SUPPLEMENTAL MATERIAL: Semi-supervised semantic segmentation needs strong, varied perturbations

Geoff French[1]
g.french@uea.ac.uk
Samuli Laine[2]
slaine@nvidia.com
Timo Aila[2]
taila@nvidia.com
Michal Mackiewicz[1]
m.mackiewicz@uea.ac.uk
Graham aFinalyson[1]
g.finlayson@uea.ac.uk

[1] School of Computing Sciences
University of East Anglia
Norwich, UK
[2] NVIDIA
Helsinki, Finland

A Pascal VOC 2012 performance across network architectures

We demonstrate the effectiveness of our approach using a variety of architectures on the PASCAL dataset in Table 1. Using an ImageNet pre-trained DeepLab v3+ our baseline and semi-supervised results are stronger than those of [10].

B Smoothly varying sample density in semantic segmentation

B.1 Derivation of signal processing explanation

In this section we explain the derivation of our signal-processing based explanation of the lack of low-density regions in semantic segmentation problems.

To analyse the smoothness of the distribution of patches over an image we need to compute the L^2 pixel content distance between patches centred on neighbouring pixels. Let us start with two patches A and B – see Figure 1(a,b) – extracted from an image I, centred on horizontally neighbouring pixels, with A one pixel to the left of B. The L^2 distance is $|B-A|$. Given that each pixel in $B-A$ is the difference between horizontally neighbouring pixels, $B-A$ is therefore a patch extracted from the horizontal gradient image $\Delta_x I$ (see Figure 1(c)). The squared distance is the sum of the element-wise squares of $B-A$; it is the sum of the elements in a patch extracted from $(\Delta_x I)^{\circ 2}$. Computing the sums of all patches of size $H \times W$

Prop. Labels	1/100	1/50	1/20	1/8	Full (10582)
	Results from [6, 10] with ImageNet pre-trained DeepLab v2				
Baseline	–	48.3%	56.8%	62.0%	70.7%
Adversarial [6]	–	49.2%	59.1%	64.3%	71.4%
s4GAN+MLMT [10]	–	60.4%	62.9%	67.3%	73.2%
	Our results: Same ImageNet pre-trained DeepLab v2 network				
Baseline	33.09%	43.15%	52.05%	60.56%	72.59%
CutMix	53.79%	64.81%	66.48%	67.60%	72.54%
	Results from [10] with ImageNet pre-trained DeepLab v3+				
Baseline	–	unstable	unstable	63.5%	74.6%
s4GAN+MLMT [10]	–	62.6%	66.6%	70.4%	74.7%
	Our results: ImageNet pre-trained DeepLab v3+ network				
Baseline	37.95%	48.35%	59.19%	66.58%	76.70%
CutMix	59.52%	67.05%	69.57%	72.45%	76.73%
	Our results: ImageNet pre-trained DenseNet-161 based Dense U-net				
Baseline	29.22%	39.92%	50.31%	60.65%	72.30%
CutMix	54.19%	63.81%	66.57%	66.78%	72.02%
	Our results: ImageNet pre-trained ResNet-101 based PSPNet				
Baseline	36.69%	46.96%	59.02%	66.67%	77.59%
CutMix	67.20%	68.80%	73.33%	74.11%	77.42%

Table 1: Performance (mIoU) on augmented PASCAL VOC validation set across a variety of architectures, using same splits as Mittal *et al*. [10]. The results for [6] and [10] are taken from [10].

in a sliding window fashion across $(\Delta_x I)^{\circ 2}$ is equivalent to convolving it with a box kernel $1^{H\times W}$, thus the distance between all horizontally neighbouring patches can be computed using $\sqrt{(\Delta_x I)^{\circ 2} * 1^{H\times W}}$. A box filter – or closely related uniform filter – is a low-pass filter that will suppress high-frequency details, resulting in a smooth output. This is implemented in a Jupyter notebook [7] that is distributed with our code.

