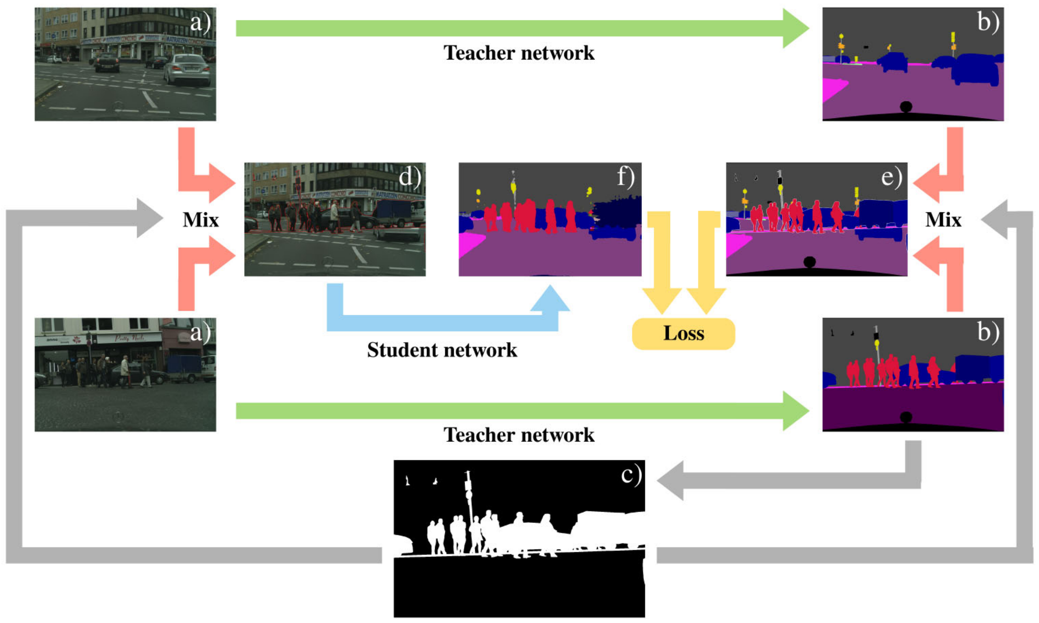


Consistency Regularization for Semantic Segmentation

Segmentation-based mixed sample data augmentation for semi-supervised learning and unsupervised domain adaptation

Master's thesis in Complex Adaptive Systems

Wilhelm Tranheden
Viktor Olsson

Department of Electrical Engineering

Master's thesis 2020

Consistency Regularization for Semantic Segmentation

Segmentation-based mixed sample data augmentation for semi-supervised learning and unsupervised domain adaptation

Wilhelm Tranheden
Viktor Olsson

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal Processing and Biomedical engineering
Chalmers University of Technology
Gothenburg, Sweden 2020

Consistency Regularization for Semantic Segmentation
Segmentation-based mixed sample data augmentation for semi-supervised learning and unsupervised domain adaptation
Wilhelm Tranheden
Viktor Olsson

Supervisor: Oskar Karnblad, Volvo Cars
Supervisor: Saudin Botonjic, Volvo Cars
Supervisor: Lennart Svensson, Department of Electrical Engineering
Supervisor: Juliano Pinto, Department of Electrical Engineering
Examiner: Lennart Svensson, Department of Electrical Engineering

Master's Thesis 2020
Department of Electrical Engineering
Division of Signal Processing and Biomedical engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Schematic showing a summary of the unsupervised pipeline of our proposed semi-supervised learning algorithm.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2020

Consistency Regularization for Semantic Segmentation
Segmentation-based mixed sample data augmentation for semi-supervised learning and unsupervised domain adaptation
WILHELM TRANHEDEN
VIKTOR OLSSON
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The idea of using unlabeled- or synthetic data for training deep learning models is becoming increasingly important as the need for more data grows and computational resources improve. This master's thesis deals with the case of semantic segmentation, a field in which the cost of annotations are particularly high, thereby rendering the need for data-efficient solutions paramount.

In order to exploit unlabeled data, a frequently used class of algorithms is semi-supervised learning, where in addition to a supervised signal the model also learns from unlabeled data. A common strategy to this is consistency regularization, where the model is encouraged to make consistent predictions over different perturbations of unlabelled data points. A key challenge is that common augmentations used in semi-supervised classification have proven less effective for semantic segmentation. In some scenarios, the labeled and unlabeled data originates from different distributions, such as when learning from synthetic data. When no ground truth labels are accessible for the actual domain of interest, this problem is called unsupervised domain adaptation. This domain shift can cause methods to require adjustments, as they behave differently compared to application on semi-supervised learning tasks.

We propose a novel data augmentation scheme, that we call ClassMix, which leverages on the fact that, in semantic segmentation, we can find accurate silhouettes of objects in an image. Using this, objects can be cut out of one image and pasted onto another, forming new, strongly augmented, samples. We use the ClassMix augmentation strategy in semi-supervised semantic segmentation based on consistency regularization, obtaining, to the best of our knowledge, state-of-the-art results, with a margin of up to 3%.

We also use the ClassMix augmentation for unsupervised domain adaptation by, in a modified pipeline, mixing samples across domains. To the best of our knowledge, we outperform all existing methods on two common synthetic-to-real semantic segmentation benchmarks, obtaining the state of the art with a margin of up to 2%.

Keywords: semantic segmentation, consistency regularization, mixed sample data augmentation, semi-supervised learning, unsupervised domain adaptation

Acknowledgements

We would like to thank our academic supervisor Lennart Svensson for his great commitment to helping us develop our ideas throughout the thesis project. During the last weeks of the project we have also received excellent help from Juliano Pinto, something we are very grateful for. Our two supervisors at Volvo Cars, Oskar Karnblad and Saudin Botonjic, have been really helpful and have been able to accommodate us whenever we have had any issues, for which we are very thankful. Finally, in addition to making the thesis possible, the machine learning team as well as Volvo Cars as a whole have been very welcoming and engaging, making our project run smoothly.

Wilhelm Tranheden
Viktor Olsson
Gothenburg, June 2020

Contents

List of Figures

List of Tables

1 Introduction

Computer algorithms are great at handling problems such as sorting huge lists or solving difficult numerical calculations, which are near impossible for humans to solve. More complex problems, however, quickly become very difficult for traditional algorithms. Such is the case for, for instance, natural language processing and computer vision, where computers have struggled with tasks that are trivial for humans. However, with the rapid progress of machine learning, this is no longer the case, and computers now outperform humans in a quickly growing number of new tasks. The reason for this progress is, largely, because of the success of deep learning; the process of providing a deep neural network with data, such that the algorithm itself learns to perform the task at hand. Deep learning is the foundation for recent technical progress such as smart AI assistants, self-driving cars and computers beating humans in games that have previously been seen as needing inherently human capabilities.

A major bottleneck in machine learning research and application is the need for large amounts of annotated data. This requires humans to spend a lot of time looking at and labeling data, a process which is both time consuming and expensive. The need for more data-efficient solutions in machine learning is therefore paramount. This problem can be approached in many different ways, such as carefully picking the most informative data samples possible to label, augmenting the annotation process such that the human effort is smaller, or using unlabeled data to assist training. In this thesis we regard the latter scenario. Specifically, we investigate two different classes of techniques, namely Semi-Supervised Learning (SSL) and unsupervised Domain Adaptation (UDA).

The most common type of machine learning is called supervised learning, in which all data samples have one or many labels, and a model is trained to predict these labels for all samples. In semi-supervised learning, on the other hand, one also includes data that has not been labeled. The purpose of this is to increase the performance of the model without having to provide more annotations, or equivalently, to obtain a certain performance with as few labels as possible. Since unlabeled data is often considerably easier to get than labeled data, semi-supervised learning can be a powerful way to train successful models at a lower cost.

Another approach to solving the problem with expensive annotations is unsupervised domain adaptation. Here, one deals with the case where the target dataset has no labels available whatsoever. Instead, the training of the model is aided by a second source dataset, often similar to the first, which is labelled. This idea is suitable if the target dataset does not have any labels and if these would be hard to get by, whereas there is another dataset available that shares features with the first, and that is already labeled. Unsupervised domain adaptation is often a more challenging problem than semi-supervised learning. However, the possible gain might be larger as no labels at all are required from the target dataset.

The field that we are handling in this thesis is the computer vision task semantic segmentation. It is the task of assigning each pixel in an image one of several predetermined classes, such as for instance car, person or tree. Because of the high level of accuracy, semantic segmentation is useful in applications where detailed understanding of images is important, such as autonomous driving, robotics and analysis of medical images. The high level of detail, however, also comes with the downside that images are very time consuming to label. One example is the commonly used Cityscapes dataset [2], where each image required on average 1.5 hours to label, or medical image segmentation, where images have to be labeled by highly trained medical practitioners. For this reason the study of data efficient solutions, such as semi-supervised learning and unsupervised domain adaptation, is especially important for semantic segmentation.

In this thesis we use an approach called consistency regularization in order to take advantage of unlabeled data in semi-supervised learning and unsupervised domain adaptation. We propose a novel data augmentation strategy, ClassMix, in which new samples are created by mixing two images, such that some semantic classes are cut from one image and pasted onto the other image. For semi-supervised learning, we mix unlabeled images by basing the mixing on the predictions of the network, exploiting the fact that the network learns to predict a pixel-level semantic map of the images. For unsupervised domain adaptation, we introduce the notion of mixing images across domains and instead base the cut on labeled images from the source dataset. In both settings, the predictions on mixed images are subsequently trained to be consistent with predictions made on the images before mixing. Our proposed method is evaluated on established benchmarks, and an ablation study is included to further analyze the solutions.

1.1 Problem statement

This thesis aims to investigate how unlabeled samples can be used in the computer vision task semantic segmentation. This is relevant because semantic segmentation is necessary in many fields where a highly detailed understanding of an image is crucial, such that image classification or object detection is not sufficient. Because of the highly detailed output, it is also necessary that the data annotations are detailed, which means that labeling becomes a costly process, motivating the investigation of data efficiency in particular. Specifically, two classes of techniques are considered, namely semi-supervised learning and unsupervised domain adaptation. Both being highly active research fields with promising results. Potential application areas are many, with one of the most prominent ones being autonomous vehicles.

Given certain amounts of labeled data from several commonly used public datasets, we have developed methods with the purpose of obtaining as high performance as possible using deep neural networks. In particular, our focus has been on a class of techniques called consistency regularization, and more specifically on utilizing augmentation strategies known as mixing, where two samples are combined to create new, highly perturbed, samples.

1.2 Scope

This thesis builds on consistency regularization developed for semi-supervised learning and unsupervised domain adaptation in the last few years. An extensive study of related works in this field has been performed and ideas from many different sources have been explored and adjusted to our domain. In addition to this, several novel ideas have been developed and evaluated during the project and compared to state-of-the-art methods for common benchmarks. In order to give fair comparisons with existing works, we kept our settings as close to these works as possible in terms of parameters and models. Therefore, no optimization has been performed on any aspects that are not directly connected to our own developed methods (e.g. network architecture). This implies that we did not aim at obtaining the absolute strongest possible results for the tasks that we are considering, but rather to obtain the best results possible relative to the existing works that we consider to be the most relevant comparisons.

Our developed methods are applicable to any semantic segmentation data, however, we have confined ourselves to a a few datasets. The investigated datasets are however the most common benchmarks in research, which is why we do not consider this limitation to be problematic.

1.3 Contributions

The main contributions from this master thesis project are:

- We introduce a novel augmentation strategy for semantic segmentation, which we call ClassMix. In this technique two images are mixed intelligently by basing the mixing on the semantics of the images.
- We develop a unified framework for semantic segmentation, combining consistency regularization and pseudo-labeling.
- We introduce the notion of mixing samples from two different domains in unsupervised domain adaptation.
- We analyze the relation between similarity within a dataset and the performance of mixing augmentations.

Using our developed techniques we contribute to the scientific community by obtaining new state-of-the-art results for several datasets, in both semi-supervised learning and unsupervised domain adaptation. These results have resulted in two papers, one about using ClassMix for semi-supervised semantic segmentation [7], and one about the mixing of images across domains, for unsupervised domain adaptation [8].

1.4 Related work

Since the output in semantic segmentation is a semantic map where positional information is paramount, fully convolutional neural networks are generally most successful, since there is a close correspondence between pixels in input and output space. Network architectures such as DeepLab [9] and PSPNet [10] have obtained strong results on important benchmarks, and are frequently used in research.

A common way of using unlabeled samples in semi-supervised learning is by training a model to make the same prediction for different perturbations of a sample. This is what is known as consistency regularization, and it has proven to be very successful in image classification [11, 12]. Sohn et al. showed that training benefits from using very strong image augmentations [13]. One class of strong augmentation techniques is mixed sample data augmentation, where new samples are created by combining two or more existing images, either by interpolation or by having some pixels of the new image coming from one sample and some pixels coming from another image. One such technique is CutMix [14], where a rectangular region is copied from one image and pasted on top of another, forming a new, strongly perturbed sample.

Consistency regularization has, until recently, not been applied to semantic segmentation of natural images, when French et al. trained for consistency over images augmented using CutMix [15]. This resulted in state-of-the-art results for several important semi-supervised learning semantic segmentation benchmarks, showing that consistency regularization indeed can be a successful method for semantic segmentation.

The solutions used in unsupervised domain adaptation for semantic segmentation are often similar to those used in semi-supervised learning, as both domains handle training using a combination labeled and unlabeled data. Approaches based on consistency have been used with some success [16, 17], however only very recently. Another technique common in both semi-supervised learning and domain adaptation is entropy minimization. The idea there is to train a model to give predictions with high confidence, i.e. low entropy. One such method is pseudo-labeling [18], where the network's own predictions are used as targets. Pseudo-labeling has been used in combination with consistency regularization with positive result for image classification [13]. However, to the best of our knowledge, we are among the first to combine them in the setting of semantic segmentation.

2

Theory

This chapter covers the theory necessary for our proposed solution. We give an explanation of some crucial parts of deep learning. However, some prior knowledge is assumed, namely what deep neural networks are and how they are trained. Semantic segmentation is explained in more detail, as well as semi-supervised learning and unsupervised domain adaptation, along with an account for related work. Lastly, we walk through the techniques of consistency regularization that we have used.

