Loss Max-Pooling for Semantic Image Segmentation

Samuel Rota Bulò*,† Gerhard Neuhold† Peter Kontschieder†
*FBK - Trento, Italy - rotabulo@fbk.eu
†Mapillary - Graz, Austria - {samuel,gerhard,pkontschieder}@mapillary.com

Abstract

We introduce a novel loss max-pooling concept for handling imbalanced training data distributions, applicable as alternative loss layer in the context of deep neural networks for semantic image segmentation. Most real-world semantic segmentation datasets exhibit long tail distributions with few object categories comprising the majority of data and consequently biasing the classifiers towards them. Our method adaptively re-weights the contributions of each pixel based on their observed losses, targeting under-performing classification results as often encountered for under-represented object classes. Our approach goes beyond conventional cost-sensitive learning attempts through adaptive considerations that allow us to indirectly address both, inter- and intra-class imbalances. We provide a theoretical justification of our approach, complementary to experimental analyses on benchmark datasets. In our experiments on the Cityscapes and Pascal VOC 2012 segmentation datasets we find consistently improved results, demonstrating the efficacy of our approach.

1. Introduction

Deep learning approaches have undoubtedly matured to the new de facto standards for many traditional computer vision tasks like image classification, object detection or semantic segmentation. Semantic segmentation aims to assign categorical labels to each pixel in an image and therefore constitutes the basis for high-level image understanding. Recent works have contributed to the progress in this research field by building upon convolutional neural networks (CNNs) [30] and enriching them with task-specific functionalities. Extending CNNs to directly cast dense, semantic label maps [2, 34], including more contextual information [9, 16, 33, 46] or refining results with graphical models [31, 47], have led to impressive results in many real-world applications and on standard benchmark datasets.

Few works have focused on how to properly handle imbalanced (or *skewed*) class distributions, as often encountered in semantic segmentation datasets, within deep neural network training so far. With imbalanced, we refer to datasets having dominant portions of their data assigned to (few) majority classes while the rest belongs to minority classes, forming comparably under-represented categories. As (mostly undesired) consequence, it can be observed that classifiers trained without correction mechanisms tend to be biased towards the majority classes during inference.

One way to mitigate this *class-imbalance* problem is to emphasize on balanced compilations of datasets in the first place by collecting their samples approximately uniformly. Datasets following such an approach are ImageNet [11], Caltech 101/256 [15, 17] or CIFAR 10/100 [29], where training, validation and test sets are roughly balanced w.r.t. the instances per class. Another widely used procedure is conducting over-sampling of minority classes or under-sampling from the majority classes when compiling the actual training data. Such approaches are known to change the underlying data distributions and may result in suboptimal exploitation of available data, increased computational effort and/or risk of over-fitting when repeatedly visiting the same samples from minority classes (*c.f*. SMOTE and derived variants [6, 8, 19, 24] on ways to avoid over-fitting). However, its efficiency and straightforward application for tasks like image-level classification rendered sampling a commonly-agreed practice.

Another approach termed *cost-sensitive* learning changes the algorithmic behavior by introducing class-specific weights, often derived from the original data statistics. Such methods were recently investigated [7, 35, 44, 45] for deep learning, some of them following ideas previously applied in shallow learning methods like random forests [27, 28] or support vector machines [38, 42]. Many of these works use statically-defined cost matrices [7, 12, 35, 44, 45] or introduce additional parameter learning steps [26]. Due to the spatial arrangement and strong correlations of classes between adjacent pixels, cost-sensitive learning techniques are preferred over resampling methods when performing dense, pixel-wise classification as in semantic segmentation tasks. However, current trends of semantic segmentation datasets show strong increase in complexity with more minority classes being added.

Contributions. In this work we propose a principled solution to handling imbalanced datasets within deep learning approaches for semantic segmentation tasks. Specifically, we introduce a novel loss function, which upper bounds the traditional losses where the contribution of each pixel is weighted equally. The upper bound is obtained via a generalized max-pooling operator acting at the pixel-loss level. The maximization is taken with respect to pixel weighting functions, thus providing an adaptive re-weighting of the contributions of each pixel, based on the loss they actually exhibit. In general, pixels incurring higher losses during training are weighted more than pixels with a lower loss, thus indirectly compensating potential inter-class and intra-class imbalances within the dataset. The latter imbalance is approached because our dynamic re-weighting is class-agnostic, *i.e.* we are not taking advantage of the class label statistics like previous cost-sensitive learning approaches.

The generalized max-pooling operator, and hence our new loss, can be instantiated in different ways depending on how we delimit the space of feasible pixel weighting functions. In this paper, we focus on a particular family of weighting functions with bounded p-norm and ∞-norm, and study the properties that our loss function exhibits under this setting. Moreover, we provide the theoretical contribution of deriving an explicit characterization of our loss function under this special case, which enables the computation of gradients that are needed for the optimization of the deep neural network.

As additional, complementary contribution we describe a performance-dependent sampling approach, guiding the minibatch compilation during training. By keeping track of the prediction performance on the training set, we show how a relatively simple change in the sampling scheme allows us to faster reach convergence and improved results.

The rest of this section discusses some related works and how current semantic segmentation approaches typically deal with the class-imbalance problem, before we provide a compact description for the notation used in the rest of this paper. In Sect. 2 we describe how we depart from the standard, uniform weighting scheme to our proposed adaptive, pixel-loss max-pooling and the space of weighting functions we are considering. Sect. 3 and 4 describe how we eventually solve the novel loss function and provide algorithmic details, respectively. In Sect. 5 we assess the performance of our contributions on the challenging Cityscapes and Pascal VOC segmentation benchmarks before we conclude in Sect. 6. Finally, refer to [39] for more in-depth analyses and correctness proofs for our approach.

Related Works. Many semantic segmentation works follow a relatively simple cost-sensitive approach via an inverse frequency rebalancing scheme, *e.g.* [7, 35, 44, 45] or median frequency re-weighting [12]. Other approaches construct best-practice heuristics by *e.g.* restricting the number of pixels to be updated during backpropagation: The work in [3] suggests increasing the minibatch size while decreasing the absolute number of (randomly sampled) pixel positions to be updated. In [43], an approach coined online bootstrapping is introduced, where pixel losses are sorted and only the k highest loss positions are updated. A similar idea termed *online hard example mining* [41] was found to be effective for object detection, where high-loss bounding boxes retained after a non-maximum-suppression step were preferably updated. The work in [23] tackles class imbalance via enforcing inter-cluster and inter-class margins, obtained by employing quintuplet instance sampling with a triple-header hinge loss. Another recent work [26] proposed a cost-sensitive neural network for classification, jointly optimizing for class dependent costs and the standard neural network parameters. The work in [40] addresses the problem of contour detection with convolutional neural networks (CNN), combining a specific loss for contour versus non-contour samples with the conventional log-loss. In separate though related research fields, focus was put on directly optimizing the target measures like Area under curve (AUC), Intersection over Union (IoU or Jaccard Index) or Average class (AC) [1, 4, 36, 37]. The work in [18] is proposing a non-linear activation function computing the L_p norm of projections from the lower layers, allowing to interpret max-, average- and root-mean-squared-pooling operators as special cases of their activation function.