B.2 Analysis of patch-to-patch distances within Cityscapes

Our analysis of the CITYSCAPES indicates that semantic segmentation problems exhibit *high intra*-class variance and *low inter*-class variance. We chose 1000 image patch triplets each consisting of an anchor patch A_i and positive P_i and negative N_i patches with the same and different ground truth classes as A_i respectively. We used the L^2 pixel content intra-class distance $|P_i - A_i|^2$ and inter-class distance $|N_i - A_i|^2$ as proxies for variance. Given that a segmentation model must place a decision boundary between neighbouring pixels of different classes within an image we chose A_i and N_i to be immediate neighbours on either side of a class boundary. As the model must also generalise from a labelled images to unlabelled images we searched all images except that containing A_i for the P_i belonging to the same class that minimises $|P_i - A_i|^2$. Minimising the distance chooses the best case intra-class distance over which the model must generalise. The inter-class to intra-class distance ratio histogram on the left of Figure 2 (main paper) underlies the illustration to the right in which the blue intra-class distance is approximately $3\times$ that of the red inter-class distance. The model must

(a) Patch A (b) Patch B (c) Patch from $\Delta_x I$

Figure 1: (a, b) Two patches centred on horizontally neighbouring pixels, extracted from the Cityscapes Image in Figure 1(a) (main paper). The ground truth vegetation class is overlaid in green. The red dot indicates the central pixel. (c) The same patch extracted from the horizontal gradient image.

learn to place the decision boundary between the patches centred on neighbouring pixels, while orienting it sufficiently accurately that it intersects other images at the correct points.

C Setup: 2D toy experiments

The neural networks used in our 2D toy experiments are simple classifiers in which samples are 2D x, y points ranging from -1 to 1. Our networks are multi-layer perceptrons consisting of 3 hidden layers of 512 units, each followed by a ReLU non-linearity. The final layer is a 2-unit classification layer. We use the mean teacher [14] semi-supervised learning algorithm with binary cross-entropy as the consistency loss function, a consistency loss weight of 10 and confidence thresholding [9] with a threshold of 0.97.

The ground truth decision boundary was derived from a hand-drawn 512×512 pixel image. The distance map shown in Figure 1(c) (main paper) was computed using the `scipy.ndimage.morphology.distance_transform_edt` function from SciPy [16], with distances negated for regions assigned to class 0. Each pixel in the distance map therefore has a signed distance to the ground truth class boundary. This distance map was used to generate the contours seen as lines in Figure 1(c) (main paper) and used to support the constrained consistency regularization experiment illustrated in Figure 1(d) (main paper).

The constrained consistency regularization experiment described in Section 3.2 (main paper) required that a sample x should be perturbed to $\hat{x}$ such that they are at the same — or similar — distance to the ground truth decision boundary. This was achieved by drawing isotropic perturbations from a normal distribution $\hat{x} = x + h$ where $h \sim \mathcal{N}(0, 0.117)$ ($0.117 \approx 30$ pixels in the source image), determining the distances $m(x)$ and $m(\hat{x})$ from x and $\hat{x}$ to the ground truth boundary (using a pre-computed distance map) and discarding the perturbation – by masking consistency loss for x to 0 – if $|m(\hat{x}) - m(x)| > 0.016$ ($0.016 \approx 4$ pixels in the source image).

D Semantic segmentation experiment setup

D.1 Adapting semi-supervised classification algorithms for segmentation

In the main paper we explain how we adapted Cutout [3] and CutMix [17] for segmentation. Here we will discuss our approach to adapting standard augmentation, Interpolation Consistency Training (ICT) and Virtual Adversarial Training (VAT). We note that implementations of all of these approaches are supplied with our source code.

D.1.1 Standard augmentation

Our standard augmentation based consistency loss uses affine transformations to modify unsupervised images. Applying different affine transformations within the teacher and student paths results in predictions that not aligned. An appropriate affine transformation must be used to bring them into alignment. To this end, we follow the approach used by Perone *et al*. [12] and Li *et al*. [9]; the original unaugmented image x is passed to the teacher network g_ϕ producing predictions $g_\phi(x)$, aligned with the original image. The image is augmented with an affine transformation $a(\cdot)$: $\hat{x} = a(x)$, which is passed to the student network f_θ producing predictions $f_\theta(a(x))$. The same transformation is applied to the teacher prediction: $a(g_\phi(x))$. The two predictions are now geometrically aligned, allowing consistency loss to be computed.

At this point we would like to note some of the challenges involved in the implementation. A natural approach would be to use a single system for applying affine transformations, e.g. the affine grid functionality provided by PyTorch [2]; that way both the input images and the predictions can be augmented using the same transformation matrices. We however wish to exactly match the augmentation system used by Hung *et al*. [5] and Mittal *et al*. [10], both of which use functions provided by OpenCV [1]. This required gathering a precise understanding of how the relevant functions in OpenCV generate and apply affine transformation matrices in order to match them using PyTorch's affine grid functionality, that must be used to transform predictions.