2.1 Deep Learning

Traditional computer algorithms rely on a set of explicit instructions, such that when it gets a certain input, it will perform a set of actions. These instructions will have been given to the program by a human programmer. A different approach to doing things, however, is using machine learning, in which a model is fed data, and from that learns to perform a task, without being explicitly told what to do. Machine learning approaches have proven to be very useful for many tasks that are otherwise too complex. If this model is an artificial neural network, where several layers of neurons are connected to make subsequent calculations, we arrive at Deep learning. We will here briefly explain a few concepts of deep learning that are relevant to this report, however, some previous knowledge of the field is assumed.

2.1.1 Loss function

The loss function of a deep learning model is the objective function one tries to minimize during training. Hence, it is the most important component in deciding in which direction training progresses, and therefore an integral part of any deep learning model. Most commonly, the loss function is a distance measure between the output of a network $p(x)$ and the corresponding target label y. This distance measure could be any differentiable function, often used functions include the sum of squared error

$$\mathrm{SSE}(p(x), y) = \sum_{i \in C} (y_i - p(x)_i)^2, \tag{2.1}$$

and Cross entropy

$$\mathrm{H}(p(x), y) = -\sum_{i \in C} y_i \log p(x)_i, \tag{2.2}$$

where the sums are over all classes C.

2.1.2 Optimization

As a deep neural network is a very complex model, optimizing its parameters θ analytically to minimize the loss function $L(\theta)$ is not possible. Hence, a numerical optimization algorithm is required. The most straightforward such algorithm is gradient descent, where one calculates the gradient $\nabla_\theta L(\theta)$, and update the parameters according to

$$\theta_{k+1} = \theta_k - \epsilon \nabla_\theta L(\theta_k), \tag{2.3}$$

where ϵ is the learning rate and k is the training iteration. Since the training set used to train a deep learning model can be very large, it often becomes unfeasible to use the entire dataset to calculate this gradient, however. The most used optimization algorithm solving this problem is Stochastic Gradient Descent (SGD).

Here, only a small subset of the entire dataset is used, i.e. mini-batches, to calculate an estimate of the gradient in equation 2.3, and taking many such smaller steps, instead of fewer large steps based on all the data. It is important to note that the mini-batches are sampled randomly from the pool of data, with a new sampling each iteration. This method has the added effect of introducing noise in the optimization, giving the training a regularizing effect [19].

A further improvement to the optimization can be achieved by introducing momentum [20]. Using momentum, a weighted average of the gradients from previous iterations is added to the current gradient update, causing the optimization process to be more smooth and less sensitive to noise. This effect from previous gradients is contained in a velocity parameter v, defined by the recursive equation

$$v_{k+1} = \alpha v_k - \epsilon \nabla^*_\theta L(\theta_k), \tag{2.4}$$

where α is the momentum parameter with a value between 0 and 1, and ∇^*_θ is the estimated gradient.

An issue when training deep neural networks is that the size of parameters sometimes explodes, causing generalization problems. To remedy this issue, a common solution is to add a regularizing term to the optimization algorithm, called weight decay. What that does is penalize the model for having parameters with large values, limiting their size. In practice, the most common way to do this is to add a weighted L_2 norm, $\frac{\lambda}{2}\theta^2$, to the loss function, where λ is the weight decay parameter. Taking the gradient of the weight decay term yields the term $\lambda\theta$.

Combining the stochastic updates from using mini-batches, momentum and weight decay we arrive at the parameter update rule

$$\theta_{k+1} = \theta_k - \epsilon \nabla^*_\theta L(\theta_k) + \alpha v_{k+1} - \epsilon \lambda \theta_k. \tag{2.5}$$

When using SGD, or another optimizer based on the gradient from a subset of all data samples, it is often important to gradually decrease the learning rate as training progresses [21]. The reason for this is that when using a subset of samples to calculate the gradient, like in SGD, we introduce noise that does not vanish when a minimum in the loss function is reached, meaning we never reach convergence. As a contrast, the true gradient of the loss function will be zero in a minimum. However, when calculating the gradient from only a few samples, it might not be zero.

Learning rate schedules used in practice often take into account a maximum number of iterations, after which the learning rate is constant, possibly zero (i.e. training stops). For semantic segmentation, it has been shown that a polynomial learning rate schedule,

$$\epsilon_k = \epsilon_0\Big(1 - \frac{k}{k_{max}}\Big)^p, \tag{2.6}$$

with the power p set to 0.9, works well [22, 23]. ϵ_0 is the base learning rate and k_{max} is set to the number of iterations the model is trained for. This schedule has been applied in previous semi-supervised semantic segmentation research [24, 25].

2.1.3 Architecture: ResNet

To increase the capacity of a deep neural network, a powerful alteration is to increase the number of layers in the network, making it deeper. This greatly increases the representative capability of the network, but at the same time makes the training process more difficult. This is largely because of the vanishing/exploding gradient problem, where gradients for parameters early in the network disappear/explode as they are a product of gradients from all later layers in the network. This problem was in part solved by He et al., with the introduction of ResNet [1]. They noted that representing a function $H(x)$ is equivalent to representing the residual function $F(x) := H(x) - x$, and to that adding the identity function x, getting $F(x) + x$. The authors hypothesised that a residual mapping is easier to optimise than the original function, which they also proved empirically to be the case. In the extreme, one can consider the case of mapping the identity function, it would be easier to push parameters to zero such that $F(x) = 0$ than to fit the identity function $H(x) = x$. This type of mapping is performed by utilising skip connections, as is shown in figure 2.1. These skip connections can skip any number of layers, in the original paper they skip two or three layers [1].

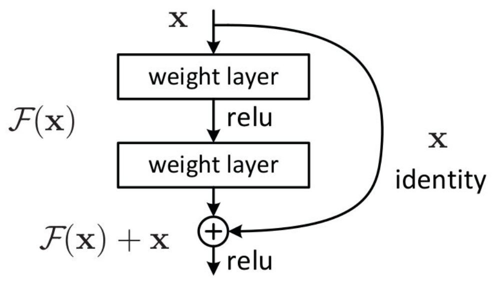


Figure 2.1: The main building block of ResNet. Skip connections are added on top of layers in a normal feedforward neural network to help the optimization process, making it possible to train deeper neural networks. Image taken from [1].

2.2 Semantic segmentation

The most well known, and most studied, task in computer vision is image classification, where the objective is to determine which of several predetermined classes

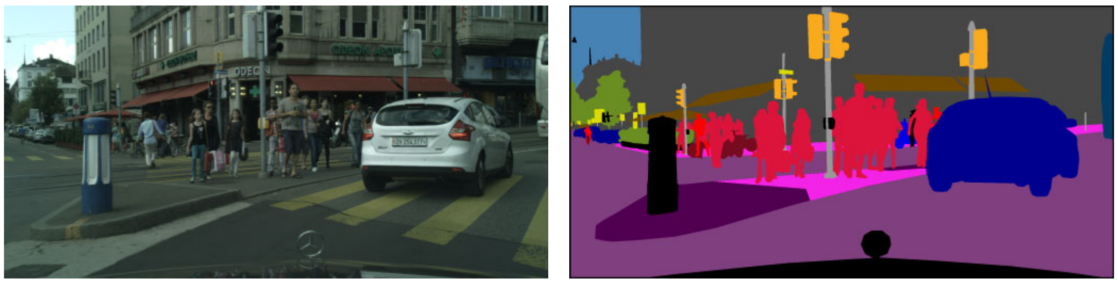


Figure 2.2: An example of an image and its corresponding semantic map. Each pixel is given a classification. This particular frame is from the Cityscapes dataset [2].

an image belongs to. These classes could be, for example, person, car, or dog. A related task, which is more detailed than image classification, is semantic segmentation, which is what this Master thesis is about. It is a task in which the objective is to classify every pixel in an image. So in contrast to image classification, a semantic segmentation model will have as many outputs as there are input pixels, whereas image classification only gives one output. Hence, the result is a lot more detailed. Figure 3.5 shows an example of what semantic segmentation looks like, to the left is the image, and to the right is the corresponding semantic map, or label. The semantic map is the same size as the image, and each pixel contains information about which of a number of predetermined classes this pixel belongs to.

2.2.1 DeepLab

There are many different network architectures specialised for semantic segmentation, one of which is the DeepLab framework [9, 23]. It is a fully convolutional network architecture with a backbone such as ResNet [1] or VGG [26], but with some alterations to make the network better suited for semantic segmentation. In some layers, normal convolutions are exchanged for Atrous convolutions, and in DeepLabv2 the last layer is made into an Atrous Spatial Pyramid Pooling module. These two methods are described below. After classification by the network, the output in DeepLab is also run a few iterations through a conditional random field. This has not been used in our work, however, and will therefore not be further explored here. The authors also propose multi-scale processing of images, which they show improve results. However, this is computationally intensive and is therefore also not used or explored further in this thesis.

Discrete convolutions are a frequently used component in neural networks, especially for image analysis tasks where they have played a huge part in recent success. Convolutional layers in a neural network stand in contrast to fully connected layers, where each output neuron is connected to each input neuron, which yields a large number of parameters and does not take spatial dependence into account. In a convolutional layer, on the other hand, each output neuron is the result of an element-wise multiplication between a kernel and only a few neurons in a limited spatial area in the previous feature map. This kernel is moved across the input feature map, reusing the same weights, meaning that the layer is shift invariant. An

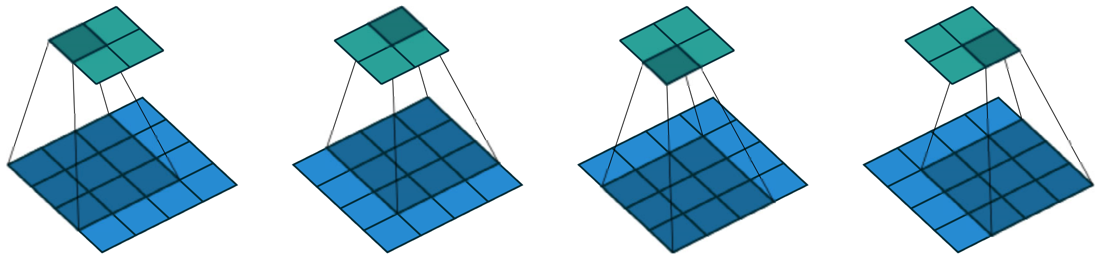


Figure 2.3: Illustration of a standard convolution. The blue map is the input and the cyan map is the output. A kernel, here of size 3×3, is moved over and element-wise multiplied with the input to create the output. Image taken from [3].

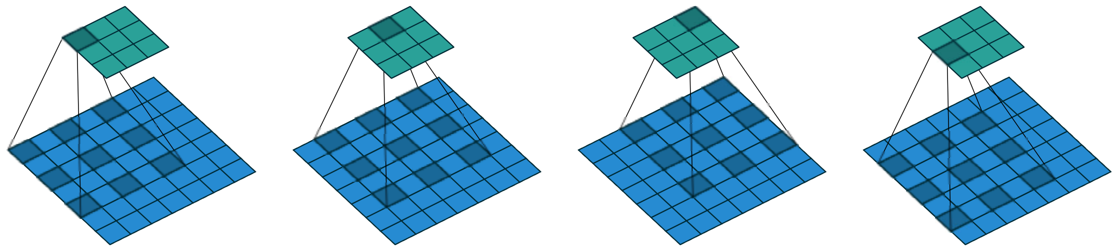


Figure 2.4: Illustration of an atrous, or dilated, convolution. The blue map is the input and the cyan map is the output. Here, there is a hole with size 1 between each element in the convolution kernel. In this example, $K = 3$ and $r = 2$. Image taken from [3].

illustration of a convolution is shown in figure 2.3. Using convolutional layers allows the network to better localise and identify spatial features.

Atrous, or dilated, convolutions are a type of convolutions where the kernel is not dense. Instead, holes (trous in french), or zeros, are introduced between the different kernel elements. The idea of atrous convolutions was originally developed for the Algorithme à trous for the undecimated wavelet transform [27], and later applied to neural networks [28]. The 2-D atrous convolution is mathematically described by

$$y(i,j) = \sum_{k=1}^{K}\sum_{l=1}^{K} x(i + r \cdot \lfloor k - \frac{K}{2} \rfloor, j + r \cdot \lfloor l - \frac{K}{2} \rfloor) \cdot w(k,l) + b. \tag{2.7}$$

Here, $y(i,j)$ is the output and $x(i,j)$ is the input at index (i,j), K is the size of the kernel, r is the rate parameter which decides the size of the holes and w and b are weights and biases respectively. A standard convolution is obtained by setting $r = 1$. This equation is correct for odd K but needs to be adjusted slightly for even kernel sizes. If the input is two-dimensional, $x(i,j)$ and $w(k,l)$ are scalars, if it is three-dimensional they are instead vectors. An illustration to show how an atrous convolution differs from a standard convolution is shown in figure 2.4.

Using atrous convolutions rather than standard convolutions means that the effective kernel size and the size of the receptive field increases, without increasing the number of parameters, which would hurt efficiency. Since the filters can be made arbitrarily big by introducing holes of different sizes, the method becomes a powerful way to control spatial resolution.

A difficulty in image analysis is handling objects with different sizes. A model could be able to identify e.g. a car perfectly if it is a certain size, but making it a different size could dramatically hinder the model's ability to identify the same car. A common solution to this problem in Deep convolutional neural networks is to present to the model several rescaled versions of the same image and in the end aggregate the results, also known as multi-scale processing. This method is, however, computationally intensive, as each rescaled image has to be individually processed by the model. The authors of DeepLabv2 propose another method to solve the same problem, which they call Atrous Spatial Pyramid Pooling (ASPP) [23]. Here, features are sampled in parallel branches at different rates with atrous convolutions, for one layer of the network. The outputs of the different branches are then fused to create the final result. A schematic of ASPP is shown in figure 2.5, with four parallel atrous convolutions with rates 6, 12, 18 and 24, whereafter the feature maps are added.