Notation. In this paper, we denote by $\mathcal{A}^\mathcal{B}$ the space of functions mapping elements in the set $\mathcal{B}$ to elements in the set $\mathcal{A}$, while $\mathcal{A}^n$ with n a natual number denotes the usual product set of n-tuples with elements in $\mathcal{A}$. The sets of real and integer numbers are $\mathbb{R}$ and $\mathbb{Z}$, respectively. Let $f, g \in \mathbb{R}^\mathcal{A}$, $c \in \mathbb{R}$. Operations defined on $\mathbb{R}$ such as, *e.g.* addition, multiplication, exponentiation, *etc.*, are inherited by $\mathbb{R}^\mathcal{A}$ via pointwise application (for instance, $f + g$ is the function $z \in \mathcal{A} \mapsto f(z) + g(z)$ and f^c is the function $z \in \mathcal{A} \mapsto f(z)^c$). Additionally, we use the notations:

- $\langle f \rangle_\mathcal{B} = \sum_{z \in \mathcal{A} \cap \mathcal{B}} f(z)$, and $\langle f \rangle = \langle f \rangle_\mathcal{A}$
- $\|f\|_{p,\mathcal{B}} = \left(\sum_{z \in \mathcal{A} \cap \mathcal{B}} f(z)^p\right)^{1/p}$ and $\|f\|_p = \|f\|_{p,\mathcal{A}}$
- $f \cdot g = \sum_{z \in \mathcal{A}} f(z) g(z)$
- $f \preceq c \iff (\forall z \in \mathcal{A})(f(z) \leq c)$
- $(f)_+$ denotes the function $z \in \mathcal{A} \mapsto \max\{f(z), 0\}$.

2. Pixel-Loss Max-Pooling

The goal of semantic image segmentation is to provide an assignment of class labels to each pixel of an image. The input space for this task is denoted by $\mathcal{X}$ and corresponds to the set of possible images. For the sake of simplicity, we assume all images to have the same number of pixels . We denote by $\mathcal{I} \subset \mathcal{Z}^2$ the set of pixels within an image, and let n be the number of pixels, *i.e.* $n = |\mathcal{I}|$. The output space for the segmentation task is denoted by $\mathcal{Y}$ and corresponds to all pixelwise labelings with classes in $\mathcal{C}$. Each labeling $y \in \mathcal{Y}$ is a function mapping pixels to classes, *i.e.* $\mathcal{Y} = \mathcal{C}^{\mathcal{I}}$.

Standard setting. The typical objective used to train a model $f_\theta \in \mathcal{Y}^{\mathcal{X}}$ with parameters θ (*e.g.* a fully-convolutional network), given a training set $\mathcal{T} \subset \mathcal{X} \times \mathcal{Y}$, takes the following form:

$$\min \left\{ \sum_{(x,y)\in\mathcal{T}} L(f_\theta(x), y) + \lambda R(\theta) \;:\; \theta \in \Theta \right\}, \tag{1}$$

where Θ is the set of possible network parameters, $L \in \mathbb{R}^{\mathcal{Y}\times\mathcal{Y}}$ is a loss function penalizing wrong image labelings and $R \in \mathbb{R}^{\Theta}$ is a regularizer. The loss function L commonly decomposes into a sum of pixel-specific losses as follows

$$L(\hat{y}, y) = \frac{1}{n}\langle \ell_{\hat{y}y} \rangle, \tag{2}$$

where $\ell_{\hat{y}y} \in \mathbb{R}^{\mathcal{I}}$ assigns to each pixel $u \in \mathcal{I}$ the loss incurred for predicting class $\hat{y}(u)$ instead of $y(u)$. In the rest of the paper, we assume $\ell_{\hat{y}y}$ to be *non-negative* and *bounded* (*i.e.* pixel losses are finite).

Loss max-pooling. The loss function defined in (2) weights uniformly the contribution of each pixel within the image. The effect of this choice is a bias of the learner towards elements that are dominant within the image (*e.g.* sky, building, road) to the detriment of elements occupying smaller portions of the image. In order to alleviate this issue, we propose to adaptively reweigh the contribution of each pixel based on the actual loss we observe. Our goal is to shift the focus on image parts where the loss is higher, while retaining a theoretical link to the loss in (2). The solution we propose is an upper bound to L, which is constructed by relaxing the pixel weighting scheme. In general terms, we design a convex, compact space of weighting functions $\mathcal{W} \subset \mathbb{R}^{\mathcal{I}}$, subsuming the uniform weighting function, *i.e.* $\{\frac{1}{n}\}^{\mathcal{I}} \subset \mathcal{W}$, and parametrize the loss function in (2) as

$$L_w(\hat{y}, y) = w \cdot \ell_{\hat{y}y}, \tag{3}$$

with $w \in \mathcal{W}$. Then, we define a new loss function $L_{\mathcal{W}} \in \mathbb{R}^{\mathcal{Y}\times\mathcal{Y}}$, which targets the highest loss incurred with a weighting function in $\mathcal{W}$, *i.e.*

$$L_{\mathcal{W}}(\hat{y}, y) = \max\{L_w(\hat{y}, y) \;:\; w \in \mathcal{W}\}. \tag{4}$$

Since the uniform weighting function belongs to $\mathcal{W}$ and we maximize over $\mathcal{W}$, it follows that $L_{\mathcal{W}}$ upper bounds L, *i.e.* $L_{\mathcal{W}}(\hat{y}, y) \geq L(\hat{y}, y)$ for any $\hat{y}, y \in \mathcal{Y}$. Consequently, we obtain an upper bound to (1) if we replace L by $L_{\mathcal{W}}$.

The title of our work, which ties the loss to max-pooling, is inspired by the observation that the loss proposed in (4) is the application of a generalized max-pooling operator acting on the pixel-losses. Indeed, we recover a conventional max-pooling operator as a special case if $\mathcal{W}$ is the set of probability distributions over $\mathcal{I}$. Similarly, the standard loss in (2) can be regarded as the application of an average-pooling operator, which again can be boiled down to a special case of (4) under a proper choice of $\mathcal{W}$.

