text{dataset–mIoU} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} J_c(\boldsymbol{y}^*, \tilde{\boldsymbol{y}}), \tag{14}$$

where $\boldsymbol{y}^*$ and $\tilde{\boldsymbol{y}}$ contain the ground truth and predicted labels of *all pixels in the testing dataset*.

The Lovász-Softmax loss considers an ensemble of pixel predictions for the computation of the surrogate to the Jaccard loss. In a stochastic gradient descent setting, only a small numbers of pixel predictions are taken into account in one optimization step. Therefore, the Lovász-Softmax loss cannot directly optimize the dataset–mIoU. We can compute this loss over individual images, optimizing for the expected image–mIoU, or over each minibatch, optimizing for the expected batch–mIoU. However, it is not true in general that

$$\mathbb{E}\left(\frac{\text{intersection}}{\text{union}}\right) \approx \frac{\mathbb{E}(\text{intersection})}{\mathbb{E}(\text{union})}, \tag{15}$$

and we found in our experiments that optimizing the image–mIoU or batch–mIoU generally degrades the dataset–mIoU compared with optimizing the standard cross-entropy loss.

The main difference between the dataset and image–mIoU measures resides in the absent classes. When the network wrongly predicts a single pixel belonging to a class that is absent from an image, the image intersection over union loss corresponding to that class changes from 0 to 1. By contrast, a single pixel misprediction does not substantially affect the dataset–mIoU metric.

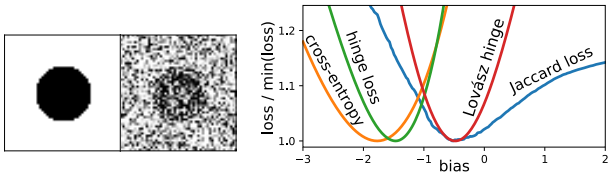
Given this insight, we propose as an heuristic for optimizing the dataset–mIoU to compute the batch Lovász-Softmax surrogate by taking the average in Equation (13) only over the classes present in the batch's ground truth. As a result, the loss is more stable to single predictions in absent classes, mimicking the dataset–mIoU. As outlined in our experiments, the optimization of the Lovász-Softmax restricted to classes present in each batch, effectively translates into gains for the dataset–mIoU metric.

We propose an additional trick for the optimization of the dataset–mIoU. Since the mIoU gives equal importance to each class, and to make the expectation of the batch–mIoU closer to the dataset–mIoU, it seems important to ensure that we feed the network with samples from all classes during training. In order to enforce this requirement, we sample the patches from the training by cycling over every classes, such that each class is visited at least once every $|\mathcal{C}|$ patches. This method is referred to as *equibatch* in our experiments.

4. Experiments

4.1. Synthetic experiment

We demonstrate the relevance of using the Jaccard loss for binary segmentation with a synthetic binary image segmentation experiment. We generate $N = 10$ binary images of size 50×50 representing circles of various radius, and extract for each pixel i a single feature using a unit variance Gaussian perturbation of the ground truth, $f_i \sim \mathcal{N}(\epsilon, 1)$ where



(a) Sample label & features (b) Relative losses for varying bias b

Figure 3: Synthetic model studied in 4.1 and loss objectives.

$\epsilon = 1/2$ for the foreground and $-1/2$ for the background, as illustrated in Figure 3a.

We consider a model classifying pixels in the foreground class for $f_p > -b$, and we learn the bias term b. An exhaustive search, illustrated in Figure 3b, shows that among the losses considered, only the Lovász hinge efficiently captures the absolute minimum of the Jaccard loss.

4.2. Binary segmentation on Pascal VOC

We base our Pascal VOC experiments on the DeeplabV2-single-scale semantic segmentation network [16]. The network uses a Resnet-101 [13] based architecture, re-purposed for image segmentation, notably using dilated (or *atrous*) convolutions. We use the initialization weights provided by the authors. These weights were pre-trained on MS-COCO [17] using cross-entropy loss and weight decay. We further fine-tune these weights on a segmentation dataset consisting of Pascal VOC 2012 training images [8] and the extra images provided by [11], as is common in recent semantic image segmentation applications.