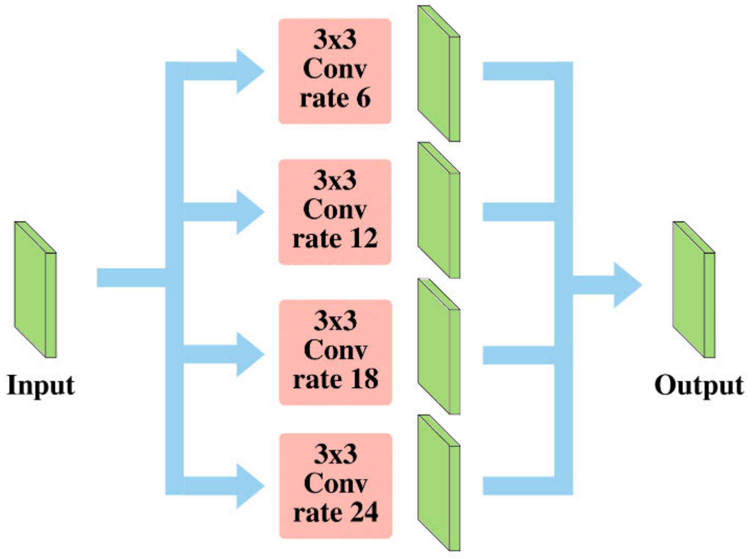


Figure 2.5: An illustration of Atrous Pyramid Pooling (ASPP). Four parallel atrous convolutions with different rates are applied on an input feature map, after which the maps are summed to create the final output.

2.3 Supervised and semi-supervised learning

Semantic segmentation is commonly done in a supervised learning setting. In supervised learning one has access to a joint distribution $p(\mathbf{x}, \mathbf{y})$ of images $\mathbf{x}$, and ground truth semantic maps $\mathbf{y}$ to sample from. For classification the analysis is identical but with $\mathbf{y}$ having a scalar value. The objective of supervised learning can be formulated as maximizing the expectation of the conditional probability

$$\mathbb{E}_{\mathbf{x},\mathbf{y}\sim(\mathbf{x},\mathbf{y})}[\log p(\mathbf{x},\mathbf{y})]. \tag{2.8}$$

This is accomplished by sampling points uniformly, comparing the network's prediction to the ground truth by some loss function as described in section 2.1.1, and updating the network's parameters with stochastic gradient descent, section 2.1.2.

For semi-supervised learning we instead have two datasets; one labeled $D_l = p(\mathbf{x},\mathbf{y})$ and one unlabeled $D_u = p(\mathbf{x})$. An assumption is made that the distribution D_u is the marginal distribution of D_l, i.e. the images $\mathbf{x}$ are from the same underlying distribution, though in practice this can not always be guaranteed. For urban scene segmentation for instance, the unlabeled data could come from a different geographic location or have different weather conditions. The objective of semi-supervised learning is identical to that of supervised learning, but the unlabeled data is now simultaneously being used to constrain decision boundaries and learning more general representations. In this sense semi-supervised learning is much closer to supervised than unsupervised learning, where the goal is often more general representation learning. The motivation for semi-supervised learning comes from the fact that collecting labeled data is often costly and time-consuming, in contrast to unlabeled data which is usually cheap.

2.3.1 Related work in semi-supervised learning

Two common classes of methods for SSL are self-training, where a model trains against its own predictions [18], and consistency regularization, where a model is trained to make consistent predictions for various perturbations of a sample. These perturbations can be of various forms, such as adversarial [29], dropout [30], or augmentations [11, 12]. Berthelot et al. showed the usefulness of combining self-training and consistency regularization [31, 13], which is also essential for our proposed method. Related to this, Laine and Aila leverage the fact that an ensemble of models often gives more accurate predictions than a single model. By averaging the predictions for unlabeled samples over several epochs, and using these as targets [5], they achieve more stable targets. Tarvainen and Valpola further develop this idea by instead keeping a second 'teacher' network with parameters being a moving average of the standard network's parameters, which is less memory demanding [4].

Similar to classification, semi-supervised semantic segmentation has gotten a lot of attention lately, and since the labeling of images for segmentation is very time consuming, successfully employing SSL can be very useful. Though successfully applied for medical segmentation [32, 33], consistency regularization for semantic segmentation of natural images has until recently struggled to compete with other approaches, such as those based on adversarial learning [24, 25]. In semi-supervised image classification, however, consistency regularization has shown promising results, something that researchers have attributed to the cluster assumption [34]. This assumption states that different classes are, in some feature space, clustered in individual regions, with the boundaries separating classes lying in regions with a low density of samples. In semantic segmentation, there are no such low-density regions separating classes, meaning that the cluster assumption is violated. This is because the receptive fields on the input layer do not change more when moving over object boundaries, than within objects, which was shown by experimental results of previous researchers [15, 35]. The violation of the cluster assumption has made researchers try to either apply perturbations on the encoder's output [35] where the cluster assumption was shown to be upheld or to use non-isotropic perturbations such as image augmentations [15].

Recent success in consistency regularization for classification has shown that very strong data augmentations are key [13]. This may further exacerbate the difficulties of semi-supervised semantic segmentation, as information about class-affiliation on a pixel level is easier to destroy than for the whole image. One form of augmentation that has proven successful in image classification and recently also in semantic segmentation is so-called mixed-sample data augmentations. What makes this kind of augmentation especially interesting for this work is its key role in augmentation-based consistency regularization for semi-supervised learning [12, 31, 36, 37]. One such technique is CutMix, introduced by Yun et al. [14]. They mix images using a binary mask, such that a resulting image has a set of pixels coming from one image, and the rest coming from another. For semantic segmentation, CutMix has recently been used to obtain state-of-the-art results in semi-supervised learning [15].

Some works also try to exploit the use of weak annotations in a setting called weakly supervised learning [38, 39, 35]. For this project, we operate strictly in the semi-supervised setting.

2.4 Unsupervised domain adaptation

Another way to get around the fact that labeled data might be hard to come by is to train a model using data from a different dataset than the one that the model will later make predictions for. For semi-supervised learning, we make the assumption that the distribution of unlabeled data D_u is the marginal distribution of the labeled data D_l. Unsupervised domain adaptation covers the case when this assumption is violated, and the targeted distribution is the unlabeled one. For our thesis, we cover the challenging case of when the labeled data is synthetic. This is a very powerful approach, as labels might be easier to obtain for one domain than for another, with an obvious example being that already labeled datasets can be reused when training for a new target dataset. When pretraining a model using one dataset and then fine-tuning the model using the target dataset, this is exactly what is being done. Normally, it is however still necessary to have at least a few labels for the target dataset.

When no such labels are available, we arrive at unsupervised domain adaptation. The naive way of doing this would be to train a model on the labeled source dataset and then use it to make predictions on the target dataset. This, however, rarely works well, as there is almost always a domain gap between the source and target datasets, meaning that the model will learn patterns that are not present or not applicable in the target data. There is, therefore, a need to teach the learning system to adapt to this domain gap.

A common scenario for domain adaptation is when plenty of labeled synthetic data is available, which can often be supplied cheaply. The data can come in the form of an existing labeled dataset, or it could be dynamically generated [40]. This synthetic data can be used to generate novel and diverse training examples, but it usually has a severe domain gap relative to natural data as it is not quite realistic and perfect alignment with real-world statistics is immensely difficult. For a more comprehensive study of the use of synthetic data in Deep Learning, see [41]. Needless to say, being able to effectively transfer knowledge from synthetic data to natural data would be a huge help in training deep learning models.

2.4.1 Related work in unsupervised domain adaptation

Previous work in unsupervised domain adaptation for semantic segmentation can be categorised similarly to in semi-supervised learning, though the methods differ in their details. Predominant approaches include adversarial, self-training and consistency based approaches. Adversarial learning focuses on aligning the distributions of the two domains at different levels such as at the pixel level [42, 43, 44], feature level [45, 46, 47] or semantic level [48, 49]. Self-training methods instead try to train directly on the target data by the use of pseudo-labels, further explained in section [18]. Variants have focused on creating more useful pseudo-labels by specialised sampling procedures [50], or accounting for uncertainty [51, 52]. Recently, consistency based approaches have been used as well [16, 17]. These have been based on consistency criteria in regards to aligning distributions on a pixel-level in contrast to semi-supervised learning, where consistency over traditional augmentations, as mentioned earlier, are common.

2.5 Consistency regularization

When training a neural network in a supervised manner one usually minimizes a loss function L_l on labeled samples using an optimization procedure such as stochastic gradient descent, sections 2.1.1 and 2.1.2. Consistency regularization adds an auxiliary loss $L_u = d(f_\theta(\hat{x}_1), f_\theta(\hat{x}_2))$, where d is some distance measure such as Cross entropy or Mean squared error, f_θ is the function of the neural network parameterized by its weights θ and $\hat{x}_1$ and $\hat{x}_2$ are two different perturbations of the same data point x. Meaning that the objective is to minimize the difference in output from $\hat{x}_1$ and $\hat{x}_2$. This requires no ground truth label and allows unlabeled data samples to be used to constrain the decision boundary of the network by enforcing consistency over various perturbations.

Consistency regularization is used frequently in research, in the setting of image classification it has several times pushed the state-of-the-art [12, 11, 31, 13], and for semantic segmentation, it has recently also seen success [15, 53]. In this section, we describe some of the techniques used in existing works that are relevant to our proposed method.

2.5.1 Augmentation Anchoring

Recent success in consistency based semi-supervised learning owes in large part to the application of more diverse, and especially stronger, augmentation policies [13, 11]. A key technique that has emerged, augmentation anchoring [31], requires two different sets of augmentation policies. One policy of weak augmentations, and one policy of stronger (more difficult) augmentations. The weak augmentation policy is applied to labeled samples in the supervised part of the training, as well as to the unlabeled samples. The strong augmentation policy is applied to the same unlabeled images. Both variants of the unlabeled images are then passed through the network. The predictions from the weakly augmented samples are then treated as labels, and the gradient is only allowed to flow backwards from the predictions on the strongly augmented samples. This encourages the network to make the same predictions on the strongly augmented samples as on the weakly augmented samples, which is intuitively reasonable, as the predictions on the weakly augmented samples are more likely to be correct. This has previously been combined with some form of temperature sharpening [12, 11] and pseudo-labeling [13] (explained below) of the prediction from the weakly augmented sample, which has been motivated to enforce entropy minimization.

2.5.2 Mixed Sample Data Augmentations

To achieve more robust results, especially despite of violation of the cluster assumption present in semantic segmentation, section 2.3.1, French et al. [15] argues that strong, varied perturbations are required.

One can also observe that the risk of altering semantic information is higher for dense prediction tasks. For classification, it is enough simply to be able to identify the class of an object in an image, even if information of its position is lost by strong augmentations. For semantic segmentation, this is not true, as each pixel has to be able to be classified, and strongly augmenting an image might alter the classes of pixels, whereas the semantic information of the entire image is not likely to change as drastically. This fact limits the strength of augmentations and thus the potential use of augmentation anchoring in this setting. Augmentation techniques such as cropping also come with difficulties, unlike classification where this augmentation has played an important role for semi-supervised learning [13] and contrastive learning [54], consistency for predictions over crops can only be measured on the union of the two cropped images, decreasing the training signal from samples, requiring more training.

Mixed sample data augmentations are augmentations that mix two or more samples, and in that way extends the dataset. In the case of the samples being images, as in semantic segmentation, this means creating new images where not all pixel values are from the original image, but are mixed with pixel values from other images. This could, for example, be done by having a set of the pixels in the mixed image coming from one original image, and the rest of the pixels coming from another image. For classification, class-labels have to be interpolated in accordance with the original images, while for semantic segmentation the corresponding semantic maps can simply be identically mixed. Mixed sample data augmentation strategies that have seen success in recent consistency-based semi-supervised learning algorithms for classification [12, 13, 31, 11, 14], offer a promising approach for semantic segmentation, as they strongly alter the contents of the receptive fields in a varied way [15]. Another potential benefit is that unlike for classification tasks, mixed sample data augmentations for semantic segmentation not only offer more reasonable interpolation of labels for supervised learning, but the interpolations in the output space also become more varied (especially with more complex masks) allowing for consistency to be enforced across images with drastically different semantics. In contrast to this, traditional augmentation techniques such as translation, rotation and simple color distortions, while creating variation in the input space, provides considerably less variation in the output space for semantic segmentation.

In this project, we have experimented with two previously used mixed sample data augmentations, or mixing algorithms, namely CutMix [14] and CowMix [36], as well as our own, novel approach, which we call ClassMix. Below, we go through CutMix and CowMix, and ClassMix is explained in section 3.1. All three of these techniques are examples of what can be described as mask-based mixing. This type of mixing stems from regional dropout strategies, which is a form of regularization in which parts of the input are being left out, meaning that a subset of all pixels is made black (or removed in some other way). This technique can be called mask-based erasure, since the operation can be quantified by multiplying the image with a binary mask of the same size as the image,

$$x' = M \odot x. \tag{2.9}$$

Here, M is the binary mask, x is the original image, and $\odot$ is the element-wise product. This will produce an image x' that is identical to x for the pixels where M is one, and zero for the pixels where M is zero. Removing patches in this way creates a varied perturbation, but since many pixels are set to zero, and hence not useful for training, each image contains less information, which leads to less efficient training.

A solution to this is to, instead of setting the masked out pixels to zero, replace them with the corresponding pixels from a second image, forming a mixed image, which was proposed by Yun et al. in the CutMix algorithm [14]. This technique can be called Mask-based mixing and has been a core part in achieving state-of-the-art results in semi-supervised semantic segmentation [15, 53]. It can be formalized by

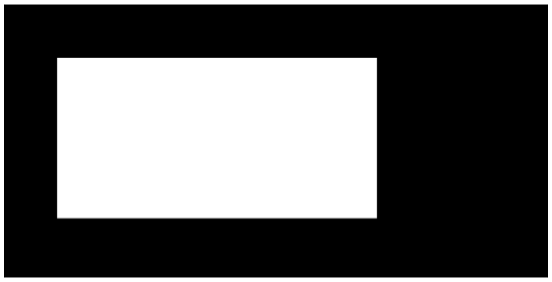
$$x' = M \odot x_1 + (1 - M) \odot x_2, \tag{2.10}$$

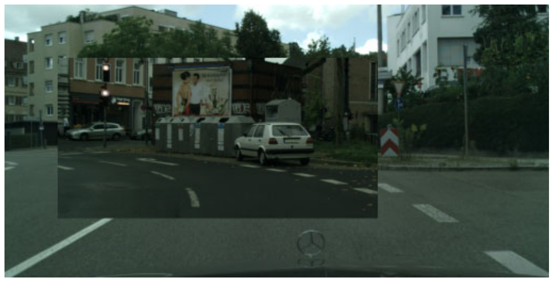
where x_1 and x_2 are two different images. The mixing algorithm can be used on both images and semantic maps. In the CutMix algorithm, rectangular regions are cut out of one image and pasted onto another, thus forming a strongly augmented image. An example is shown in figure 2.6.