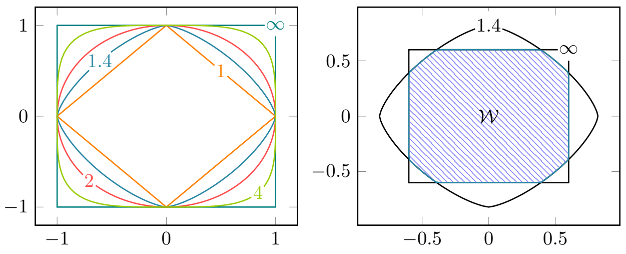

Figure 1: Left: Plot of $\|w\|_p = 1$ in the 2-dimensional case for $p \in \{1, 1.4, 2, 4, \infty\}$. Right: Set $\mathcal{W}$ when $n = 2$, $p = 1.4$ and $\tau = 0.6$.

The space $\mathcal{W}$ of weighting functions. The property that the loss max-pooling operator exhibits depends on the shape of $\mathcal{W}$. Here, we restrict the focus to weighting functions with p-norm ($p \geq 1$) and ∞-norm upper bounded by γ and τ, respectively (see Fig. 1 for an example):

$$\mathcal{W} = \left\{ w \in \mathbb{R}^{\mathcal{I}} \;:\; \|w\|_p \leq \gamma,\ \|w\|_\infty \leq \tau \right\}. \tag{5}$$

We fix the bound on the p-norm to $\gamma = n^{-1/q}$ with $q = \frac{p}{p-1}$, which corresponds to the p-norm of a uniform weighting function. Instead, p and τ are left as hyper-parameters. Possible values of τ should be chosen in the range $[n^{-1}, \gamma]$. Indeed, lower values would prevent the uniform weighting function from belonging to $\mathcal{W}$, while higher values would be equivalent to putting $\tau = \gamma$.

Intuitively, the user can control the pixel selectivity degree of the pooling operation in (4) by changing p. Indeed, the optimal weights will be in general concentrated around a single pixel as $p \to 1$ and be uniformly spread across pixels as $p \to \infty$. On the other hand, τ allows to control, through the relation $m = \left(\frac{\gamma}{\tau}\right)^p$, the minimum number of pixels (namely $\lceil m \rceil$) that should be supported by the optimal weighting function. In Fig. 2 we show some examples, given synthetically-generated losses for $n = 100$ pixels (sorted for better visualization). On the left, we fix $m = n/3$ (*i.e.* at least $1/3$ of the pixels should be supported) and report the optimal weightings for different values of p. As we can see, the weights get more peaked on high losses as p moves towards 1, but the constraint on m prevents selecting less than $\lceil m \rceil$ pixels. On the other hand, the weights tend to become uniform as p approaches ∞. The plot on the right fixes $p = 1.7$ and varies $m \in \{0, 0.1n, 0.2n, 0.4n, 0.8n, n\}$. We see that the weights tend to uniformly support a larger share of pixels as we increase m, yielding the uniform distribution when $m = n$.

3. Computation of $L_{\mathcal{W}}$

The maximization problem in (4) is concave and has an explicit-form solution if $\mathcal{W}$ is defined as in (5). We provide the details by cases, considering the parametrization (p, m) in place of (p, τ), because m has a clear intuitive meaning as mentioned in the previous section. Valid parametrizations satisfy $p \geq 1$ and $1 \leq m \leq n$.

3.1. Case $p > 1$

To address this case, we consider the following dual formulation of the maximization problem in (4):

$$L_{\mathcal{W}}(\hat{y}, y) = \min\left\{g(\lambda) : \lambda \succeq 0, \lambda \in \mathbb{R}^{\mathcal{I}}\right\}, \tag{6}$$

where λ is the dual variable accounting for the constraint $w \preceq \tau$, which is equivalent to $\|w\|_\infty \leq \tau$, and

$$g(\lambda) = \tau\langle\lambda\rangle + \max\left\{w \cdot (\ell_{\hat{y}y} - \lambda) : \|w\|_p \leq \gamma, w \in \mathbb{R}^{\mathcal{I}}\right\}.$$

Moving from the primal to the dual formulation is legitimate because both formulations share the same optimal value. Indeed, the Slater's condition applies [5] (*e.g.* function $z \in \mathcal{I} \mapsto 0$ is strictly feasible).

The maximization in $g(\lambda)$ is the definition of the *dual norm* [5, Appendix A.1.6] of the p-norm, which corresponds to the q-norm with $q = \frac{p}{p-1}$, evaluated in $\ell_{\hat{y}y} - \lambda$ and scaled by γ. Accordingly, we have that

$$g(\lambda) = \tau\langle\lambda\rangle + \gamma\|\ell_{\hat{y}y} - \lambda\|_q. \tag{7}$$

We get a solution to (6) by finding a point λ^* that satisfies

$$\lambda^* = \left(\ell_{\hat{y}y} - m^{-1/q}\|\ell_{\hat{y}y} - \lambda^*\|_q\right)_+ \tag{8}$$

and maximizes $\|\ell_{\hat{y}y} - \lambda\|_q$ (see, [39, Prop. 3]). However, computing such a solution from (8) is not straightforward due to the recursive nature of the formula involving multiple variables (elements of λ^*). We reduce it to the problem of finding the *largest* root of the single-variable function

$$\eta(\alpha) = (m - |\mathcal{J}_\alpha|)\alpha^q - \langle\ell^q_{\hat{y}y}\rangle_{\overline{\mathcal{J}}_\alpha}, \tag{9}$$

where $\mathcal{J}_\alpha = \{u \in \mathcal{I} : \ell_{\hat{y}y}(u) > \alpha\}$ and $\overline{\mathcal{J}}_\alpha = \mathcal{I} \setminus \mathcal{J}_\alpha$ is its complement. This characterization of solutions to the dual formulation (6) in terms of roots of η is proved correct in [39, Prop. 1] and it is used to derive the theorem below, which provides an explicit formula for $L_{\mathcal{W}}(\hat{y}, y)$, the optimal weighting function w^* of the maximization in (4) and the optimal dual variable λ^*:

Theorem 1. *Let $1 \leq q < \infty$, $1 \leq m \leq n$ and $\alpha^* = \frac{\|\ell_{\hat{y}y}\|_{q,\overline{\mathcal{J}}^*}}{(m-|\mathcal{J}^*|)^{1/q}}$, where $\mathcal{J}^* = \{u \in \mathcal{I} : \eta(\ell_{\hat{y}y}(u)) > 0\}$ and $\overline{\mathcal{J}}^* = \mathcal{I} \setminus \mathcal{J}^*$. Then*

$$L_{\mathcal{W}}(\hat{y}, y) = \tau\left[\langle\ell_{\hat{y}y}\rangle_{\mathcal{J}^*} + (m - |\mathcal{J}^*|)\alpha^*\right]. \tag{10}$$

Moreover, $\lambda^* = |\ell_{\hat{y}y} - \alpha^*|_+$ is a minimizer of the dual formulation in (6), while

$$w^*(u) = \begin{cases} \tau & \text{if } u \in \mathcal{J}^* \\ \tau\left(\dfrac{\ell_{\hat{y}y}(u)}{\alpha^*}\right)^{q-1} & \text{if } u \in \overline{\mathcal{J}}^* \text{ and } \alpha^* > 0 \\ 0 & \text{otherwise} \end{cases}$$

is a maximizer of the primal formulation in (4).