For our binary segmentation experiments, we perform an initial fine-tuning of the weights using cross-entropy loss alone jointly on the 21 classes of Pascal VOC (including the `background` class); this constitutes our basis network. We then turn to binary segmentation by selecting one particular class and finetune the output of the network for the selected class. In order to consider a realistic binary segmentation setting, for each class, we sample the validation set such that half of the images contain at least one foreground pixel. The training is done on random crops of size 321×321 extracted from the training set, with random scale and horizontal flipping. Training batches are randomly sampled from the training set such that half of the selected images contain the foreground class on average.

Our experiments revolve around the choice of the training loss during fine-tuning to binary segmentation. We do a fine-tuning of 2 epoch iterations, with an initial learning rate of $5 \cdot 10^{-4}$, reduced to $1 \cdot 10^{-4}$ after 1 epoch.

Performance of the surrogate Table 1 shows the average of the losses considered after a training with different loss objectives. Evidently, training with a particular loss leads generally to a better objective value of this loss on the validation set. Moreover, we see that the Lovász hinge acts as a good surrogate of the discrete image–IoU, leading to a better validation accuracy for this measure.

Table 1: Average of mean validation binary losses over the 20 Pascal VOC categories, after a training with cross-entropy, hinge, and Lovász hinge loss. The image–mIoU of the basis network, trained for all categories, is equal to 78.29.

Training loss →	Cross-entropy	Hinge	Lovász hinge
Cross-entropy	**6.84**	6.96	7.91
Hinge	7.81	**6.95**	7.11
Lovász hinge	8.37	7.45	**5.44**
Image–IoU	77.14	75.8	**80.5**

Figure 4: Binary `bicycle` masks predicted on a validation image after training the network under various losses.

Figure 4 shows example binary segmentation mask outputs. We notice that the Jaccard loss tends to fill gaps in segmentation, recover small objects, and lead to a more sensible segmentation globally, than other losses considered.

Comparison to prior work [22] propose separately approximating $I \simeq \sum_{i=1}^{p} F_i \, [y_i^* = 1]$ and $U \simeq \sum_{i=1}^{n} (p_i + [y_i^* = 1]) - I$ for optimization of binary IoU $\simeq I/U$. In our experiments, we were not able to observe a consistent improvement of the IoU using this surrogate, contrary to the Lovász hinge. Details on this comparison are included in the Supplementary Material, Section A.

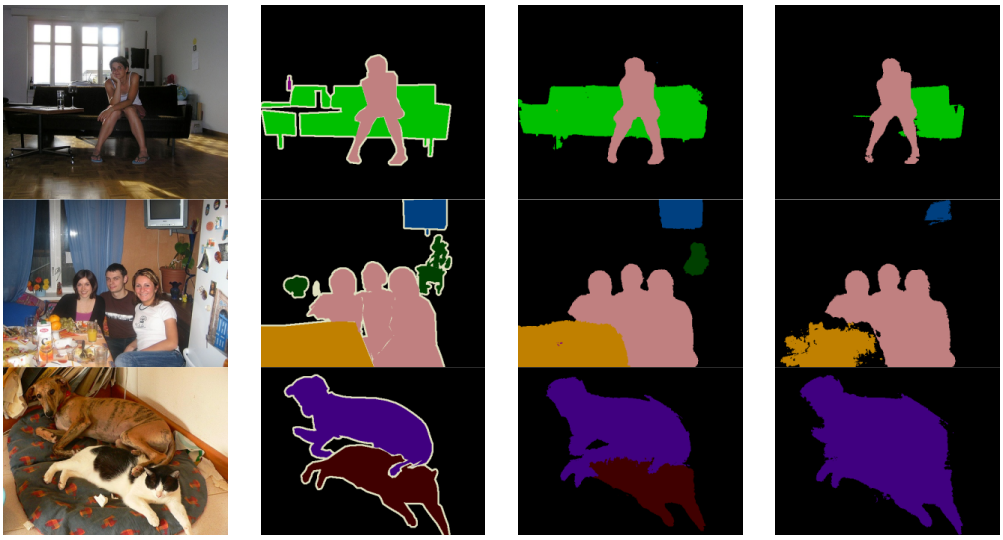
4.3. Multi-class segmentation on Pascal VOC