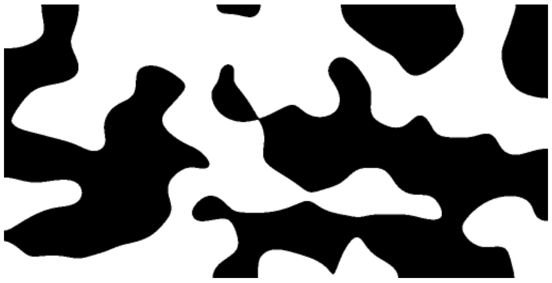
A downside to the CutMix approach is that regions are always rectangular and perpendicular to the image, restricting the variability of the perturbations that can be produced. French et al. [36] propose a mixing algorithm similar to CutMix, but generated using Gaussian noise, which makes the resulting mask more complex and with more degrees of freedom, a method they call CowMix because of the look of

(a) Image 1.

(b) Image 2.


(c) CutMask.


(d) Mixed image, using CutMix.


(e) CowMask.

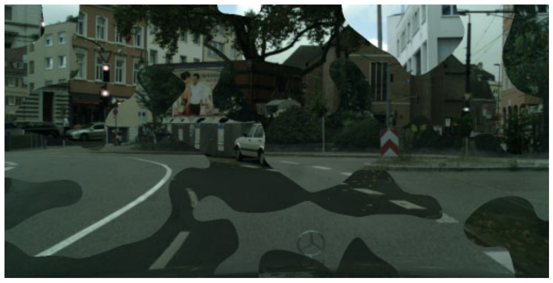

(f) Mixed image, using CowMix.

Figure 2.6: Examples of the CutMix and CowMix mixed sample data augmentations. The two images are mixed using the binary masks, CutMask in **(c)** and CowMask in **(e)**.

the patterns it forms. A similar mask was proposed by Harris et al. [55], formed using Fourier transforms instead of Gaussian noise.

In CowMix, a mask, CowMask, is created by generating a Gaussian field N the size of the input image and convolving it with a Gaussian smoothing kernel with strength σ, and lastly thresholding the values at a prespecified value such that a boolean mask is created. See [36] for a more detailed description. The scale of features in this mask can vary a lot depending on the choice of σ, as shown in figure 2.7. Figure 2.6 shows an example of CowMix.

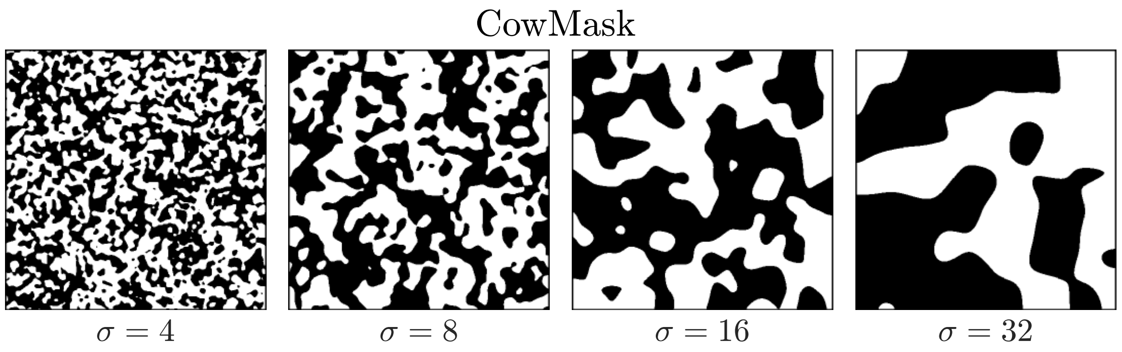


Figure 2.7: Examples of CowMasks generated with different σ, on images with size 512×512. Details are more fine grained the smaller σ is.

2.5.3 Pseudo-labeling

As we use augmentation anchoring, section 2.5.1, where we train the model to make the output from a strongly augmented sample to be similar to the output from a weakly augmented sample, we have the opportunity to use pseudo-labeling (also known as self-training). Pseudo-labeling is the process of sharpening the output distribution of a model maximally. Given an output distribution vector $p(x)$ from for example a neural network, which will typically contain values corresponding to the predicted probabilities of each option being the correct one, all values are set to zero except the maximum value which is set to one, forming a one-hot-encoding $\boldsymbol{e}_n$,

$$\widehat{p(x)} = \boldsymbol{e}_n, \qquad n = \operatorname{argmax} p(x). \tag{2.11}$$

n can take integer values between one and the number of possible classes. Figure 2.8 shows an illustration of pseudo-labeling. Using pseudo labeling has proven useful for semi-supervised learning tasks [18, 13, 56] as well as for domain adaptation [50, 51, 52]. The use of pseudo-labels differ between research; sometimes models are iteratively trained with more and more pseudo-labels included from evaluating on all unlabeled images selectively chosen for retraining [50, 51], while others generate them dynamically during the training process in each mini-batch [13].

The entropy of an output distribution is a measure of its uncertainty, where a more uncertain output has higher entropy, like the left one in figure 2.8, and an output with high certainty has low entropy, like the right one in figure 2.8. The use of pseudo-labels implies enforcing entropy minimization which has been shown to improve the generalization ability by creating low-density regions in the encoded state

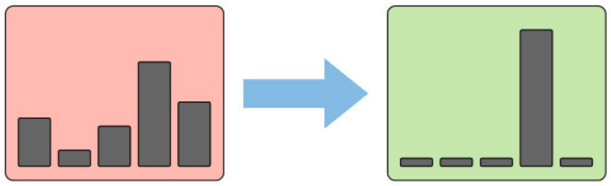


Figure 2.8: A schematic of how pseudo-labeling works. Given the output from a model in the form of a probability distribution, left, all values are set to zero except the largest value which is set to one, forming a one-hot-encoding, right. The entropy is higher on the left than on the right.

of the network [18]. The presence of low-density regions within the encoder's output was shown by Ouali et al. and exploited for semi-supervised semantic segmentation [35]. The usefulness of pseudo-labels for semi-supervised learning has been shown previously for classification [13]. This motivates the use of pseudo-labels for semi-supervised segmentation.

With infinite capacity, the model would be able to simply assign all unannotated samples with zero-entropy labels to minimize its objective function, though in practice this is rarely an issue for semi-supervised learning as the network is also guided by the supervisory signal from the labeled data. For unsupervised domain adaptation, however, the network might be able to discriminate between the two domains based on the different statistics of the datasets, and simply correlate the target domain with minimal entropy pseudo-labels. This is a special case of confirmation bias for self-training [37]. For this reason, when using mixed sample data augmentations in this setting it may be beneficial to combine images between domains to ensure that all classes are represented in the targets of the network. This means that the network can not learn to discriminate on global statistics of the images alone, and would have to learn to fit around the pasted objects, which is a considerably more difficult task, thus working against the minimal entropy collapse just described.

2.5.4 Loss weighting

When the total loss function for training a model is comprised of more than one component it is important to weigh these components appropriately relative to each other with weight factors λ, so that the training signal is balanced. This is the case for semi-supervised learning where there is one supervised and one unsupervised component, and for unsupervised domain adaptation where there can be one component from the labeled source domain data and one component from the unlabeled target domain data. The total loss then becomes

$$L = L_l + \lambda_u L_u, \tag{2.12}$$

where L_l and L_u are the losses from the labeled and unlabeled samples respectively and λ_u is the weight factor. When using consistency regularization one often wants the loss from the unsupervised components to be small at the beginning of training, as to not train too much on incorrect predictions, and have it gradually grow as the network improves during the training process [12, 15]. The most straight forward way of doing this is by having the unsupervised weight being a function of the training iteration, $\lambda_u(k)$, for example in the form of a sigmoidal ramp-up

$$\lambda_u(k) = \lambda_{u0} \cdot \exp(-c(k_{max} - k)^2), \tag{2.13}$$

used in [5, 4]. Here, λ_{u0} and c determine scale and slope of the ramp-up respectively, and k_{max} is the iteration at which the ramp-up should reach its maximum, after which the weight is kept constant.

More sophisticated methods can use the predictions of the network to determine the weight λ_u. When using pseudo-labeling, a threshold is often set, such that only samples that have output predictions with a probability assigned to any class above this threshold are used for training. This reduces the risk that the model is trained to give the wrong predictions, as the samples on which it yields a high output prediction are more likely to be correct. The weights become

$$\lambda_u = \lambda_{u0} \begin{cases} 1, & \text{if } \max(p(x)) > \tau \\ 0, & \text{otherwise} \end{cases}, \tag{2.14}$$

where $p(x)$ is the output distribution for sample x, and τ is a threshold value. This method was employed by [6] for image classification, with $\tau = 0.968$. This threshold has to be tuned. If it is set too low, samples that the model is insecure on will be included, but if the threshold is too high a lot of information will be lost, as many samples will not be used for training.

This issue is straightforward in image classification, where there is only one output distribution per sample. In semantic segmentation, however, we have an output distribution, and hence a separate weight $\lambda_u(k)$, for each pixel in the image. This makes the problem more complex, as some types of pixels will always be more certain than others. In particular, pixels close to the boundary between two different classes are especially hard for the model to predict, and the certainty of these predictions will hence be lower. In practice, we have seen that this yields pseudo-labels that miss most boundaries, meaning that no unsupervised training will ever be performed on these parts, see figure 5.5d. This means a large loss of important training data. Figure 2.9 shows the proportion of pixels that have a certainty above a threshold of 0.968, as used in [6], as a function of training iteration. From this we can see that almost 15% of pixels are below this threshold, meaning that not training on these pixels would mean a loss of 15% of the training data.

Another way to weigh the loss without having the problem that pixels near class boundaries are completely masked is to weigh the loss uniformly for all pixels by the proportion of pixels for which the certainty of the network is above a given threshold. This method was used by French et al. [15]. The unsupervised weight factor will then be

$$\lambda_u = \frac{\sum_i \lambda_u^{(i)}}{\sum_i 1} = \lambda_{u0} \frac{\sum_i \mathbb{1}(\max(p(x)) > \tau)}{\sum_i 1}, \tag{2.15}$$

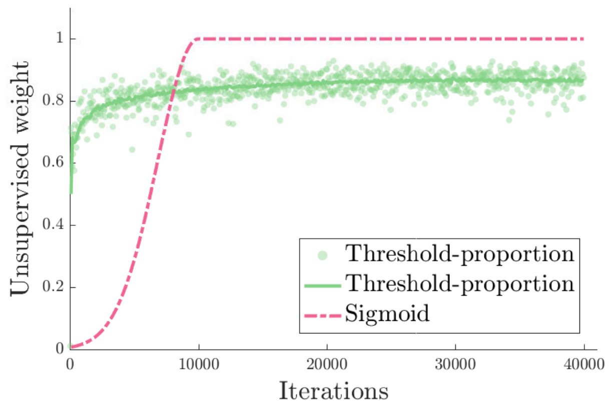


Figure 2.9: The proportion of pixels in images where the model's predicted confidence is above the threshold $\tau = 0.968$, as a function of the training iteration. The dots mark the proportion for individual batches, while the values of the solid line are averages of the proportions for 100 batches. It is compared to the sigmoid ramp up from equation 2.13, with $c = 5$ as in [4, 5], and $k_{max} = 10000$. The confidence rises as training progresses, and seems to settle at a value slightly above 85%. Each value in the curve is an average of 100 iterations. These particular values were generated during training of a semi-supervised model with 372 labeled samples from the Cityscapes dataset.

where $\lambda_u^{(i)}$ is the weight factor in equation 2.14 for pixel i. The weight factor in equation 2.15 is exactly the one shown in figure 2.9. Its value will grow during training, giving a ramp up of the unsupervised loss similar to that of a sigmoidal function. This is the method used in our results, however, the other weighting methods described are also examined in the ablation study in section 5.2.

2.5.5 Mean Teacher

In school, humans learn by being instructed by a teacher, who is assumed to be more knowledgeable than the student. Inspired by this, one can imagine an approach to machine learning where a teacher model is used as a guide for a student model. Especially for unsupervised consistency regularization, this can be a very useful method, as the teacher model can be used to make a more stable prediction, and the student network can then be trained to be consistent with this.

Model ensembling is a method often used when trying to achieve high-performance results using machine learning models. The idea is that one trains several models to perform the same task, and in the inference stage, the outputs of all these models are averaged, which will give a more stable and reliable result. Laine and Aila introduced self-ensembling [5], where the ensemble is formed by averaging over several outputs from consecutive training epochs for each sample, under different perturbations of the data. This method can be used to create a teacher model, which has proven successful when used for semi-supervised image classification tasks.

An issue with the method developed by Laine and Aila is that predictions have to be saved for all samples in the data set. This is not a problem for small data sets, but as the amount of data grows it becomes infeasible. With this motivation, the approach was further developed by Tarvainen and Valpola, with the introduction of the Mean Teacher framework [4]. Here, the teacher model is not achieved by averaging over predictions of each data point, but instead by averaging over model parameters for many training iterations of the network. The average is an exponential moving average (EMA), meaning that the parameters of the teacher model for each training step are given by

$$\theta'_k = \alpha\theta'_{k-1} + (1-\alpha)\theta_k. \tag{2.16}$$

Here, θ'_k is the model parameters for the teacher network in iteration k, θ_k is the model parameters for the student network in iteration k and α is the smoothing coefficient of the EMA. Using an EMA results in a teacher model that is an average of the student model from all previous training steps, with the most weight given to the most recent versions of the student, without having to store more than one set of parameters. This averaging will make the teacher model a more stable version of the student model, experiencing smaller fluctuations, while still being improved continuously, as shown for image classification by Tarvainen and Valpola [4]. The same seems to be the case for semantic segmentation, as stated by French et al. [15].