3.2. Case $p = 1$

For this case, the solution takes the same form as in (10), but $\mathcal{J}^*$ becomes the subset of $\lfloor m \rfloor$ pixels with the highest losses, while α^* is the highest loss among the remaining pixels ($\alpha^* = 0$ if $\mathcal{J}^* = \mathcal{I}$). As for the optimal weighting function w^*, let $\mathcal{J}^+ = \{u \in \mathcal{I} : \ell_{\hat{y}y}(u) = \alpha^*\} \setminus \mathcal{J}^*$. Then for any probability distribution μ over $\mathcal{J}^+$

$$w^*(u) = \begin{cases} \tau & \text{if } u \in \mathcal{J}^* \\ \tau(m - \lfloor m \rfloor)\mu(u) & \text{if } u \in \mathcal{J}^+ \\ 0 & \text{otherwise,} \end{cases}$$

is an optimal solution for the primal (see, [39, Thm. 1]).

4. Algorithmic Details

The key quantities to compute are $\mathcal{J}^*$ and α^*. Indeed, once those are available we can determine the loss $L_{\mathcal{W}}(\hat{y}, y)$ and compute gradients with respect to the segmentation model's parameters (we show it later in this section). We report in Algorithm 1 the pseudo-code of the computation of $\mathcal{J}^*$ and α^*. We start sorting the losses (line 1). This yields a bijective function $\pi \in \mathcal{I}^{\{1,\dots,n\}}$ satisfying $\ell_{\hat{y}y}(\pi_i) \leq \ell_{\hat{y}y}(\pi_j)$ if $i < j$ (we wrote π_i for $\pi(i)$). Case $p = 1$ (line 13) is trivial, since we know that the last $\lfloor m \rfloor$ ranked pixels will form $\mathcal{J}^*$, while α^* corresponds to the highest loss among the remaining pixels, or 0 if no pixel is left (see, Subsection 3.2). As for case $p > 1$, we walk through the losses in ascending order and stop as soon as we find an index i satisfying one of the following conditions: a) $i = n$ and $\eta_n \leq 0$, or b) $\eta_i > 0$. If the first condition is hit, then $\mathcal{J}^* = \emptyset$ and, hence, $\alpha^* = \|\ell_{\hat{y}y}\|_q / m^{1/q}$. This is indeed what we obtain in line 11, where $i = n + 1$ so that $\alpha^* = (a_n/c_n)^{1/q}$, where $c_n = m$ and $a_n = \sum_{j=1}^n \ell^q_{\hat{y}y}(\pi_j)$. Instead, if condition b is hit, then we have by [39, Prop. 10] that $\mathcal{J}^* = \{\pi_j : i \leq j \leq n\}$. Consequently, $c_i = m - n + i = m - |\mathcal{J}^*|$ and $a_i = \langle\ell^q_{\hat{y}y}\rangle_{\overline{\mathcal{J}}^*}$ so that $\alpha^* = (a_i/c_i)^{1/q}$.

Gradient. In order to train the semantic segmentation model we need to compute the partial derivative $\frac{\partial L_{\mathcal{W}}}{\partial \hat{y}}(\hat{y}, y)$. It exists *almost* everywhere[1] and is given by (see derivations

[1]Precisely, it exists in all $(\hat{y}, y)$ having an open neighborhood where $\mathcal{J}^*$ does not change.

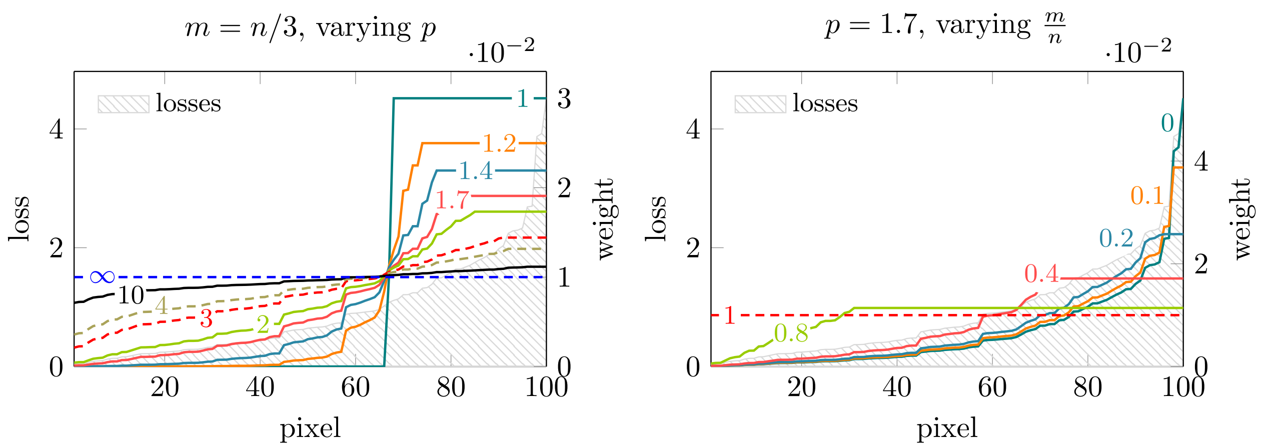

Figure 2: Example of optimal weightings w^* for $n = 100$ pixels. Left: $m = n/3$ and varying values of $p \in \{1, 1.2, 1.4, 1.7, 2, 3, 4, 10, \infty\}$. Right: $p = 1.7$ and varying values of $\frac{m}{n} \in \{0, 0.1, 0.2, 0.4, 0.8, 1\}$. Losses are synthetically generated and sorted for visualization purposes.