We again use Deeplab-resnet-v2. This time, we exactly replicate the training procedure of the authors and following the same learning rate schedule, simply swapping out the loss for our multiclass surrogate, the Lovász-Softmax loss as described in Equation (13), with the mean being restricted to the classes present in a given batch.

As in the reference implementation, we use a stochastic gradient descent optimizer with momentum 0.9 and weight decay $5 \cdot 10^{-4}$; the learning rate at training iteration k is

$$lr^{(k)} = lr_{base} \left(1 - \frac{k}{max\_iter}\right)^{power} \tag{16}$$

where $power = 0.9$ and $lr_{base} = 2.5 \cdot 10^{-4}$. We experiment either with 20K iterations of batches of size 10 as in the reference paper, or with 30K iterations. We train the network with patches of size 321×321, with random flipping and rescaling. The 1449 validation images of Pascal VOC are



(a) Input images (b) Ground truth masks (c) Lovász-Softmax + CRF (d) Cross-entropy + CRF

Figure 5: Multiclass segmentations after training with the Lovász-Softmax or the cross-entropy loss, and post-processed with Gaussian CRF. The color scheme follows the standard convention of the Pascal VOC dataset [8].

Table 2: Performance of Deeplab-v2 single-scale trained with cross-entropy (x-loss) vs. Lovász-Softmax loss, for different network evaluations: raw single-scale network output, multi-scale, and Gaussian CRF post-processing.

	validation mIoU (%)			test mIoU (%)
	single-scale	multi-scale	multi-scale + CRF	multi-scale + CRF
x-loss	74.64	76.23	76.53	76.44
x-loss + equibatch	75.53	76.70	77.31	78.05
x-loss + equibatch – 30K iterations	74.97	76.24	76.73	
Lovász	76.56	77.24	77.99	
Lovász + equibatch	76.53	77.28	78.49	
Lovász + equibatch – 30K iterations	**77.41**	**78.22**	**79.12**	**79.00**

included in the training only for experiments evaluated on the official test evaluation server.

We train Deeplab-resnet at a single input scale, which fits the memory constraints of a single GPU. We optionally evaluate the learned weights in a multiscale setting by taking the mean of the probabilities given by the network at scales 1, 0.75, and 0.5, and also include the Gaussian CRF post-processing step used by Deeplab-v2. In this evaluation setting, we found that the baseline performance of the network trained with cross-entropy reaches 76.44% dataset–mIoU on the test set of Pascal VOC.

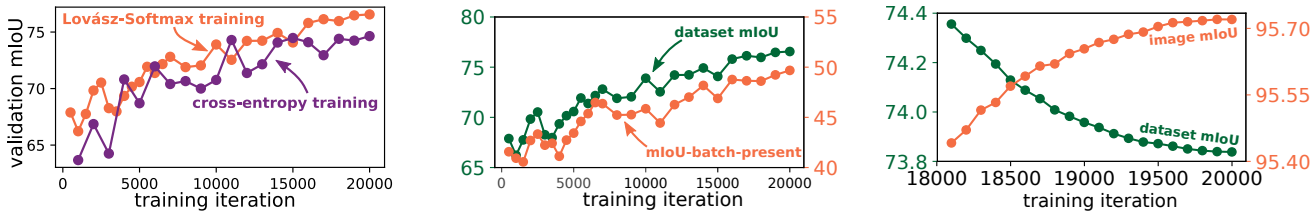
Tables 2 and 3 present the scores obtained after training the network with cross-entropy or Lovász-Softmax loss, with and without *equibatch*, under various evaluation regimes. For a given training and evaluation setting, our loss achieves higher mIoU. Figure 5 shows some example outputs.