D.1.2 Interpolation consistency training

ICT was the simplest approach to adapt. We follow the procedure in [15], except that our networks generate pixel-wise class probability vectors. These are blended and loss is computed from them in the same fashion as [15]; the only different is that the arrays/tensors have additional dimensions.

D.1.3 Virtual Adversarial Training

Following the notation of Oliver *et al*. [11], in a classification scenario VAT computes the adversarial perturbation r_{adv} as:

$$r \sim \mathcal{N}(0, \frac{\xi}{\sqrt{dim(x)}} I)$$

$$g = \nabla_r d(f_\theta(x), f_\theta(x+r))$$

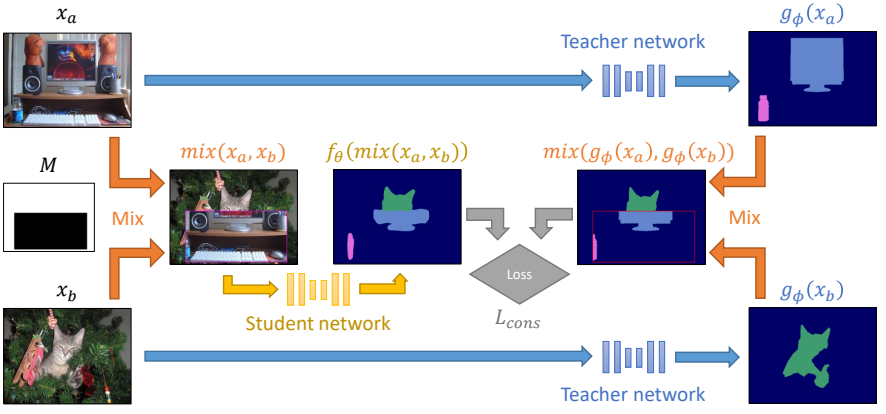

Figure 2: Illustration of mixing regularization for semi-supervised semantic segmentation with the mean teacher framework. f_θ and g_ϕ denote the student and teacher networks, respectively. The arbitrary mask M is omitted from the argument list of function *mix* for legibility.

$$r_{adv} = \varepsilon \frac{g}{||g||}$$

We adopt exactly the same approach, computing the adversarial perturbation that maximises the mean of the change in class prediction for all pixels of the output.

We scale the adversarial radius ε adaptively on a per-image basis by multiplying it by the magnitude of the gradient of the input image. We find that a scale of 1 works well and used this in our experiments. We also tried using a fixed value for ε – as normally used in VAT – and found that doing so caused a slight but statistically insignificant reduction in performance. We therefore recommend the adaptive radius on the basis of ease of use. It is implemented in our source code.

D.2 Illustration of computation of CutMix and Cutout

We illustrate the computation of CutMix based consistency loss L_{cons} in Figure 2 and Cutout consistency loss in Figure 3.

D.3 CutMix with full-sized crops on CITYSCAPES

As stated in our main text, when using the CITYSCAPES dataset, using full size image crops – 1024×512 rather than the usual 512×256 – impairs the performance of semi-supervised learning using CutMix regularization, reducing the mIoU score from $60.34\% \pm 1.24$ to $58.75\% \pm 0.75$. We believe that optimal performance is obtained when the scale of the elements in the mixing mask are appropriately matched to the scale of the image content. We can alleviate this reduction in performance by constructing our mixing mask by randomly choosing three smaller boxes whose area is $1/3$ of that used for one box (the normal case). Given that a CutMix mask consisting of a single box uses a box that covers 50% of the image area (but with random aspect ratio and position), the three boxes each cover $1/6$ of the image

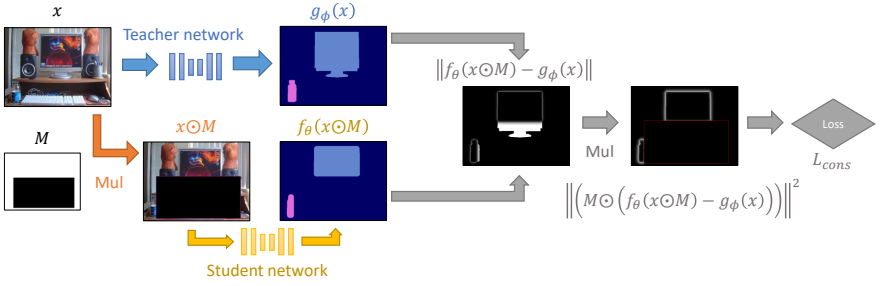

Figure 3: Illustration of Cutout regularization for semi-supervised semantic segmentation with the mean teacher framework. Note that we include additional detail in final steps of the computation of L_{cons} in comparison to Figure 2 in order to illustrate the masking of the consistency loss.

area. The masks for the three boxes are combined using an `xor` operation. Figure 4 contrast mixing with one-box and three-box masks.