Algorithm 1 Compute $\mathcal{J}^*$, α^*

```
Require: $m \in [1, n]$, $p \in [1, \infty]$, $n > 0$ pixel losses $\ell_{\hat{y}y}$
 1: $\pi \leftarrow$ sort($\ell_{\hat{y}y}$)
 2: if $p > 1$ then
 3:     $q \leftarrow \frac{p}{p-1}$
 4:     $c_0 \leftarrow m - n, \quad i \leftarrow 0, \quad a_0 \leftarrow 0$
 5:     repeat
 6:         $i \leftarrow i + 1, \quad c_i \leftarrow c_{i-1} + 1$
 7:         $a_i \leftarrow a_{i-1} + \ell^q_{\hat{y}y}(\pi_i)$
 8:         $\eta_i \leftarrow c_i \, \ell^q_{\hat{y}y}(\pi_i) - a_i$
 9:     until $\eta_i > 0$ or $i = n$
10:     if $\eta_i \leq 0$ then $i \leftarrow i + 1$
11:     $\alpha^* \leftarrow \left(\frac{a_{i-1}}{c_{i-1}}\right)^{1/q}$
12: else
13:     $i \leftarrow n - \lfloor m \rfloor + 1$
14:     $\alpha^* \leftarrow \ell_{\hat{y}y}(\pi_{i-1})$ if $i > 0$ else 0
15: return $\mathcal{J}^* \leftarrow \{\pi_j : i \leq j \leq n\}, \quad \alpha^*$
```

in [39, Appendix B])

$$\frac{\partial L_{\mathcal{W}}}{\partial \hat{y}}(\hat{y}, y) = \frac{\partial \ell_{\hat{y}y}}{\partial \hat{y}} w^* .$$

Note that we will use the same function also where the partial derivative technically does not exist.[2]

Implementation notes. For values of p close to 1, we have that q becomes arbitrarily large and this might cause numerical issues in Algorithm 1. A simple trick to improve stability consists in normalizing the losses with a division by the maximum loss, *i.e.* we consider $\frac{\ell_{\hat{y}y}}{\ell_{\hat{y}y}(\pi(n))}$ in place of $\ell_{\hat{y}y}$. This modification then requires multiplying $L_{\mathcal{W}}(\hat{y}, y)$ by $\ell_{\hat{y}y}(\pi(n))$ to adjust the objective, while the optimal primal solution w^* remains unaffected by the change.

[2]This is a common practice within the deep learning community (see, *e.g.* how the derivative of ReLU, or max-pooling, are computed).

Complimentary sampling strategy. In addition to our main contribution described in the previous sections, we propose a complimentary idea on how to compile mini-batches during training. We propose a mixed sampling approach taking both, uniform sampling from the training data's global distribution and the current performance of the model into account. As a surrogate for the latter, we keep track of the per-class Intersection over Union (IoU) scores on the training data and conduct inverse sampling, which will suggest to preferably pick from under-performing classes (that are often strongly correlated with minority classes). Blending this performance-based sampling idea with uniform sampling ensures to maintain stochastic behavior during training and therefore helps not to over-fit to particular classes.

5. Experiments

We have evaluated our novel loss max-pooling (LMP) approach on the Cityscapes [10] and the extended Pascal VOC [14] semantic image segmentation datasets. In particular, we have performed an extensive parameter sweep on Cityscapes, assessing the performance development for different settings of our hyper-parameters p and m (see Equ. (5) and Fig. 2). All reported numbers are Intersection-over-Union (IoU) (or *Jaccard*) measures in [%], either averaged over all classes or provided on a per-class basis.

5.1. Network architecture

For all experiments, we are using a network architecture similar to the one of DeepLabV2 [9], implemented within Caffe [25] using cuDNN for performance improvement and NCCL[3] for multi-GPU support. In particular, we are using ResNet-101 [21] in a fully-convolutional way with atrous extensions [22, 46] for the base layers before adding DeepLab's atrous spatial pyramid pooling (ASPP). Finally, we apply upscaling (via deconvolution layers with fixed, uniform weights and therefore performing bilinear upsampling) before using standard softmax loss for all baseline methods `BASE`, $\texttt{BASE}_+$ and for the inverse median frequency weighting [12] while we use our proposed loss max-pooling layer in `LMP`. Both our approaches, $\texttt{BASE}_+$ and `LMP` are using the complimentary sampling strategy for minibatch compilation as described in the previous section, while plain uniform sampling in `BASE` leads to similar results as reported in [9]. We also report results of our new loss with plain uniform sampling ("Proposed loss only"). To save computation time and provide a conclusive parameter sensitivity study for our approach, we disabled both, multi-scale input to the networks and post processing via conditional random fields (CRF). We consider both of these features as complementary to our method and highly relevant for improving the overall performance in case time and hardware budgets permit to do so. However, our primary intention is to demonstrate the effectiveness of our `LMP`, under comparable settings with other baselines like our $\texttt{BASE}_+$. All our reported numbers and plots are obtained from fine-tuning the MS-COCO [32] pre-trained CNN of [9], which is available for download[4]. In order to provide statistically more significant results, we provide mean and standard deviations obtained by averaging the results at certain steps over the last 30k training iterations. We only report results obtained from a single CNN as opposed to using an ensemble of CNNs, trained using the stochastic gradient descent (SGD) solver with polynomial decay of the learning rate ("poly" as described in [9]) setting both, decay rate and momentum to 0.9. For data augmentation (Augm.), we use random scale perturbations in the range $0.5 - 1.5$ for patches cropped at positions given by the aforementioned sampling strategy, and horizontal flipping of images.

5.2. Cityscapes

This recently-released dataset contains street-level images, taken at daytime from driving scenes in 50 major central European cities in Germany, France and Switzerland. Images are captured at high resolution (2.048 $\times$ 1.024) and are divided into training, validation and test sets holding 2.975, 500 and 1.525 images, respectively. For training and validation data, densely annotated ground truth into 20 label categories (19 objects + ignore) is publicly available, where the 6 most frequent classes account for $\approx$90% of the annotated pixel mass. Following previous works [9, 43], we report results obtained on the validation set. During training, we use minibatches comprising 2 image crops, each of size 550 $\times$ 550. The initial learning rate is set to $2.5e^{-4}$, and we run a total number of $165k$ training iterations.

p	150k	160k	165k	Mean	Std.Dev.
1.0	74,35	74.64	74.64	74.54	0.17
1.1	74.34	74.61	74.60	74.52	0.15
1.2	74.42	74.60	74.77	<u>74.60</u>	0.18
1.3	74.52	74.71	74.69	**74.64**	0.10
1.4	74.33	74.51	74.49	74.44	0.10
1.5	74.03	73.99	74.04	74.02	0.03
1.6	74.05	74.42	74.52	74.33	0.25
1.7	74.10	74.56	74.74	74.57	0.17
1.8	73.65	74.18	74.17	74.00	0.30
1.9	73.97	74.21	74.48	74.22	0.26
2.3	73.93	74.23	74.12	74.09	0.15
$\texttt{BASE}_+$	73.12	73.16	73.10	73.13	0.03

Table 1: Sensitivity analysis for p parameter with m fixed to 25% of valid pixels per crop using *efficient tiling* at test time. Numbers in [%] correspond to results on Cityscapes validation set after indicated training iterations (and averages with corresponding std.dev. thereof). Boldface and underlined values are in correspondence with best and second best results, respectively. Bottom-most row shows results from our baseline $\texttt{BASE}_+$ under the efficient tiling setting.