2.5.6 Distribution alignment

An issue when deploying consistency based semi-supervised learning for classification tasks is that the model can be biased to predict some classes proportionally more than others in regards to the underlying distribution. This will often hurt the model's ability to predict classes with lower frequency, in favor of classes with high frequency. To remedy this, one can introduce Distribution alignment, which pushes the model to give higher predictions to certain classes and lower to others. The idea was first introduced by Bridle et al., who encouraged their classifier to equally predict all classes with what they called a fair objective [57]. This was accomplished by maximising mutual information between a model's input and output. An issue with this, however, is that the underlying distribution might not be uniform, in which case it is not desirable to enforce just that.

An alternative way of performing distribution alignment was introduced by Berthelot et al. in their ReMixMatch algorithm for semi-supervised image classification [31]. They calculate the marginal class distribution $p(y)$, i.e. the proportion of labels belonging to each class, of all the labeled training data. They then encourage the model to make the aggregate of its predictions on unlabeled samples follow this distribution. They do this by maintaining a running average of the model's predictions, which they refer to as $\tilde{p}(y)$. This can be seen as a representation of the current bias of the model. Outputs q from the model when fed a weakly augmented image are scaled using the ratio $p(y)/\tilde{p}(y)$, and then normalized such that they represent a proper probability distribution $\tilde{q}$. The idea is that this modified output $\tilde{q}$ is pushed to give higher predicted probabilities for classes that have recently not been predicted as much as they occur in the marginal distribution $p(y)$, and the opposite for classes given too high probabilities. Berthelot et al. compute the running average over the output from the latest 128 unlabeled samples. Using this method they see a substantial decrease in loss when training on the CIFAR-10 data set [58].

We have used Distribution alignment as a proposed method for Unsupervised domain adaptation. It is, however, not a part of the method used to obtain our results, but is examined in our ablation study in section 5.2.

3 Methods

In this chapter, we provide a detailed description of our proposed algorithm for semi-supervised learning and unsupervised domain adaptation. We also present the datasets that have been used, and which augmentations that have been used apart from the mixed sample data augmentations described in the previous chapter. Lastly, the employed network architecture is described.

3.1 ClassMix

We here introduce a novel way of mixing multi-class images, based on semantic maps, which we call ClassMix. It is similar to the mixed sample data augmentations CutMix and CowMix presented in section 2.5.2. ClassMix has been an integral part for us achieving state-of-the-art results.

ClassMix builds on the principle of augmentation anchoring, where the output from a weak augmentation of an input is used as a label for a strong augmentation of the same input. From the output of the weakly augmented image, we obtain a pseudo-label, such that every pixel is predicted to belong to a class. We then randomly select n classes to keep, where n is between zero and the number of classes present in the pseudo-label, we have used n such that half of the present classes are included. From this a mask is created, such that the mask is one for all pixels belonging to one of the n classes we randomly selected, and zero for all other pixels. This mask is referred to as ClassMask. The mask is then used in the same way as described in section 2.5.2, i.e. it is pixel-wise multiplied with one image and added to the inverse of the mask multiplied with another image. The algorithm to create a new, mixed, batch, is described in Algorithm 1. Apart from the creation of the masks, this algorithm is the same for CutMix and CowMix. Figure 3.1 shows an example of ClassMix.

The motivation for this mixing strategy is that mask borders will, to a high degree, follow the semantic boundaries between objects of different classes in the image. This will minimize the occurrence of only a very small part of an object being present in a mixed image, which can be nonsensical. It also reduces the number of unnatural borders in the resulting mixed image, as the borders will be aligned with the actual boundaries of objects. This creates mixed images that better respect the semantic boundaries of the original images. They are consequently more realistic looking than images created using CutMix and CowMix, and also lie close to the underlying data distribution, meaning they are more useful for training, compare figures 2.6 and 3.1. In a setting with rich semantics such as urban scene segmentation, we further note that the mask will be very varied, owing to the variety of possible classes being transferred.

Algorithm 1: ClassMix

```
Input : Batch of samples $\mathcal{X} = \{(x_b, p_b) : b \in (1, ..., B)\}$, where $x_b$ are images
        and $p_b$ the corresponding labels or predictions made by the model,
        $B > 1$, proportion $p$ of classes being selected for the mask.
for $b = 1$ to $B$ do
    $c_{b,tot}$ = set of classes present in the semantic map $p_b$
    $c_b$ = choose $\lceil p \cdot |c_{b,tot}| \rceil$ values from $c_{b,tot}$
    $M_b = (p_b == c_b)$ // Create boolean mask
    $x'_b = M_b \odot x_b + (1 - M_b) \odot x_{b+1}$ // New sample is mix of two samples
    $p'_b = M_b \odot p_b + (1 - M_b) \odot p_{b+1}$ // New label is mix of two labels
end
$\mathcal{X}' = \{(x'_b, p'_b) : b \in (1, ..., B)\}$
return $\mathcal{X}'$
```

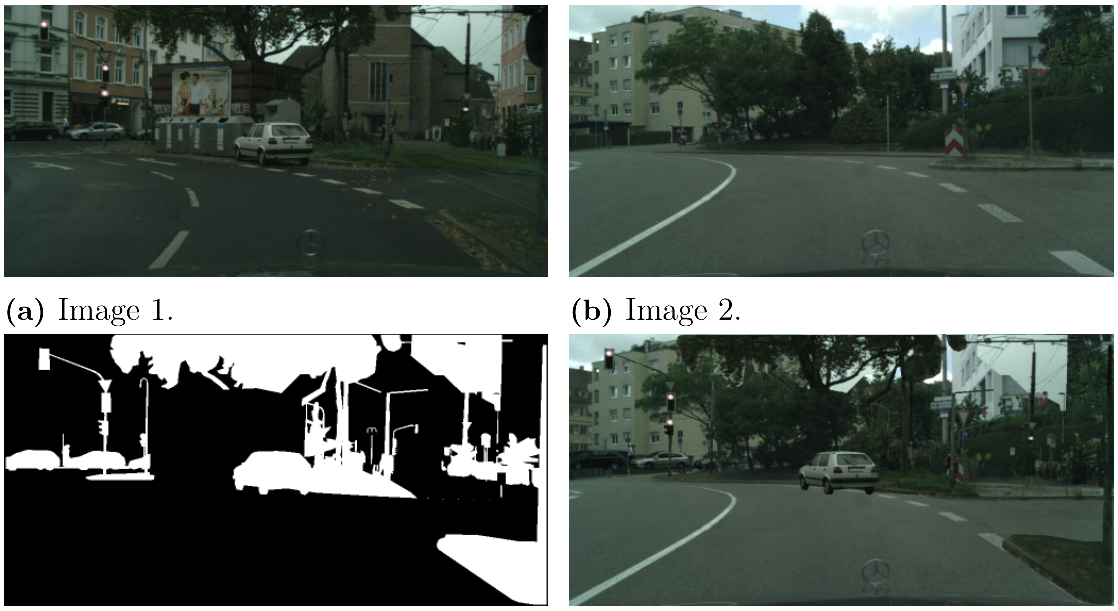

(a) Image 1. **(b)** Image 2. **(c)** ClassMask based on image 1. **(d)** Mixed image.

Figure 3.1: An example of how ClassMix works. Based on an image, **(a)**, a mask is created such that half of the classes occurring in the image are present, **(c)**. These classes are then pasted on top of a second image, **(b)**, forming a strongly augmented image, **(d)**.

When using ClassMix it is also especially beneficial to use pseudo-labeling, introduced in section 2.5.3. This is because the boundaries between classes are usually the areas where the network is the most uncertain. The outputs close to the class boundaries will, in general, mostly be an interpolation between the two classes on the two sides of the boundary. The uncertainty between these classes is warranted and reasonable. However, when mixing samples it is probable that the resulting images will not have these classes next to each other, but rather that they will border completely different classes, see figure 3.2. This means that the uncertainty will now be between two classes where one of the classes is not present in that area of the image, which is unreasonable. If one were to not use pseudo-labels this would mean that the network would be trained towards this unreasonable uncertainty. Therefore we believe that pseudo-labeling is especially useful when using mixed sample data augmentations in general, and even more so when using ClassMix in particular, as mixing boundaries will always lie along class boundaries where this issue is the most eminent.

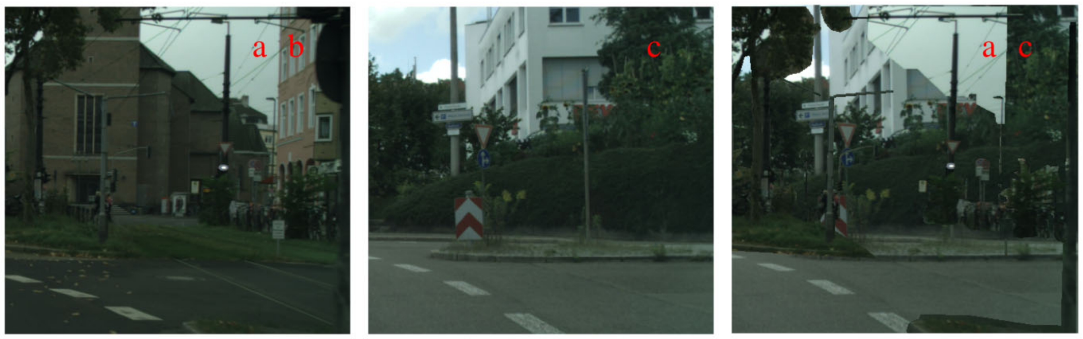


Figure 3.2: An illustration of how the context in an image changes when using ClassMix. The left and middle images are two original images and the right one is mixed using ClassMix. Consider classes sky (**a**), building (**b**) and tree (**c**). The output distributions of pixels close to the boundary between classes **a** and **b** when running the left image through a model will have some weight on both classes. After mixing and when using the corresponding mixed predictions as a target these two classes are no longer adjacent. Instead classes **a** and **c** are next to each other. The uncertainty in the target for class **a** is however still between classes **a** and **b**. It does not make sense to train a model against this uncertainty as class **b** is no longer there. Using pseudo-labels will alleviate this problem since the model is then only trained against the single, highest predicted, class.

Though formulated here as a generalisation of CutMix using a binary mask to mix two images, this augmentation also shares similarities to strategies where objects are pasted onto background scenes [59, 60, 61]. Our way of combining two images conditioned on the predicted or ground-truth semantic mask exploits the same idea of compositing images. However, it is different from these existing strategies in that we mix images based on predictions or across domains. And rather than selecting individual instances of objects and only training on these, we transfer entire classes and also train on the background image. Segmenting the background image as well means that objects recognized by the network do not only have to be invariant to their context, but invariant to a diverse set of occlusions as well. In summary, our goal is not to use ClassMix to synthesise new data for supervised-learning similar to several previous works, but rather to exploit it to enforce consistency, or in the case of domain adaptation; force entropy into the pseudo-labels.

3.2 Training algorithm

Our training algorithm is based on the concept of augmentation anchoring, where the model is trained to make the same prediction on a strong augmentation as on a weak augmentation of the same sample. One batch of samples contains $2B$ samples, where B is referred to as the batch size. B samples are labeled and B samples are unlabeled. The labeled samples are treated as in standard supervised training, i.e. run through the student network and a softmax function, and after that compared to the corresponding ground truth label, using a loss function L_l. In our solution, this loss function is the cross-entropy loss, as described in section 2.1.1. We are not using any augmentations for the labeled samples.

After the processing of the supervised component, the B unlabeled samples are handled. The handling of these are here explained in text and summarised in a schematic in Figure 3.3. Each image x is treated in two separate ways x_w and x_s, one weak and one strong augmentation scheme. The images x_w are then passed through the teacher network and after that a softmax function, creating output predictions $p(x_w)$. These predictions are used to determine the loss weighting factors λ_u for each sample, where we use the threshold-proportional weight in section 2.5.4. After this, the outputs $p(x_w)$ are transformed into pseudo-labels $\widehat{p(x_w)}$ so that we obtain a proper semantic map where each pixel is assigned one class. These pseudo-labels are used to create the ClassMasks. We randomly select half of the classes present in the pseudo-label to create a binary mask M, details are explained in algorithm 1. One mask is created for each unlabeled image in the batch, and after that used to mix the images x, giving the strongly augmented samples x_s. The same masks are applied on the corresponding pseudo-labels, to give the correct training target, resulting in the mixed pseudo-labels $\widehat{p(x_w)}_s$. The augmented samples can be modified further, however, in our default solution no further augmentations are used. The mixed samples x_s are then passed through the student network and a softmax function, forming the predictions $p(x_s)$. The unsupervised part of the loss is calculated using a cross-entropy loss L_u between the predictions $p(x_s)$ and the mixed pseudo-labels $\widehat{p(x_w)}_s$.

The training objective for the batch is then the combination of the supervised loss L_l and the unsupervised loss L_u

$$L = L_l + \lambda_u L_u, \tag{3.1}$$

which is minimized by the stochastic gradient descent algorithm, section 2.1.2. This whole process is then repeated for a new batch of samples for a set number of iterations.