Figure 6a shows the evolution of the validation mIoU over the course of the training. We notice that the performance gain manifests itself especially in the last epochs of the optimization. Therefore, we also experiment with the same training setting with 30K iterations, to further benefit from the effects of the loss at these smaller learning rates. In agreement with our intuition, we see in Table 2 that training with our surrogate benefits from a larger number of iterations, in contrast to the original training with cross-entropy.

The CRF post-processing step of Deeplab appears to bring complementary improvements to the use of our mIoU surrogate. While using *equibatch* (batches with cyclic sampling from each class) does significantly help the cross-entropy loss with respect to the dataset–mIoU, its effect on the performance with Lovász-softmax seems marginal. This may be linked with the fact that our loss ignores classes absent from the minibatch ground truth, and therefore relies less on the order of appearance of the classes across batches.

Table 3: Per-class test IoU (%) corresponding to the best-performing variants in Table 2.

	airplane	cycle	bird	boat	bottle	bus	car	cat	chair	cow	d. table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
x-loss	92.95	41.06	87.06	61.23	77.6	91.99	88.11	92.45	32.84	82.48	59.6	90.13	89.83	86.77	85.79	58.06	85.31	52.00	**84.47**	**71.26**
x-loss–equi.	**93.32**	40.29	**91.47**	63.74	77.03	93.10	**86.70**	**93.37**	34.79	**87.92**	69.74	**89.53**	90.61	84.70	85.13	59.23	**87.71**	64.46	82.89	68.57
Lovász–equi 30K	92.63	**41.55**	87.87	**68.41**	**77.75**	**94.71**	**86.71**	90.37	**38.59**	86.24	**74.50**	89.02	**91.69**	**87.28**	**86.37**	**65.92**	87.13	**65.21**	83.69	68.64



(a) Dataset mIoU on the validation set over the course of the Lovász-Softmax or cross-entropy optimization.

(b) Validation dataset–mIoU vs. batch–mIoU restricted to present classes during training with Lovász-Softmax.

(c) Validation dataset–mIoU and image–mIoU during training with Lovász-Softmax optimizing for image–mIoU.

Figure 6: Evolution of some validation measures over the course of the training.

We found however that using *equibatch* facilitates the convergence of the training, as it helps the network to consider all classes during the course of the optimization. This is especially important in the early stages of the optimization, where a class absent for too long can end up being dropped by the classifier in favor of the other classes.

Figure 7: Details of predicted masks after training with Lovász-Softmax per-batch vs. Lovász-Softmax per-image.

Figure 6b shows the joint evolution of the dataset–mIoU, and the batch–mIoU computed over present classes, during training. The correlation between these two measures justifies our choice of restricting the Lovász-Softmax to present classes as a proxy for optimizing the dataset–mIoU. As highlighted by Figure 6c, the image–mIoU is a poor surrogate for the dataset–mIoU, as discussed in Section 3.1: optimizing one measure is generally detrimental to the other.