In Tab. 1, we provide a sensitivity analysis for hyper-parameter p, fixing m to 25% of non-ignore per-crop pixels. Due to the large resolution of images and considerable number of trainings to be run, we employ different tiling strategies during inference. Numbers in Tab. 1 are obtained by using our so-called *efficient tiling* strategy, dividing the validation images into five non-overlapping, rectangular crops with full image height. With this setting, the best result was obtained for $p = 1.3$, closely followed by $p = 1.2$. It can be seen that increasing values for p show a trend towards $\texttt{BASE}_+$ results, empirically confirming the theoretical underpinnings from Sect. 2. After fixing $p = 1.3$, we conducted additional experiments with m selected in a way to correspond to selecting at least 10%, 25% or 50% of non-ignore per-crop pixels, obtaining $74.09\% \pm 0.22$, $\mathbf{74.64} \pm 0.10$ and 73.44 ± 0.21 on the validation data, respectively. Finally, we locked in on $p = 1.3$ and 25%, running an *optimized tiling* strategy on the validation set where we consider a 200 pixel overlap between tiles, allowing for improved context capturing. The final class label decisions for the first half of the overlap area are then exclusively taken by the left tile while the second half is provided by the right tile, respectively. The resulting scores are listed in Tab. 2, demonstrating improved results over `BASE`, $\texttt{BASE}_+$ and related approaches like DeepLabV2 [9] (even when using CRF) or [43] with deeper ResNet and on-

[3] https://github.com/NVIDIA/nccl
[4] http://liangchiehchen.com/projects/DeepLabv2_resnet.html

Method	mean IoU
[9] RN-101 & Augm. & ASPP	71.0
[9] RN-101 & Augm. & ASPP & CRF	71.4
[43] FCRN-101 & Augm.	71.16
[43] FCRN-152 & Augm.	71.51
[43] FCRN-152 & Online BS & Augm.	74.64
`Our approaches - Resnet-101`	
`BASE` Augm. & ASPP	72.55 ±0.04
`BASE`$_+$ Augm. & ASPP	73.63 ±0.04
[12] Inverse median freq. & Augm. & ASPP	69.81 ±0.08
Proposed loss only & Augm. & ASPP	74.17 ±0.03
`LMP` Augm. & ASPP	**75.06** ±0.09

Table 2: ResNet-based results (in [%]) on validation set of Cityscapes dataset using *optimized tiling*.

line bootstrapping (BS). Also our loss alone, *i.e*. without the complimentary sampling strategy, yields improved results over both `BASE` and `BASE`$_+$.

To demonstrate the impact of our approach on under-represented classes, we provide a plot showing the per-class performance gain (`LMP`- `BASE`$_+$ on y-axis in %) vs. the absolute number of pixels for a given object category (x-axis, log-scale) in Fig. 3. Positive values on y indicate improvements (18/19 classes) and class labels attached to x indicate increasing object class pixel label volume for categories from left to right. *E.g*., *motorcycle* is most underrepresented while *road* is most present. The plot confirms how `LMP` naturally improves on underrepresented object classes *without* accessing the underlying class statistics.

Another experiment we have run compares `BASE` to `LMP`: In order to match the result of `LMP`, one has to *e.g*. improve the worst 7 categories by 5% each or the worst 10 categories by 3% each, which we find a convincing argument for `LMP`. Additionally, we illustrate the qualitative evolution of the semantic segmentation for two training images in Fig. 4. Odd rows show segmentations obtained when training with conventional log-loss in `BASE`$_+$, while even rows show the ones obtained using our loss max-pooling `LMP` at an increasing number of iterations. As we can see, `LMP` starts improving on under-represented classes sooner than standard log-loss (see, *e.g*. traffic light and its pole on the middle right in the first image, and the car driver in the second image). Finally, we also report the individual per-class IoU scores in Tab. 3a for both, `BASE`$_+$ and `LMP`, corresponding to the setting from Tab. 2.

5.3. Pascal VOC 2012

We additionally assess the quality of our novel `LMP` on the Pascal VOC 2012 segmentation benchmark dataset [13], comprising 20 object classes and a background class. Images in this dataset are considerably smaller than the ones from the Cityscapes dataset so we increased the minibatch size to 4 (with crop sizes of 321×321), using the (extended)

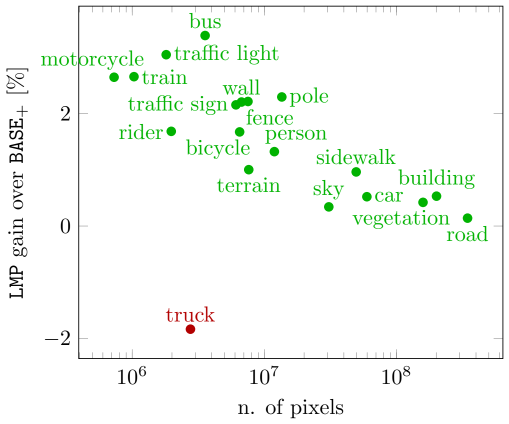

Figure 3: Improvement of `LMP` over `BASE`$_+$ (18/19 classes) as a function of overall per-category pixel count on Cityscapes validation data.

training set with 10.582 images [20]. Testing was done on the validation set containing 1.449 images. We ran a total of 200k training iterations and fixed parameters $p = 1.3$ and m to account for 25% of valid pixels per crop for our `LMP`. During inference, images are evaluated at full scale, *i.e*. no special tiling mechanism is needed. We again report the mean IoU scores in Tab. 4 (this time averaged after training iterations 180k, 190k and 200k), and list results from comparable state-of-the-art approaches [9, 43] next to ours. We can again obtain a considerable relative improvement over `BASE`$_+$ as well as comparable baselines from [9, 43]. Our approach compares slightly worse (-1.4%) with DeepLabV2's strongest variant, which however additionally uses multi-scale inputs (MSC) and refinements from a CRF (contributing 2.55% and 1.34% according to [9], respectively) but only come with increased computational costs. Additionally, and as mentioned above, we consider both of these techniques as complementary to our contributions and plan to integrate them in future works. We finally notice that also for this dataset our new loss alone without the complimentary sampling strategy yields consistent improvements over `BASE` and `BASE`$_+$.