For unsupervised domain adaptation, the algorithm changes slightly, where instead of mixing two unlabeled images we mix one unlabeled image from the target domain and one labeled image from the source domain, see figure 3.4. This is done to avoid entropy collapse, see section 2.5.3, where larger classes would otherwise dominate the transfer, merging with smaller classes. This approach is motivated by the fact that the network can no longer learn to correlate structure specific to the target domain with minimal entropy, as objects of all classes will be present. While

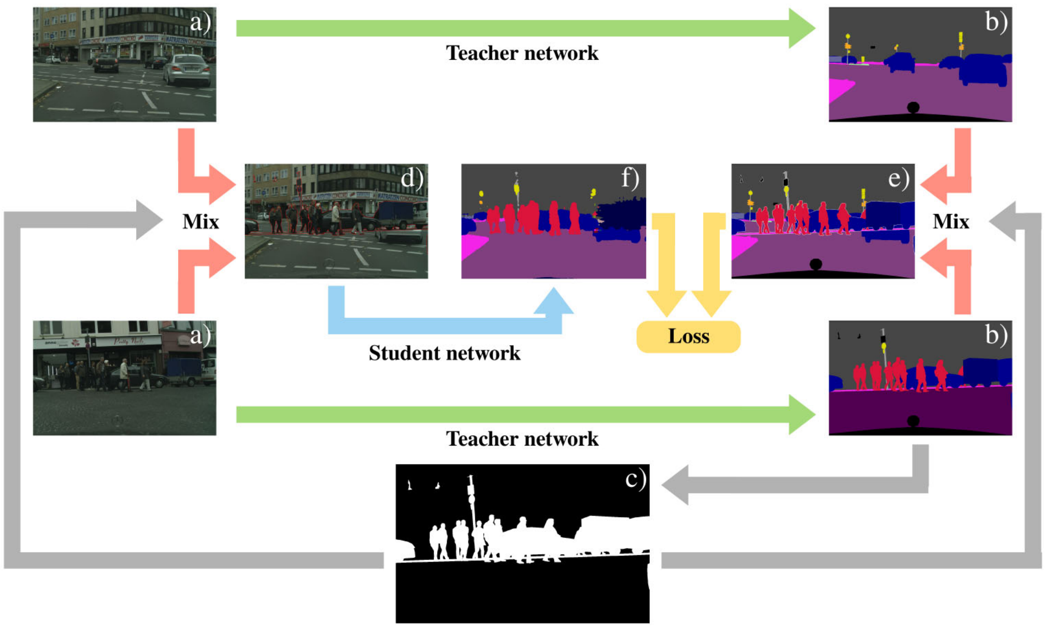


Figure 3.3: A schematic showing the unsupervised part of our SSL algorithm. Two images, **a)**, are passed through the teacher network, yielding two predicted semantic maps, **b)**. From these, two masks, **c)**, are created (however in this image only one mask and mix is shown). These masks are used to create two mixed images, **d)**, and two corresponding mixed semantic maps **e)**. The mixed images are then passed through the student network, giving predictions **f)**, after which the loss is computed against the semantic maps.

it is still possible for the network to learn to fit to the pasted objects from the source domain and output everything else with minimal entropy, this would be considerably more difficult, and hence it is more likely that the network instead learns to provide a correct segmentation. The mask is always created conditioned on the ground truth of the source domain image (which is provided in abundance in contrast to the data-sparse semi-supervised learning setting), rather than the pseudo-label of the target image. This means that the objects from the source domain will always have perfect boundaries.

Note that our method does not require retraining by iterative extracting of pseudo-labels, unlike other common self-training methods [50, 51, 62]. We also handle uncertainty by a simple weight for our unsupervised loss, which does not require the network to be converted into a Bayesian one with multiple forward passes for each pseudo-label generation [52].

3.3 Data

We have considered four different datasets in this thesis, all of which are common benchmarks for semantic segmentation. The Cityscapes and Pascal VOC 2012 datasets are both common datasets used for semantic segmentation, both supervised

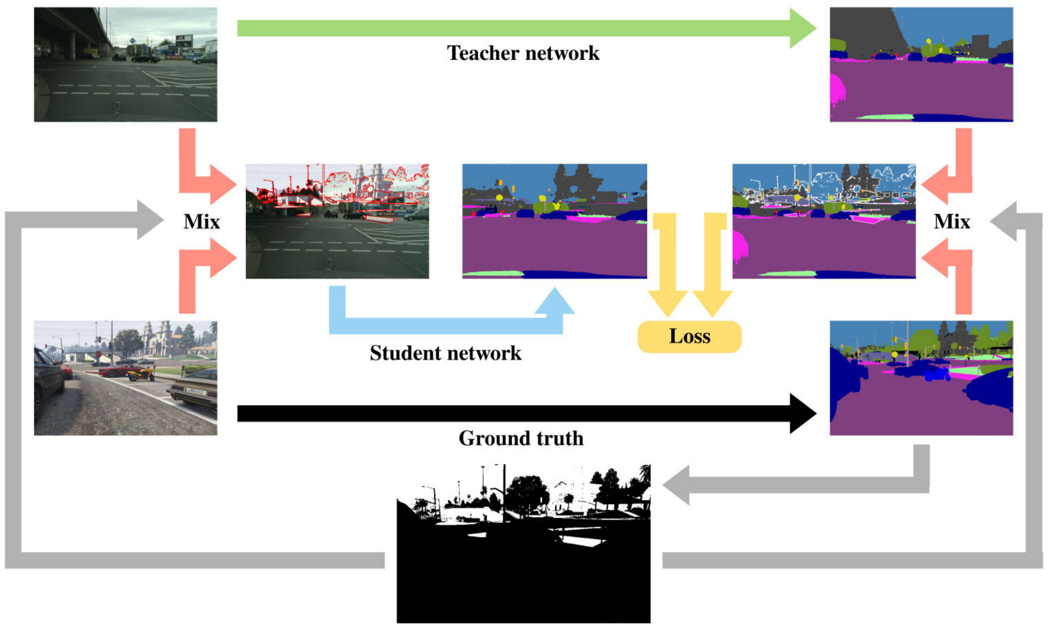


Figure 3.4: A schematic showing our unsupervised domain adaptation algorithm. A source image is passed through the teacher network to create a pseudo-label. A ClassMask is created based on the source image ground truth semantic map. Using this mask the source- and target images are mixed, as well as the source ground truth label and the target image pseudo-label. The mixed image is then passed through the student network after which the loss is computed against the mixed label.

and semi-supervised. The GTA5 and SYNTHIA datasets are collections of synthetic images commonly used for unsupervised domain adaptation, with Cityscapes being the target domain.

3.3.1 Cityscapes

The public Cityscapes data set is a freely available data set that consists of images taken of the front view of cars in urban environments, with pixel-level annotations [2]. The set consists of 5000 finely annotated frames, which are divided into training, validation and test sets with 2975, 500, 1525 frames respectively, as well as an additional approximately 20 000 coarsely annotated frames. The fine annotation of the images take on average more than 1.5 hours per image to produce, demonstrating the need for data-efficient solutions for semantic segmentation. The test set annotations are not publicly available, and that part of the data is therefore not used, the coarsely annotated frames are also not used because of the less detailed labels. Images are taken from 50 videos in 50 different cities, mostly in Germany. The images contain 19 different classes, such as road, cars and pedestrians, all classes are shown along with their relative frequency of occurrence in figure 3.9. The image resolution is 1024×2048. The Cityscapes dataset has been our primary dataset and the one that has been used for evaluation of methods as well as ablation studies. Figure 3.5

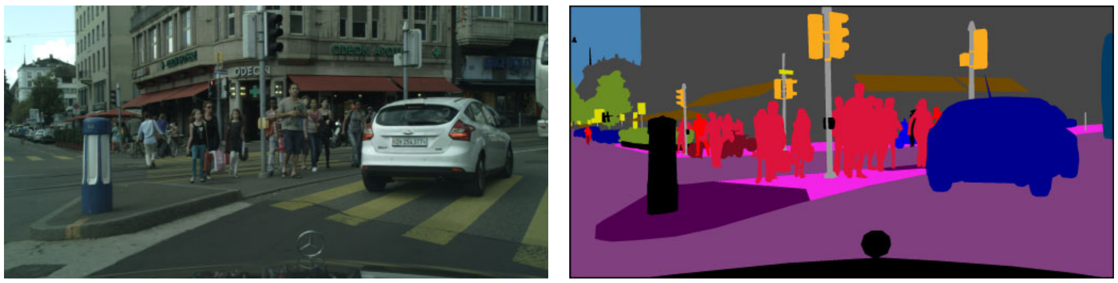


Figure 3.5: An example of a frame from the Cityscapes data set, to the left is the image and to the right is the corresponding semantic map.

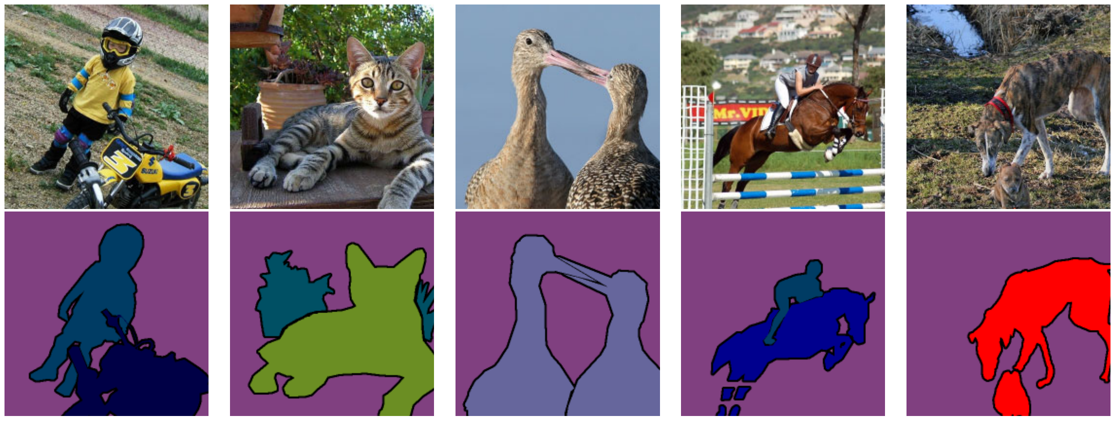


Figure 3.6: Example of images and corresponding semantic maps from the Pascal VOC 2012 dataset. Each image consists of one or more objects from 20 classes in front of a 21st background class. Images are cropped to be square.

shows an example image along with its corresponding ground truth semantic map.

3.3.2 Pascal VOC 2012

The Pascal VOC 2012 dataset is another publicly available semantic segmentation dataset commonly used for benchmarking [63], going forward we will refer to it as Pascal VOC. It contains images of everyday objects such as bicycles, birds, people and tables, totalling 21 classes, including a background class. There are originally 2914 labeled images, but including the labels from the Semantic Boundaries dataset [64], the total number of labeled images becomes 12 032, divided into 10 582 training images and 1 449 test images. The size of the images varies, but they are a maximum of 500 pixels in both x and y directions. Figure 3.6 shows five example images along with their corresponding ground truth semantic maps.

3.3.3 GTA5

The GTA5 dataset is a synthetic dataset comprised of images from the open world computer game GTA5 together with ground truth labels for semantic segmentation compatible with Cityscapes [65]. All images are taken as first person views from a car, in the same way as Cityscapes. The set consists of 24966 densely labeled frames.

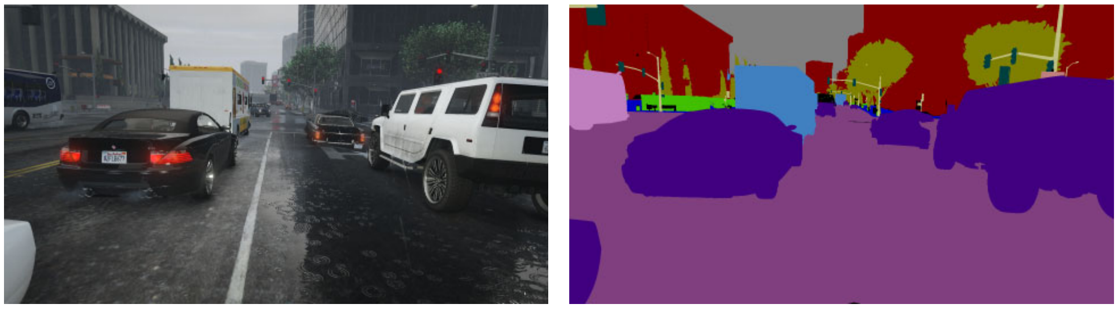


Figure 3.7: An example of a frame from the GTA5 data set, to the left is the image and to the right is the corresponding semantic map.

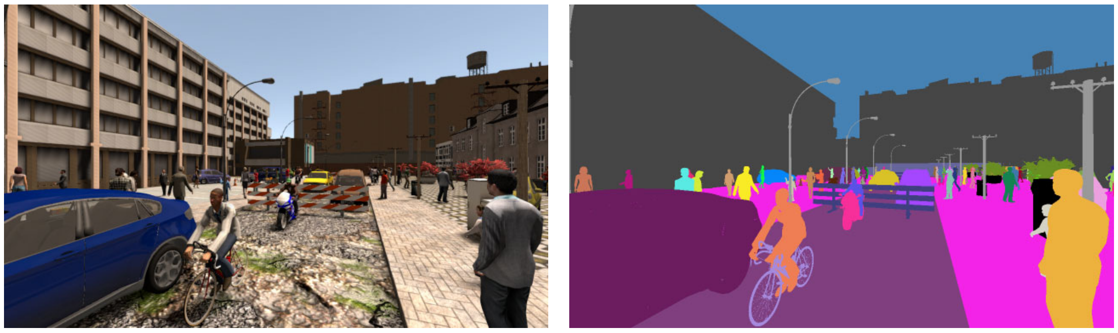


Figure 3.8: An example of a frame from the SYNTHIA data set, to the left is the image and to the right is the corresponding semantic map. In this particular visualisation the map is instance-aware, meaning the semantic map has a separate label for each person etc.

The image resolution is 1052×1914. Images in the GTA5 dataset were labeled in, on average, 7 seconds per image, a stark contrast to the 1.5 hours per image for the Cityscapes images, underlining the point of using domain adaptation from synthetic data. Figure 3.7 shows an example image along with its corresponding ground truth semantic map.

3.3.4 SYNTHIA

The SYNTHIA dataset is a collection of synthetic images for semantic segmentation of urban scenes [66], a subset of which contains 9000 images with labels consistent with Cityscapes. Three of the classes in Cityscapes are however not represented. The image resolution is 760×1280. Images are taken from different views, including close-to-ground views as in figure 3.8, as well as bird's eye views.

3.4 Augmentations used

Computer vision tasks heavily depend on the use of suitable augmentations, i.e. transforming the data for a regularizing effect. Apart from ClassMix described in section 3.1, we have also used several other, more common, augmentations.

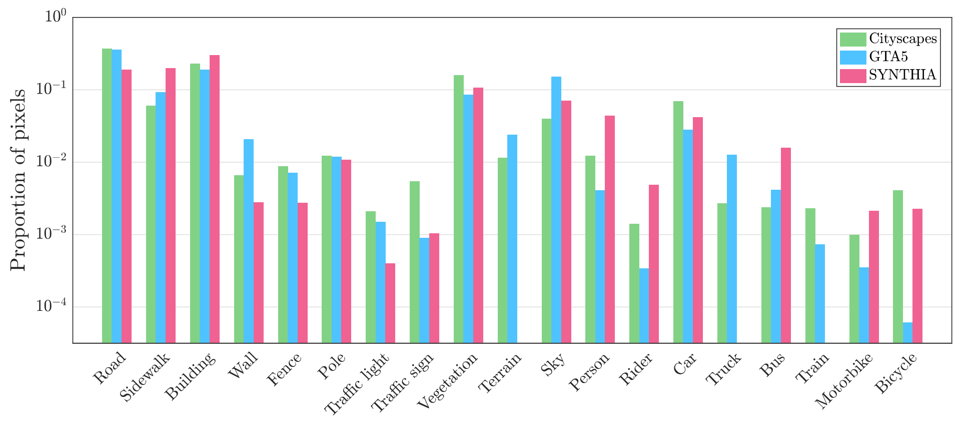


Figure 3.9: The proportion of all pixels belonging to all different classes in the Cityscapes, GTA5 and SYNTHIA datasets. Note that the vertical axis is logarithmically scaled.