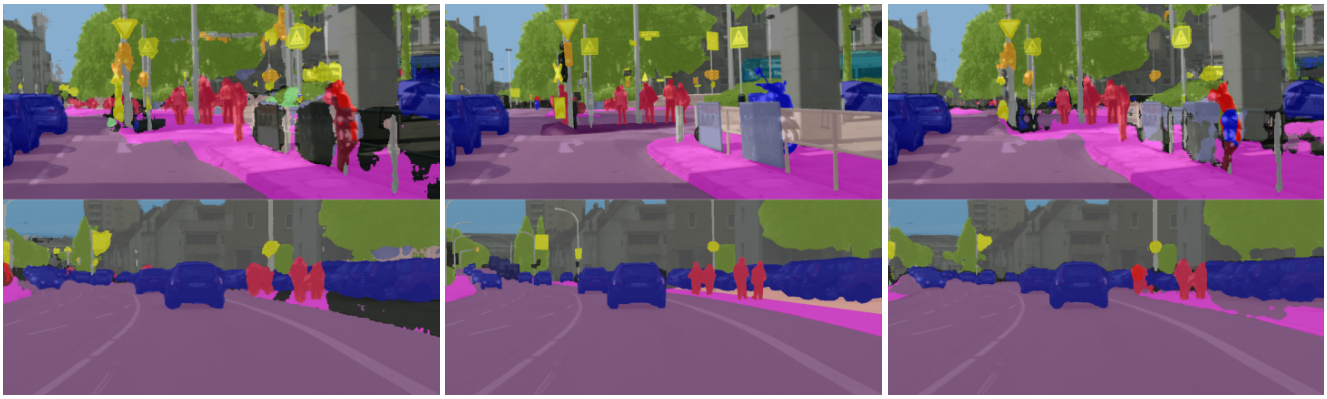
Figure 7 illustrates some qualitative differences between segmentations predicted by the network optimized for batch–mIoU and the network optimized for image–mIoU. The biggest difference between batch–mIoU and image–mIoU is the penalty associated with predicting a class that is absent from the ground truth. Accordingly, we notice that optimizing for image–mIoU tends to produce more sparse outputs, and output less extraneous classes, sometimes at the price of not including classes that are harder to detect.

Comparison to prior work Instead of changing the learning, Nowozin [20] designs a test-time decision function for mIoU based on the assumption of independent classifiers with calibrated probabilities. We applied this method on the Softmax output probabilities of the best model trained with cross-entropy loss (cross-entropy + *equibatch*), and compare with the outputs from Lovász-Softmax (Lovász + equibatch 30K). Since [20] performs a local optimization (batches), we randomly select 20 batches of 21 images with every class represented, optimize the decision function, and compare the optimized mIoU of the batch with the mIoU of the selected batch in our output. The baseline has an average mIoU of 68.7 ± 1.2, our method significantly improves it to 72.5 ± 1.2, while [20] significantly degrades it to 65.1 ± 1.4. We believe this comes from the miscalibration of the neural network's probabilities, which adversely affects the assumptions of the decision function, as discussed in [20, Sec. 5].

4.4. Cityscapes segmentation with ENet

We experiment with ENet, a segmentation architecture optimized for speed [21], on the Cityscapes dataset [5]. We fine-tune the weights provided by the authors, obtained after convergence of weighted cross-entropy loss, a loss that biases the cross-entropy loss to account for class inbalance in the training set. We do not need such a reweighing as our method inherently captures the class balancing of the mIoU.

We finetune ENet using an Adam optimizer [15] with the same learning rate and schedule as in Equation (16).



(a) Initial ENet outputs [21] (b) Ground truth masks (c) ENet + Lovász-Softmax fine-tuning

Figure 8: ENet: parts of output masks before and after fine-tuning with Lovász-Softmax (using the Cityscapes color palette).

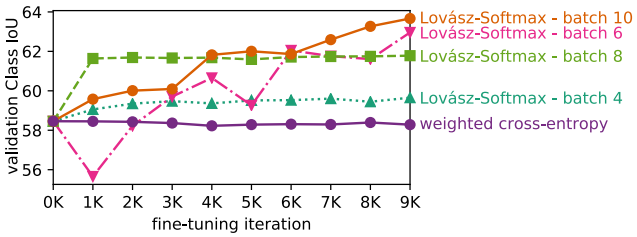


Figure 9: Convergence of ENet on the validation set under fine-tuning with Lovász-Softmax, with various batch sizes.