In Tab. 3b, we give side-by-side comparisons of per class IoU scores for `BASE`$_+$ and `LMP`. Again, the majority of categories benefits from our approach, confirming its efficacy.

6. Conclusions

In this work we have introduced a novel approach to tackle imbalances in training data distributions, which do not occur only when we have under-represented classes (inter-class imbalance), but might occur also within the same class (intra-class imbalance). We proposed a new loss function that performs a generalized max-pooling of pixel-specific losses. Our loss upper bounds the traditional one, which gives equal weight to each pixel contribution, and implicitly introduces an adaptive weighting scheme that bi-

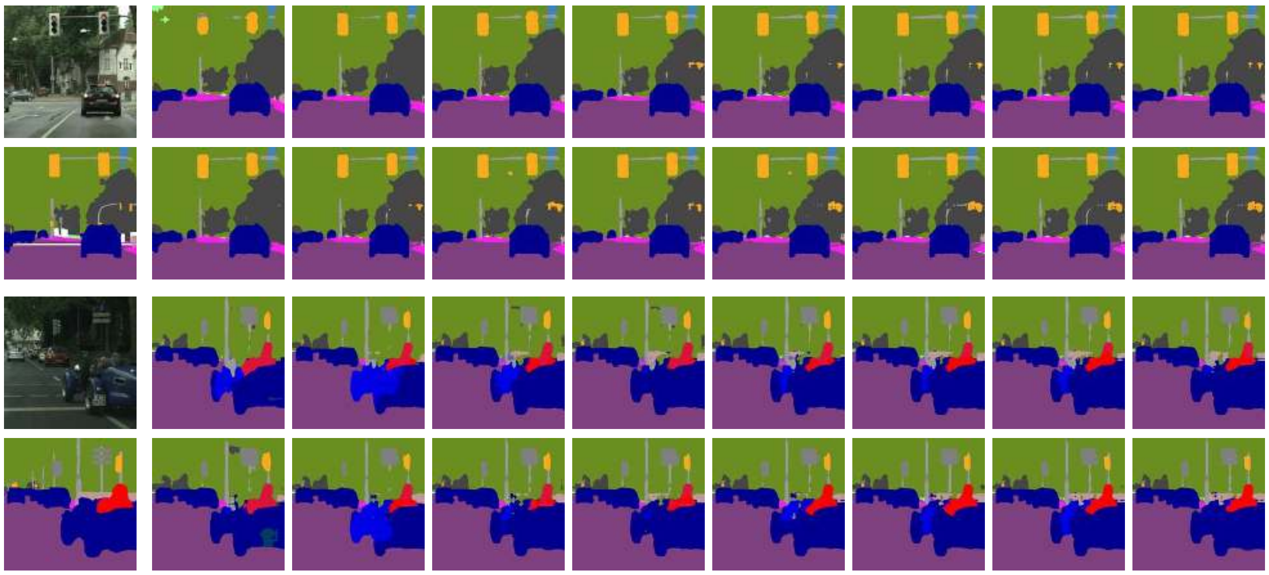

Figure 4: Evolution of semantic segmentation images during training. Left, we have pairs of original images (odd) and their ground-truth segmentations (even). The other images show semantic segmentations obtained by standard log-loss in `BASE`$_+$ (odd rows) and our loss max-pooling `LMP` (even rows) after 20k, 40k, 60k, 80k, 100k, 120k, 140k, 165k training iterations.

(a) Cityscapes

Method	Road	Sidewalk	Building	Wall	Fence	Pole	Traffic Light	Traffic Sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	Mean
`BASE`$_+$ Mean	97.37	80.60	90.99	53.23	54.67	56.72	63.29	72.62	91.19	59.85	93.46	78.59	59.08	93.41	68.94	80.49	67.77	62.51	74.09	73.63
`BASE`$_+$ Std.Dev.	0.01	0.02	0.14	0.92	0.05	0.08	0.56	0.13	0.13	0.96	0.16	0.18	0.31	0.12	0.26	1.08	3.12	0.70	0.11	0.04
`LMP` Mean	97.51	81.56	91.52	55.43	56.88	59.01	66.33	74.77	91.61	60.85	93.80	79.91	60.76	93.93	67.11	83.87	70.42	65.15	75.76	75.06
`LMP` Std.Dev.	0.05	0.32	0.06	0.80	0.68	0.31	0.27	0.21	0.07	0.27	0.09	0.08	0.29	0.01	0.77	0.44	0.53	0.44	0.14	0.09

(b) Pascal VOC 2012

Method	Background	Aeroplane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dining Table	Dog	Horse	Motorbike	Person	Potted Plant	Sheep	Sofa	Train	TV Monitor	Mean
`BASE`$_+$ Mean	92.69	83.21	78.46	81.39	67.95	77.59	92.14	80.17	86.99	38.49	80.86	55.95	81.03	80.64	79.28	81.14	61.74	81.51	47.76	82.25	72.68	75.42
`BASE`$_+$ Std.Dev.	0.00	0.32	0.04	0.03	0.14	0.02	0.12	0.14	0.08	0.03	0.06	0.10	0.02	0.22	0.06	0.03	0.35	0.21	0.24	0.09	0.24	0.04
`LMP` Mean	92.84	85.02	79.62	81.43	69.99	76.36	92.38	82.38	89.43	39.78	82.70	58.60	82.85	81.82	80.17	81.60	61.22	84.30	45.44	82.52	71.70	76.29
`LMP` Std.Dev.	0.02	0.18	0.03	0.34	0.26	0.10	0.17	0.05	0.10	0.06	0.17	0.10	0.05	0.03	0.07	0.05	0.14	0.19	0.04	0.09	0.10	0.02

Table 3: Class-specific IoU scores on Cityscapes (with *optimized tiling* during inference) and Pascal VOC 2012 validation datasets for our baseline (`BASE`$_+$) and our proposed loss max-pooling (`LMP`). All numbers in [%].