Crop	Instead of working with the full images, random cropping is performed each time an image is loaded. A random rectangle with given side lengths is cropped out of the image, discarding the rest. If the crop is larger than the image, the image is padded with zeros for the remaining pixels, and the label is ignored for the corresponding area, meaning no supervised training is performed on these pixels. Cropping has the benefit of, besides the regularizing effect, making the images smaller and hence speeding up propagation.
Horizontal flip	Images are mirrored with respect to the vertical center-line of the image.
Gaussian blur	Images are convolved with a Gaussian kernel, producing the effect of a more blurry image. The power of the blurring is decided by a parameter σ drawn from a uniform distribution $\mathcal{U}(0.15, 1.15)$. The size of the blurring kernel is set to one tenth of the image size in both the vertical and horizontal directions.
Color jitter	Adjusting the colors of images. We weakly adjust brightness, contrast, hue and saturation of the images.

3.5 Network Architecture

The network architecture is arguably the most important factor in achieving good results in any deep learning task, including semantic segmentation. We have used the DeepLabv2 framework developed by Chen et al. 2017 [23], as it has previously been used in several papers about both semi-supervised learning and unsupervised domain adaptation for semantic segmentation, making for easier comparison with those works.

DeepLabv2 is a somewhat outdated framework, and there are several newer architectures which perform better. However, these newer models in general require more computational power, not allowing us to try as many different methods. Secondly, our focus is on the training algorithm, rather than on achieving the best possible results for the specific task and dataset. Therefore we feel that it is not a downside to not have used the best possible network.

Table 3.1: The architecture of our DeepLabv2 implementation. The architecture can be divided into 6 blocks, the first block is a standard 7×7 convolution followed by a max pooling layer, after this block the input has changed from 3 RGB channels to 64 channels, with a downsampling of 1/4. Layers 2 through 5 consist of a number of residual units, as described in section 2.1.3, each unit consisting of three convolutional layers, each followed by a batch normalization layer and a ReLU function. The number of residual units in each block is specified in the table. Blocks 4 and 5 are comprised of atrous convolutions with rates 2 and 4 respectively. The last block is the Atrous Spatial Pyramid Pooling, which consists of four parallel atrous convolutions with rates 6, 12, 18 and 24 respectively.

Block	Convolutions	Output size
1	7×7, 64 3×3 Max Pool	64, 1/4
2	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	256, 1/4
3	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	512, 1/8
4	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	1024, 1/8
5	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	2048, 1/8
6	ASPP Module	*num_classes*, 1/8

Our model has a ResNet-101 backbone (meaning a ResNet with 101 layers)[1], adjusted to the DeepLabv2 framework, such that it downsamples the images eight times. The outputs from the network are bilinearly upsampled to be the same size as the inputs. The architecture can be divided into six separate blocks, each with several layers, see table 3.1. Blocks 4 and 5 use atrous convolutions instead of standard convolutions, as described in section 2.2.1, with rate parameters r set to 2 and 4 respectively. The last block is the atrous spatial pyramid pooling, described in section 2.2.1, it consists of four parallel atrous convolutions with rates 6, 12, 18 and 24 respectively.

4

Results

In this chapter, we present the results for our proposed method for consistency regularization on both semi-supervised learning and unsupervised domain adaptation and compare these against the current state of the art for public benchmarks. Before that, we explain the practical details of the experiments and present the hyperparameter values used during training, as well as describe the evaluation metric used.

4.1 Implementation details

4.1.1 Practical details

All code for this thesis is written in Python, as it is the go-to language for machine learning development. As for the deep learning framework, PyTorch is used as it offers a lot of flexibility and at the same time high performance. The code for the semi-supervised part of the thesis can be found at `https://github.com/WilhelmT/ClassMix`, and the code for unsupervised domain adaptation can be found at `https://github.com/vikolss/DACS`.

We have had access to a cluster of four NVIDIA GeForce GTX 1080 Ti GPUs, each with 11 GB of memory, on which training and testing have been performed. We have also used cloud computing instances from Amazon Web Services, in the form of NVIDIA V100 Tensor Core GPUs, each with either 16 GB or 32 GB of memory. On top of this, we have also had two smaller NVIDIA Quadro M2000M GPUs with 4 GB memory, used primarily for small scale experiments.

4.1.2 Experimental setup for semi-supervised learning

In this section, we list what hyperparameter values we use for training models using semi-supervised learning, for those that have been the same for all experiments. The search space is way too large for us to optimize every parameter, so a lot of choices have been made solely based on what previous researchers have used. In particular, we have a similar setup as Hung et al. for semi-supervised learning [24].

As optimizer, we use standard Stochastic Gradient Descent, with a learning rate of 2.5×10^{-4}. We use a momentum of 0.9 and a weight decay of 5×10^{-4}. We use the polynomial learning rate schedule presented in equation 2.6, with power 0.9, and k_{max} set to the number of iterations the model is trained for. The loss function for the supervised part of training is Cross-entropy loss. The weight factor for the unsupervised loss, λ_{u0} is set to one, as in [13]. Along with a few parameters that change for different datasets, such as image size, augmentations, and normalization, this forms our baseline hyperparameters.

For the Cityscapes dataset, the results are obtained from runs with images rescaled from the original 1024×2048 to half that size, 512×1024. It is done this way because having full-sized images requires too much memory on the GPUs. Images of this size are also used in previous research [24], making for easier comparison between methods. We are using a batch size of two, for both labeled and unlabeled images, and train for 40 000 batch iterations. Ground truth labels are only publicly available for the training set and the validation set in the Cityscapes data, therefore we use the validation set as a test set. This is common practice for the Cityscapes dataset and in particular for previous work on semi-supervised learning [24, 25, 15, 62] and unsupervised domain adaptation [67, 46, 68, 69, 48, 70, 71, 50, 51, 72, 73, 74, 52]. Because of this, it is important to not optimize hyperparameters or to deploy early stopping based on the performance on the validation set since a high result on the validation set not necessarily means that the model generalizes well to all data, but could just mean that the model is performing well on those exact images. Hence no early stopping is used and all results are from after the last training iteration.

For the Pascal VOC dataset, images have different sizes and hence have to be cropped to become the same size, described in section 3.4. We crop the images to 321×321 pixels and use a batch size of 10 in line with previous work [24, 25, 15]. We train for 40k iterations here as well. In the same way as for the Cityscapes dataset, common practice is to divide the data into only a training set and a test set, therefore all results are from the last training iteration.

As for augmentations other than mixing, we do not use any for the Cityscapes dataset. For the Pascal VOC dataset, we scale all images, both for weak and strong augmentations, between 0.5 and 1.5 times. We also flip these images horizontally with a 50% chance, as described in section 3.4. The reason we are using augmentations for Pascal VOC and not Cityscapes is that that is the way previous works have done it, making for better comparisons.

4.1.3 Experimental setup for unsupervised domain adaptation

The settings for unsupervised domain adaptation training are almost identical to those for semi-supervised learning, in particular for the Cityscapes dataset. The size of images differs in that SYNTHIA images are kept at their original 760×1280 size, and GTA5 images are resized from their original size to 760×1280 pixels. To save memory (and since ClassMix requires identical image sizes between images to be mixed), we crop all images randomly with 512×512 crops, in the way described in section 3.4. The remaining differences to the hyperparameters for semi-supervised learning for Cityscapes are in the number of training iterations; going from 40k to 250k, and in the normalization of data; now being done with respect to the mean of the pretraining dataset (ImageNet), as we want to keep it constant over domains, rather than the Cityscapes dataset. In contrast to Tsai et al. we refrain from early stopping [67], for the reason stated above.

For domain adaptation, we apply Color jitter and Gaussian blur on mixed images after applying ClassMix, as explained in section 3.4.

4.2 Evaluation Metric - Mean Intersection over Union

To measure the performance of the semantic segmentation of an image the most common metric to use is the mean Intersection over Union (mIoU). It is the between-class mean of the quotients of the number of intersecting pixels between the prediction and the ground truth, and the number of pixels either predicted by the model or present in the ground truth, or both. Synonymously it is the number of true positives (TP) divided by the sum of true positives, false positives (FP) and false negatives (FN) for each class. The IoU, for each class, is given by

$$\mathrm{IoU} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}. \tag{4.1}$$

This yields values between zero and one, where one is perfect. Figure 4.1 shows an illustration of how the mIoU is calculated. In the figure, only one class is demonstrated. In reality, one would calculate the IoU for every class present in the image separately and after that take the mean over all classes. Using mIoU instead of the more simple pixel accuracy measure, where one only calculates the proportion of correctly predicted pixels, has the effect that missing a class entirely is penalised heavily. For example, if all ground truth pixels except one belong to the same class and the prediction is that all pixels belong to the majority class, pixel accuracy would yield a score very close to one. In the same example, mIoU would instead yield a score very close to 0.5. This means that pixels from small classes, such as signs or traffic lights in the Cityscapes dataset, become more important to correctly classify than pixels from larger classes, such as road. This intuitively makes sense as missing a few pixels of road will not change the overall shape of the road, but missing a few pixels of a much smaller sign might seriously alter its size and shape.

4.3 Results for semi-supervised learning

4.3.1 Cityscapes dataset

Table 4.1 shows the results from using our method to train models on the Cityscapes dataset. In the same table, the results from four previous papers regarding semi-supervised semantic segmentation are also shown, all using the same DeepLabv2 network with ResNet101 backbone. Hung et al. have the same hyperparameters as us [24]. Mittal et al. crop their images to 256×512 before processing [25], and French et al. also crop their images as well as using a different optimizer [15], both papers also used a larger batch-size of five and four respectively. Unlike Mittal et al. and French et al. we use weights pretrained on MSCOCO similar to Hung et al. Hung et al. have not made their code for Cityscapes publicly available, therefore we process our data in the same way as Mittal et al. who do have a public implementation.

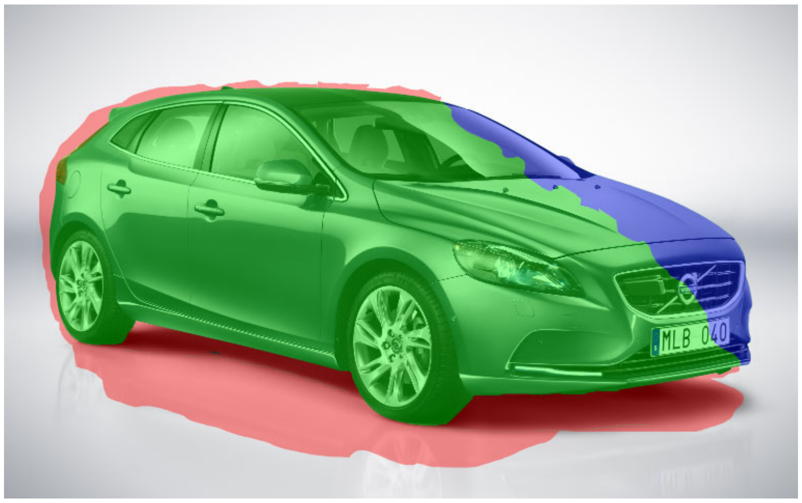


Figure 4.1: An illustration of how the mean intersection over union metric works. This example shows only the car class. Green pixels are pixels that both the ground truth label and the model's prediction agree belong to the car, true positives. Blue pixels are pixels that according to the ground truth belong to the car class, but the model did not predict them as such, false negatives. Red pixels are predicted to be car by the model but are not in fact car according to the ground truth. The intersection over union for the car class will be the number of green pixels divided by the sum of numbers of green, blue and red pixels. Often there is more than one class in an image. In that case, the IoU is calculated like here for every class, after which the mean is taken.

We, therefore, normalize the data using the mean from Cityscapes rather than the statistics from the data used for pretraining, which is otherwise a common way of doing it. We believe that this is the main reason for our baseline, i.e. the results from a model trained only on the given amount of labeled data, being slightly different from Hung et al.

The columns in table 4.1 correspond to different amounts of labeled data samples, the remaining samples are used for unsupervised training. For all five works, the results are presented with a baseline for each labeled data amount. The SSL rows show the results using the respective proposed methods. The Delta rows show the difference between the baseline and the SSL results. This is arguably the most important measure, as the absolute results are not only affected by the proposed method but also by the underlying baseline. Our results are presented as the mean of three independent trainings of a model, and the standard deviation of the three values is also displayed. All numbers are mean intersection over union scores. For three out of four data amounts our method achieves the best results, marked by bold font. For 744 labeled data samples our result is lower than that of Feng et al. and French et al. but our Delta is higher, as it is for all amounts of data.

Figure 4.2 shows qualitative results for a baseline model and a model trained with our SSL method, both models are trained with 1/8 of all labeled samples. Predictions are shown for a few images along with the corresponding images and

[1]Same Deeplabv2 network but with Image-Net pretraining instead of MSCOCO

Table 4.1: Results from applying our semi-supervised learning method on the Cityscapes dataset. The columns correspond to different amounts of labeled samples. Our results are compared to four previous articles solving the same task. For each data amount and article, including ours, a supervised baseline is presented along with the result, as well as the difference between the two. All numbers are mean intersection over union. Our results are the mean result from three runs, with the standard deviation also presented.