Consistent with [21], we use images of size 512×1024 with no data augmentation. We snapshot every 1K iterations and report the test performance of snapshot 9K with batches of size 10, which corresponds to the highest validation score.

Fig. 9 shows that our fine-tuning leads to a higher validation mIoU, while further training with weighted cross-entropy barely affects the performance – as expected. Higher batch sizes generally lead to more improvement thanks to a better approximation of the dataset IoU. *Equibatch* training did not make a difference in our experiments, which can be explained by the fact that the dataset is more uniform than Pascal VOC in terms of class representation. Note that we optimize for the mIoU measure, named *Class IoU* in Cityscapes. Accordingly, we observe a substantial gain in performance in Cityscapes IoU metrics, with the Class IoU increasing from 58.29% to 63.06%. Reweighting the different classes in the average of the Lovász-Softmax loss (Equation (13)) could allow us to target IoU-based measures which are weighted differently, such as CityScapes' iIoU metrics. Figure 8 presents some example output masks; we find that our fine-tuning generally reduces false positives and leads to finer details. Of course, our improved segmentation accuracy does not impact the high inference speed for which ENet is designed.

Table 4: Cityscapes results with Lovász-Softmax finetuning

	Class IoU	Class iIoU	Cat. IoU	Cat. iIoU
ENet [21]	58.29	34.36	80.40	**63.99**
Finetuned	**63.06**	34.06	**83.58**	61.05

5. Discussion and Conclusions

In this work, we have demonstrated a versatile approach for optimizing the Jaccard loss for image segmentation. Our proposed method can be flexibly applied to a large number of function classes for segmentation, and we have demonstrated their effectiveness on state-of-the-art deep network architectures, substantially improving accuracies on semantic segmentation datasets simply by optimizing the correct loss during training. Qualitatively, we see greatly improved segmentation quality, in particular on small objects, while large objects tend to have consistent but smaller improvement in accuracy.

This work shows that submodular measures such as the Jaccard index can be readily optimized in a continuous optimization setting. Further work includes the application of the approach to different tasks and losses exhibiting submodularity, and a derivation of specialized optimization routines given the piecewise-linear nature of the Lovász extension.

The code associated with this publication, with replication of the experiments and implementations of the Lovász-Softmax loss, is released on `https://github.com/bermanmaxim/LovaszSoftmax`.

Acknowledgements. This work is partially funded by Internal Funds KU Leuven and FP7-MC-CIG 334380. We acknowledge support from the Research Foundation - Flanders (FWO) through project number G0A2716N. The authors thank J. Yu, X. Jia and Y. Huang for valuable comments and discussions.

References

[1] F. Ahmed, D. Tarlow, and D. Batra. Optimizing expected intersection-over-union with candidate-constrained CRFs. In *The IEEE International Conference on Computer Vision (ICCV)*, 2015. 2

[2] F. Bach et al. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in Machine Learning*, 6(2-3):145–373, 2013. 2

[3] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision – ECCV*, volume 5302 of *Lecture Notes in Computer Science*, pages 2–15. Springer, 2008. 2

[4] L. Bourdev, S. Maji, T. Brox, and J. Malik. Detecting people using mutually consistent poselet activations. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *European Conference on Computer Vision, Part VI*, pages 168–181. Springer, 2010. 2

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 7

[6] G. Csurka, D. Larlus, F. Perronnin, and F. Meylan. What is a good evaluation measure for semantic segmentation? In *Proceedings of the British Machine Vision Conference*, volume 27, 2013. 2

[7] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 4

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://host.robots.ox.ac.uk/pascal/VOC/voc2012/. 1, 5, 6

[9] S. Fujishige. *Submodular Functions and Optimization*. Annals of Discrete Mathematics. Elsevier Science, 2005. 2

[10] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. 1

[11] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision (ICCV)*, 2011. 5

[12] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV, Part VII*, pages 297–312, 2014. 2

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 5

[14] P. Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901. 1

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014. 7

[16] 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. 5

[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 5

[18] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2

[19] L. Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pages 235–257. Springer, 1983. 2, 3