Method	mean IoU
[9] RN-101 Base w/o COCO	68.72
[9] RN-101 & MSC & Augm. & ASPP	76.35
[9] RN-101 & MSC & Augm. & ASPP & CRF	77.69
[43] FCRN-101 & Augm.	73.41
[43] FCRN-152 & Augm.	73.32
[43] FCRN-101 & Online BS & Augm.	74.80
[43] FCRN-152 & Online BS & Augm.	74.72
`Our approaches - Resnet-101`	
`BASE` Augm. & ASPP	75.74 $\pm$0.05
`BASE`$_+$ Augm. & ASPP	75.42 $\pm$0.04
[12] Inverse median freq. & Augm. & ASPP	74.93 $\pm$0.03
Proposed loss only & Augm. & ASPP	76.01 $\pm$0.01
`LMP` Augm. & ASPP	76.29 $\pm$0.02

Table 4: ResNet based results on Pascal VOC 2012 segmentation validation data. All numbers in [%].

ases the learner towards under-performing image parts. The space of weighting functions involved in the maximization can be shaped to enforce some desired properties. In this paper we focused on a particular family of weighting functions, enabling us to control the pixel selectivity and the extent of the supported pixels. We have derived explicit formulas for the outcome of the pooling operation under this family of pixel weighting functions, thus enabling the computation of gradients for training deep neural networks. We have experimentally validated the effectiveness of our new loss function and showed consistently improved results on standard benchmark datasets for semantic segmentation.

Acknowledgements. We gratefully acknowledge financial support from project *DIGIMAP*, funded under grant #860375 by the Austrian Research Promotion Agency (FFG).

References

[1] F. Ahmed, D. Tarlow, and D. Batra. Optimizing expected intersection-over-union with candidate-constrained crfs. In *(ICCV)*, pages 1850–1858, 2015. 2

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. 1

[3] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan. Pixelnet: Towards a general pixel-level architecture. *CoRR*, abs/1609.06694, 2016. 2

[4] M. Blaschko and C. Lampert. Learning to localize objects with structured output regression. In *(ECCV)*, 2008. 2

[5] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. 4

[6] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. *Safe-Level-SMOTE: Safe-Level-Synthetic Minority Over-Sampling TEchnique for Handling the Class Imbalanced Problem*, pages 475–482. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. 1

[7] H. Caesar, J. R. R. Uijlings, and V. Ferrari. Joint calibration for semantic segmentation. In *(BMVC)*, 2015. 1, 2

[8] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artificial Intell. Res. (JAIR)*, 16:321–357, 2002. 1

[9] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *CoRR*, abs/1606.00915, 2016. 1, 5, 6, 7, 8

[10] 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 *(CVPR)*, 2016. 5

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *(CVPR)*, 2009. 1

[12] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *(ICCV)*, pages 2650–2658, 2015. 1, 2, 6, 7, 8

[13] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015. 7

[14] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *(IJCV)*, 88(2):303–338, 2010. 5

[15] L. Fei-fei, R. Fergus, and P. Perona. One-shot learning of object categories. *(PAMI)*, 28:2006, 2006. 1

[16] G. Ghiasi and C. C. Fowlkes. Laplacian reconstruction and refinement for semantic segmentation. *CoRR*, abs/1605.02264, 2016. 1

[17] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. 1

[18] Ç. Gülçehre, K. Cho, R. Pascanu, and Y. Bengio. Learned-norm pooling for deep neural networks. *CoRR*, abs/1311.1780, 2013. 2

[19] H. Han, W.-Y. Wang, and B.-H. Mao. *Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning*, pages 878–887. Springer Berlin Heidelberg, 2005. 1

[20] B. Hariharan, P. Arbelez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *(ICCV)*, pages 991–998, Nov 2011. 7

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 6

[22] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In J.-M. Combes, A. Grossmann, and P. Tchamitchian, editors, *Wavelets: Time-Frequency Methods and Phase Space*, pages 286–297. Springer Berlin Heidelberg, 1987. 6

[23] C. Huang, Y. Li, C. C. Loy, and X. Tang. Learning deep representation for imbalanced classification. In *(CVPR)*, 2016. 2

[24] P. Jeatrakul, K. W. Wong, and C. C. Fung. Classification of imbalanced data by combining the complementary neural network and SMOTE algorithm. In *Intern. Conf. on Neural Information Processing*, pages 152–159. Springer, 2010. 1

[25] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 5

[26] S. H. Khan, M. Bennamoun, F. A. Sohel, and R. Togneri. Cost sensitive learning of deep feature representations from imbalanced data. *CoRR*, abs/1508.03422, 2015. 1, 2

[27] T. M. Khoshgoftaar, M. Golawala, and J. Van Hulse. An empirical study of learning from imbalanced data using random forest. In *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, volume 2, pages 310–317. IEEE, 2007. 1

[28] P. Kontschieder, P. Kohli, J. Shotton, and A. Criminisi. GeoF: Geodesic forests for learning coupled predictors. In *(CVPR)*, pages 65–72, 2013. 1

[29] A. Krizhevsky, V. Nair, and G. Hinton. Cifar-10 (canadian institute for advanced research). 1

[30] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 1

[31] G. Lin, C. Shen, I. Reid, et al. Efficient piecewise training of deep structured models for semantic segmentation. *arXiv preprint arXiv:1504.01013*, 2015. 1

[32] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 6

[33] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. *CoRR*, abs/1506.04579, 2015. 1

[34] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *(CVPR)*, pages 3431–3440, 2015. 1

[35] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *(CVPR)*, June 2015. 1, 2

[36] S. Nowozin. Optimal decisions from probabilistic models: the intersection-over-union case. In *(CVPR)*, June 2014. 2

[37] M. Ranjbar, G. Mori, and Y. Wang. Optimizing complex loss functions in structured prediction. In *(ECCV)*, 2010. 2

[38] B. Raskutti and A. Kowalczyk. Extreme re-balancing for

SVMs: a case study. *ACM Sigkdd Explorations Newsletter*, 6(1):60–69, 2004. 1

[39] S. Rota Bulò, G. Neuhold, and P. Kontschieder. Loss max-pooling for semantic image segmentation. *arXiv preprint arXiv:1704.02966*, 2017. 2, 4, 5

[40] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. Deep-contour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *(CVPR)*, June 2015. 2

[41] A. Shrivastava, A. Gupta, and R. B. Girshick. Training region-based object detectors with online hard example mining. In *(CVPR)*, 2016. 2

[42] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser. SVMs modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 39(1):281–288, 2009. 1

[43] Z. Wu, C. Shen, and A. van den Hengel. High-performance semantic segmentation using very deep fully convolutional networks. *CoRR*, abs/1604.04339, 2016. 2, 6, 7, 8

[44] J. Xu, A. G. Schwing, and R. Urtasun. Tell me what you see and i will show you where it is. In *(CVPR)*, 2014. 1, 2

[45] J. Xu, A. G. Schwing, and R. Urtasun. Learning to segment under various forms of weak supervision. In *(CVPR)*, 2015. 1, 2

[46] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *Int. Conf. on Learning Representations (ICLR)*, 2016. 1, 6

[47] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *(ICCV)*, pages 1529–1537, 2015. 1