D.4 Training details

D.4.1 Using ImageNet pre-trained DeepLab v2 architecture for Cityscapes and Pascal VOC 2012

We use the Adam [6] optimization algorithm with a learning rate of 3×10^{-5}. As per the mean teacher algorithm [14], after each iteration the weights w_t of the teacher network are updated to be the exponential moving average of the weights w_s of the student: $w_t = \alpha_t w_t + (1-\alpha_t)w_s$, where $\alpha_t = 0.99$.

The CITYSCAPES images were down-sampled to half resolution (1024×512) prior to use, as in [5]. We extracted 512×256 random crops, applied random horizontal flipping and used a batch size of 4, in keeping with [10].

For the PASCAL VOC experiments, we extracted 321×321 random crops, applied a random scale between 0.5 and 1.5 rounded to the nearest 0.1 and applied random horizontal flipping. We used a batch size of 10, in keeping with [5].

We used a confidence threshold of 0.97 for all experiments. We used a consistency loss weight of 1 for both CutOut and CutMix, 0.003 for standard augmentation, 0.01 for ICT and 0.1 for VAT.

Hyper-parameter tuning was performed by evaluating performance on a hold-out validation set whose samples were drawn from the PASCAL training set.

We trained for 40,000 iterations for both datasets. We also found that identical hyper-parameters worked well for both using DeepLab v2.

D.4.2 Using ImageNet pre-trained DenseUNet for ISIC 2017

All images were scaled to 248×248 using area interpolation as a pre-process step. Our augmentation scheme consists of random 224×224 crops, flips, rotations and uniform scaling in the range 0.9 to 1.1.

In contrast to [9] our standard augmentation based experiments allow the samples passing through the teacher and student paths to be arbitrarily rotated and scaled with respect to one

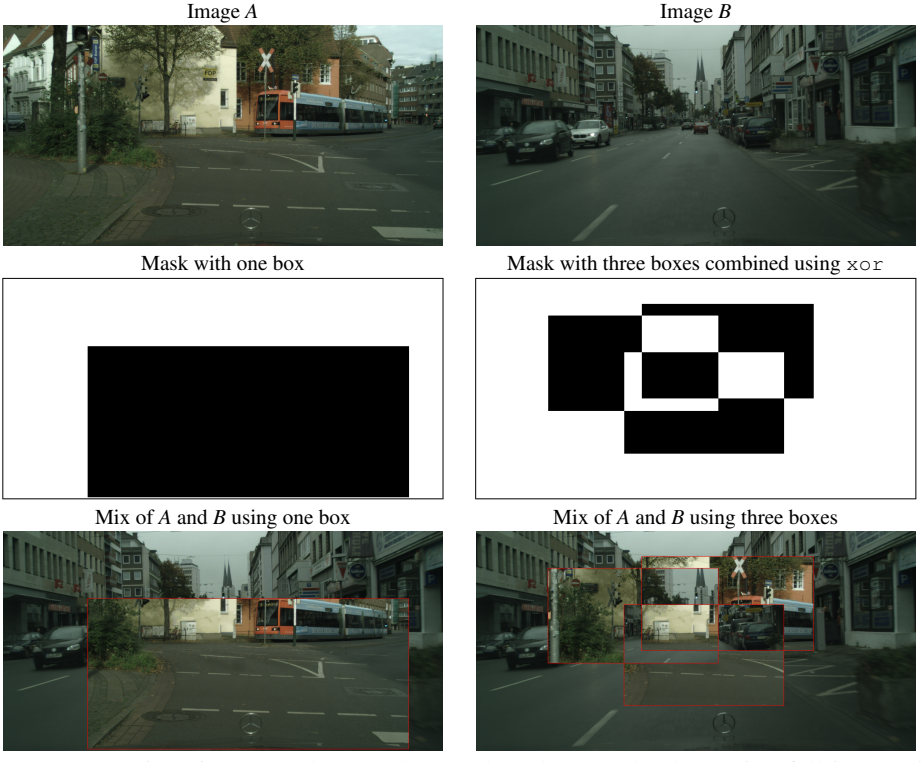

Figure 4: CutMix using a one-box mask vs a three-box mask when using full image size crops from Cityscapes.

another (within the ranges specified above), where as [9] use rotations of integer multiples of 90 degrees and flips.