[20] S. Nowozin. Optimal decisions from probabilistic models: The intersection-over-union case. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 2, 7

[21] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016. 7, 8

[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. 5

[23] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013. 2

[24] V. Vapnik. *Statistical learning theory. 1998*. Wiley, New York, 1998. 3

[25] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995. 2

[26] J. Yu and M. B. Blaschko. Learning submodular losses with the Lovász hinge. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Journal of Machine Learning Research: W&CP*, pages 1623—-1631, Lille, France, 2015. 2, 3

[27] J. Yu and M. B. Blaschko. The Lovász hinge: A convex surrogate for submodular losses. 2015. arXiv:1512.07797. 2, 3, 4

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

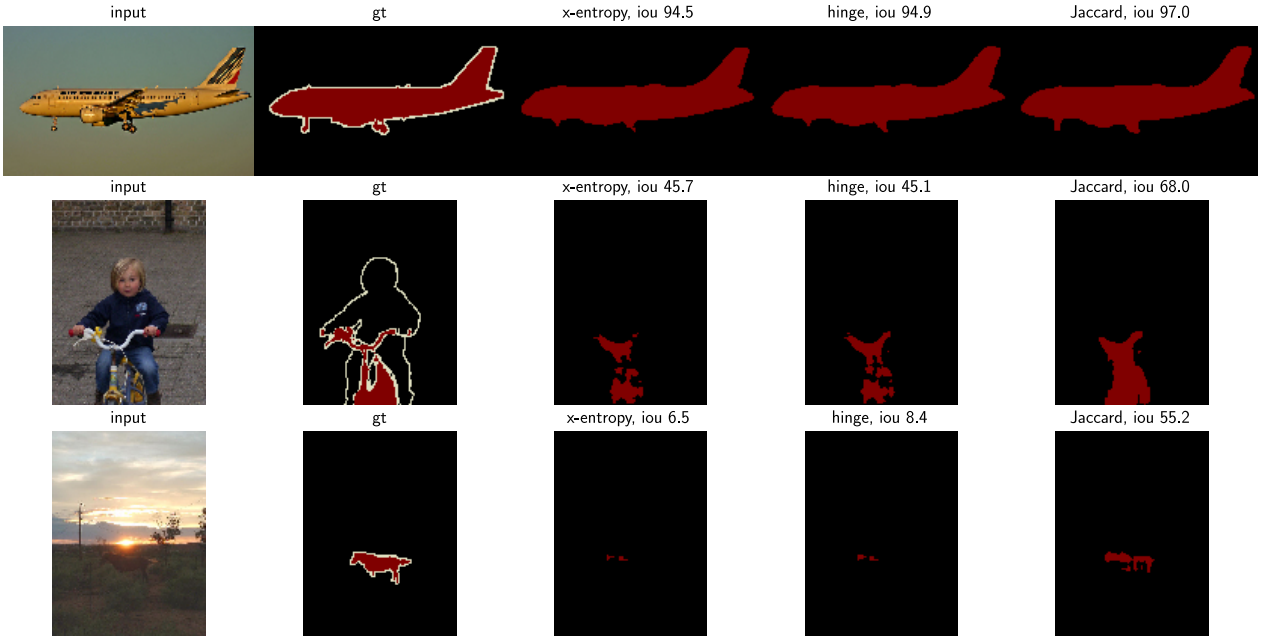


Figure A.1: Example binary segmentations trained with different losses and associated IoU scores on Pascal VOC.

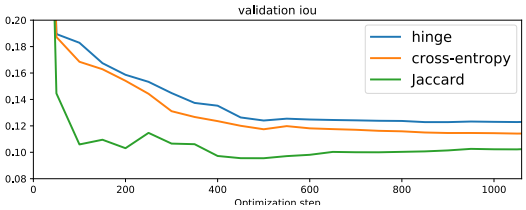


Figure A.2: Evolution of the validation IoU during the course of the optimization with the different losses considered.

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

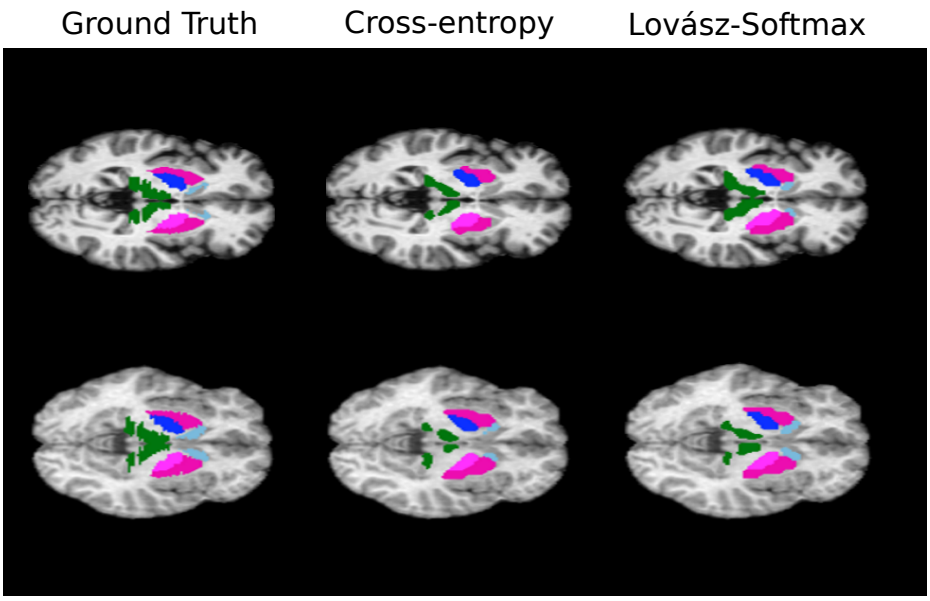


Figure B.1: Some examples of segmentation on the ISBR dataset. These examples are taken from two different patients and two different folds, and show an improvement in the segmentation of some fine structures when the Lovász-Softmax loss is used.

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.

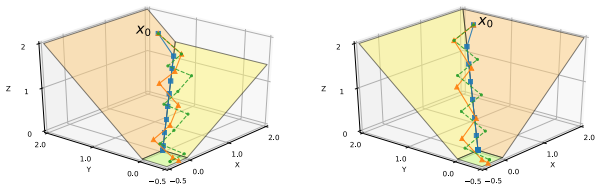
Figure C.1 shows the behavior of gradient descent with momentum on the problem

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

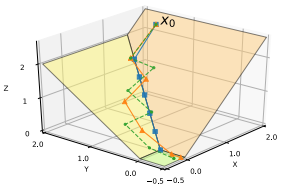
where ν is a positive scalar that allows us to adjust the relative scale of the gradients on either side of the boundary between the pieces. In all cases, the momentum oscillates around piecewise-linear edges, and in Figure C.1c, we see that traversing to a piece of the loss surface with very different 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.

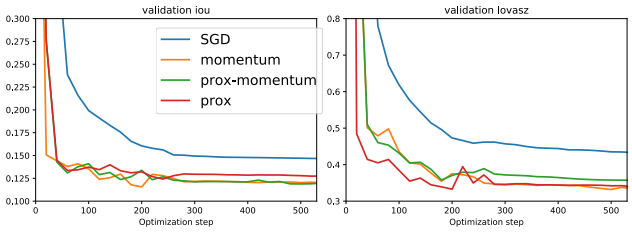


Figure C.2: Jaccard loss optimization with different optimization methods.

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

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

$\boldsymbol{v}^0$, $\pi \leftarrow$ 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.

Supplementary Material 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. I

[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. IV

[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. I, III

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

[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. I

[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. I, II

[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. IV