All of our ISIC 2017 experiments use SGD with Nesterov momentum [43] (momentum value of 0.9) with a learning rate of 0.05 and weight decay of 5×10^{-4}. For Cutout and CutMix we used a consistency weight of 1, for standard augmentation 0.1 and for VAT 0.1.

We would like to note that scaling the shortest dimension of each image to 248 pixels while preserving aspect ratio reduced performance; the non-uniform scale in the preprocessing step acts as a form of data augmentation.

D.4.3 Different architectures for augmented Pascal VOC 2012

We found that different network architectures gave the best performance using different learning rates, presented in Table 2.

We used the MIT CSAIL implementation[1] of ResNet-101 based PSPNet [48]. We had to modify[2] their code in order to use our loss functions. We note that we did *not* use the *auxiliary loss* from [48], known as the *deep supervision trick* in the MIT CSAIL GitHUb repository.

[1] Available at https://github.com/CSAILVision/semantic-segmentation-pytorch.

[2] Our modified version can be found in the `logits-from-models` branch of https://github.com/Britefury/semantic-segmentation-pytorch.

Architecture	Learning rate
DeepLab v2	3×10^{-5}
DeepLab v3+	1×10^{-5}
DenseNet-161 based Dense U-net	3×10^{-4}
ResNet-101 based PSPNet	1×10^{-4}

Table 2: Learning rates used for different architectures, for the Pascal VOC 2012 dataset. All networks used pre-trained weights for ImageNet classification.

D.4.4 Confidence thresholding

[4] apply confidence thresholding, in which they mask consistency loss to 0 for samples whose confidence as predicted by the teacher network is below a threshold of 0.968. In the context of segmentation, we found that this masks pixels close to class boundaries as they usually have a low confidence. These regions are often large enough to encompass small objects, preventing learning and degrading performance. Instead we modulate the consistency loss with the proportion of pixels whose confidence is above the threshold. This values grows throughout training, taking the place of the sigmoidal ramp-up used in [8, 14].

D.4.5 Consistency loss with squared error

Most implementations of consistency loss that use squared error (*e.g.* [14]) compute the mean of the squared error over all dimensions. In contrast we sum over the class probability dimension and computing the mean over the spatial and batch dimensions. This is more in keeping with the definition of other loss functions use with probability vectors such as cross-entropy and KL-divergence. We also found that this reduces the necessity of scaling the consistency weight with the number of classes; as is required then taking the mean over the class probability dimension [14].

References

[1] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] S. Chintala et al. Pytorch. URL http://pytorch.org.

[3] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.

[4] Geoff French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *International Conference on Learning Representations*, 2018.

[5] Wei-Chih Hung, Yi-Hsuan Tsai, Yan-Ting Liou, Yen-Yu Lin, and Ming-Hsuan Yang. Adversarial learning for semi-supervised semantic segmentation. *CoRR*, abs/1802.07934, 2018.

[6] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[7] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, and Carol Willing. Jupyter notebooks – a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87 – 90. IOS Press, 2016.

[8] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2017.

[9] Xiaomeng Li, Lequan Yu, Hao Chen, Chi-Wing Fu, and Pheng-Ann Heng. Semi-supervised skin lesion segmentation via transformation consistent self-ensembling model. In *British Machine Vision Conference*, 2018.

[10] Sudhanshu Mittal, Maxim Tatarchenko, and Thomas Brox. Semi-supervised semantic segmentation with high-and low-level consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[11] Avital Oliver, Augustus Odena, Colin Raffel, Ekin D. Cubuk, and Ian J. Goodfellow. Realistic evaluation of semi-supervised learning algorithms. In *International Conference on Learning Representations*, 2018.

[12] Christian S Perone and Julien Cohen-Adad. Deep semi-supervised segmentation with weight-averaged consistency targets. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 12–19. Springer, 2018.

[13] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[14] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, pages 1195–1204, 2017.

[15] Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. *CoRR*, abs/1903.03825, 2019.

[16] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: https://doi.org/10.1038/s41592-019-0686-2.

[17] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6023–6032, 2019.

[18] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.