Labeled samples	1/30 (100)	1/8 (372)	1/4 (744)	1/2 (1488)	Full (2975)
Hung et al.[24]					
Baseline	-	55.5%	59.9%	64.1%	66.4%
SSL	-	58.8%	62.3%	65.7%	-
Delta	-	3.3	2.4	1.6	-
Mittal et al.[25][1]					
Baseline	-	56.2%	60.2%	-	66.0%
SSL	-	59.3%	61.9%	-	65.8%
Delta	-	3.1	1.7	-	-0.2
French et al.[15][1]					
Baseline	44.41%	55.25%	60.57%	-	67.53%
SSL	51.20%	60.34%	63.87%	-	-
Delta	6.79	5.09	3.3	-	-
Feng et al.[62]					
Baseline	45.5 %	56.7%	61.1%	-	66.9%
SSL	48.7 %	60.5%	**64.4%**	-	-
Delta	3.2	3.8	3.3	-	-
Ours					
Baseline	43.84%$\pm$0.71	54.84%$\pm$1.14	60.08%$\pm$0.62	63.02%$\pm$0.14	66.19%$\pm$0.11
SSL	**54.07%**$\pm$1.61	**61.35%**$\pm$0.62	63.63%$\pm$0.33	**66.29%**$\pm$0.47	-
Delta	**10.23**	**6.51**	**3.72**	**3.27**	-

ground truth semantic maps. There are quite clearly less artifacts in the predictions made by the SSL model than those made by the SL model.

4.3.2 Pascal VOC dataset

The results for the Pascal VOC dataset are shown in table 4.2, along with results from four previous papers. The table is structured in the same way as table 4.1. Each figure in our results is taken from only one run because we did not have time for more experiments. We have the same hyperparameters as Hung et al., apart from the number of training iterations, which we increased from 20k to 40k. We note that French et al. have a baseline significantly lower than all others [15]. This is partly because they use a network pretrained on ImageNet [75], rather than on MSCOCO [76], like we are, and MSCOCO is more similar to Pascal VOC than ImageNet is. This means that they have a lot higher potential performance gain, which we believe is a big part of the reason why they obtain such high Deltas. Other differences in experimental setups relative to the other works in table 4.2 include the use of Adam optimizer instead of SGD in the case of French et al., and Feng et al. and Mittal et

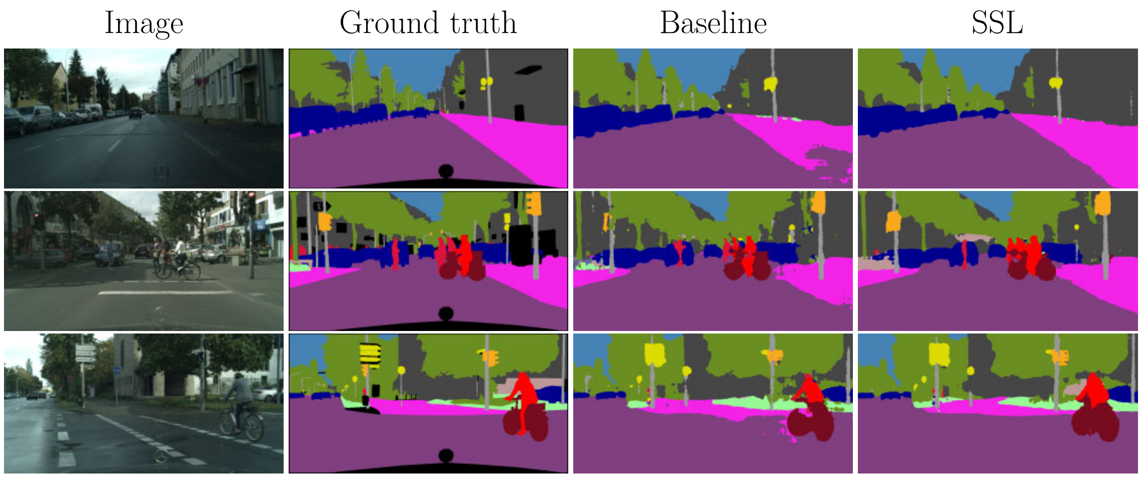


Figure 4.2: Predictions made on three Cityscapes frames using a baseline model trained on 1/8 labeled data and a semi-supervised model trained on the same 1/8 labeled data as well as unlabeled data.

al. use a batch size of 4 and 8 respectively instead of 10.

4.4 Results for unsupervised domain adaptation

Tables 4.3 and 4.4 show the results from using our method to train models on the GTA5 and SYNTHIA datasets as source domains and the Cityscapes dataset as the target domain. In the same tables the results from multiple other papers on unsupervised domain adaptation are also shown. Tsai et al. have almost the same baseline hyperparameters as us [67], with the differences being that we do not employ early stopping and that we crop the images to limit memory usage.

The columns in the tables correspond to the different classes. For all works, we have included their results for DeepLabv2 with ResNet101 backbone. The results are also presented with a baseline model only trained on the source domain, but evaluated on the target domain, where each block of rows share baseline. Our results are presented as the mean of three independent tries. All numbers are intersection over union scores.

We note that we achieve the highest overall performance as well as the highest for several individual classes. As can be observed both in tables 4.3 and 4.4 and figure 4.3, the adapted models from GTA5 perform better than those adapted from SYNTHIA. In figure 4.3 we also observe that the adapted models have significantly less artifacts present, and in contrast to figure 4.2, the segmentations follow the labeling convention of GTA5 and SYNTHIA where for example the bicycles are not fully filled in unlike in the ground truth for Cityscapes. This difference in ground truth segmentation maps among other factors will reduce the mIoU score. Specifically for SYNTHIA, 3 classes are completely absent, but are not ignored in

[2]Reported by [25]

[3]Same Deeplabv2 network but with Image-Net pretraining instead of MSCOCO.

[4]Results for 100 (1/106) samples.

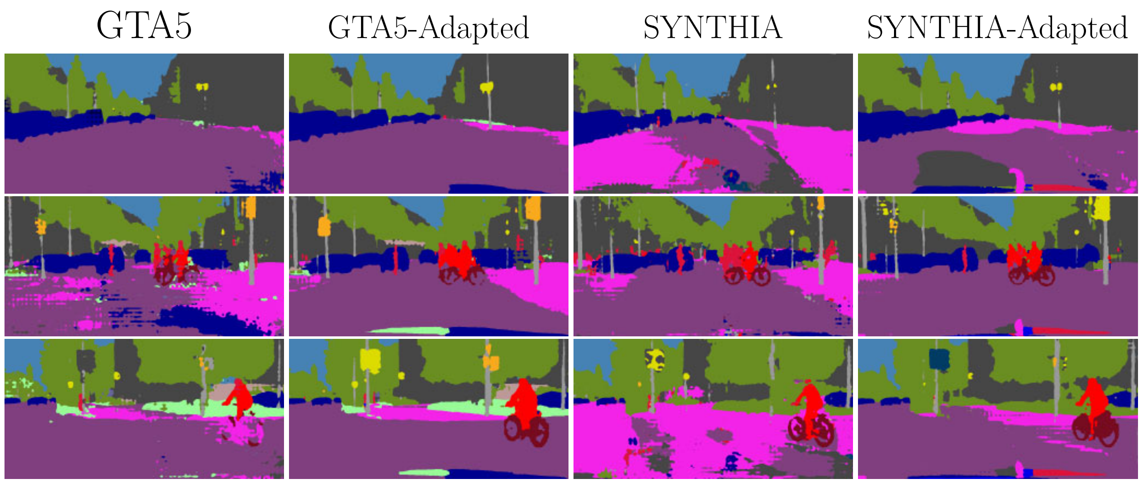


Figure 4.3: Predictions made on the same three Cityscapes frames as in Figure 4.2, using baseline models trained on the GTA5 and SYNTHIA data together with models trained with unsupervised domain adaptation. All of these models have only been trained with ground truth semantic maps for synthetic images.

the performance metric in the calculation of the union, further damaging the score.

Though the hyperparameters are different, a rough comparison can be made with the results in semi-supervised learning, table 4.1. Looking at the results for Cityscapes as target domain and GTA5 as source domain, table 4.3, unsupervised domain adaptation is almost on par with using 744 labeled samples in a supervised learning setting (or 100 semi-supervised). As the process of labeling for Cityscapes reportedly took 1.5 hours per image [2], using our unsupervised domain adaptation solution would mean a significant cost reduction.

Figure 4.3 shows qualitative results for models trained on only the source domains compared to unsupervised domain adaptation models. Predictions are shown for the same images as in Figure 4.2. There are quite clearly less artifacts in the predictions made by the adapted models, with the models trained on GTA5 data performing better.

Table 4.2: Results from applying our method on semi-supervised on the Pascal VOC dataset. The columns correspond to different amounts of labeled samples. Our results are compared to four previous articles solving the same task. For each data amount and article, including ours, a supervised baseline is presented along with the result, as well as the difference between the two. All numbers are mean intersection over union. Our results are from a single run, as we did not have time to run more experiments than that.

Labeled samples	1/100	1/50	1/20	1/8	1/4	Full (10582)
Hung et al.[24]						
Baseline	-	53.2%[2]	58.7%[2]	66.0%	68.3%	73.6%
SSL	-	57.2%[2]	64.7%[2]	69.5%	72.1%	-
Delta	-	4.0	6.0	3.5	**3.8**	-
Mittal et al.[25]						
Baseline	-	53.2%	58.7%	66.0%	-	73.6%
SSL	-	63.3%	67.2%	**71.4%**	-	75.6%
Delta	-	10.1	8.5	5.4	-	2.0
French et al.[15][3]						
Baseline	33.09%	43.15%	52.05%	60.56%	-	72.59%
SSL	53.79%	64.81%	66.48%	67.60%	-	-
Delta	**20.70**	**21.66**	**14.48**	**7.04**	-	-
Feng et al.[62]						
Baseline	45.7%[4]	55.4%	62.2%	66.2%	68.7%	73.5%
SSL	**61.6%**[4]	65.5%	**69.3%**	70.7%	71.8%	-
Delta	15.9	10.1	7.1	4.5	3.1	-
Ours						
Baseline	42.47%	55.69%	61.36%	67.14%	70.20%	74.13%
SSL	54.18%	**66.15%**	67.77%	71.00%	**72.45%**	-
Delta	11.71	10.46	6.41	3.86	2.25	-

Table 4.3: Results from applying our method on unsupervised domain adaptation on the GTA5 dataset as the source domain and the Cityscapes dataset as the target domain. The columns correspond to the different classes. Our results are compared to previous articles solving the same task with the same network. For each work, including ours, their baseline only trained on the source domain is presented along with the result. All numbers are mean intersection over union. Our results are the average from three runs.

Method	Road	SW	Build	Wall	Fence	Pole	TL	TS	Veg	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	MC	Bike	mIoU
Source	75.8	16.8	77.2	12.5	21.0	25.5	30.1	20.1	81.3	24.6	70.3	53.8	26.4	49.9	17.2	25.9	6.5	25.3	36.0	36.6
AdaptSegNet [67]	86.5	36.0	79.9	23.4	23.3	23.9	35.2	14.8	83.4	33.3	75.6	58.5	27.6	73.7	32.5	35.4	3.9	30.1	28.1	42.4
SIBAN [46]	88.5	35.4	79.5	26.3	24.3	28.5	32.5	18.3	81.2	40.0	76.5	58.1	25.8	82.6	30.3	34.4	3.4	21.6	21.5	42.6
CLAN [68]	87.0	27.1	79.6	27.3	23.3	28.3	35.5	24.2	83.6	27.4	74.2	58.6	28.0	76.2	33.1	36.7	6.7	31.9	31.4	43.2
APODA [69]	85.6	32.8	79.0	29.5	25.5	26.8	34.6	19.9	83.7	40.6	77.9	59.2	28.3	84.6	34.6	49.2	8.0	32.6	39.6	45.9
PatchAlign [48]	**92.3**	51.9	82.1	29.2	25.1	24.5	33.8	33.0	82.4	32.8	82.2	58.6	27.2	84.3	33.4	46.3	2.2	29.5	32.3	46.5
AdvEnt [70]	89.4	33.1	81.0	26.6	26.8	27.2	33.5	24.7	83.9	36.7	78.8	58.7	30.5	84.8	38.5	44.5	1.7	31.6	32.4	45.5
Source	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	29.2
FCAN [71]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	46.6
Source	71.3	19.2	69.1	18.4	10.0	35.7	27.3	6.8	79.6	24.8	72.1	57.6	19.5	55.5	15.5	15.1	11.7	21.1	12.0	33.8
CBST [50]	91.8	53.5	80.5	32.7	21.0	34.0	28.9	20.4	83.9	34.2	80.9	53.1	24.0	82.7	30.3	35.9	16.0	25.9	42.8	45.9
MRKLD [51]	91.0	**55.4**	80.0	33.7	21.4	37.3	32.9	24.5	85.0	34.1	80.8	57.7	24.6	84.1	27.8	30.1	**26.9**	26.0	42.3	47.1
BDL [72]	91.0	44.7	84.2	34.6	27.6	30.2	36.0	36.0	85.0	43.6	83.0	58.6	31.6	83.3	35.3	49.7	3.3	28.8	35.6	48.5
CADASS [73]	91.3	46.0	84.5	34.4	29.7	32.6	35.8	36.4	84.5	43.2	83.0	60.0	32.2	83.2	35.0	46.7	0.0	**33.7**	42.2	49.2
Source	51.1	18.3	75.8	18.8	16.8	34.7	36.3	27.2	80.0	23.3	64.9	59.2	19.3	74.6	26.7	13.8	0.1	32.4	34.0	37.2
MRNet [74]	89.1	23.9	82.2	19.5	20.1	33.5	42.2	39.1	85.3	33.7	76.4	60.2	33.7	86.0	36.1	43.3	5.9	22.8	30.8	45.5
R-MRNet [52]	90.4	31.2	85.1	**36.9**	25.6	37.5	**48.8**	48.5	85.3	34.8	81.1	64.4	**36.8**	**86.3**	34.9	**52.2**	1.7	29.0	**44.6**	50.3
Source:	63.31	15.65	59.39	8.56	15.17	18.31	26.94	15.00	80.46	15.25	72.97	51.04	17.67	59.68	28.19	33.07	3.53	23.21	16.73	32.85
Ours:	89.90	39.66	**87.87**	30.71	**39.52**	**38.52**	46.43	**52.79**	**87.98**	**43.96**	**88.76**	**67.20**	35.78	84.45	**45.73**	50.19	0.00	27.25	33.96	**52.14**

Table 4.4: Results from applying our method on unsupervised domain adaptation on the SYNTHIA dataset as the source domain and the Cityscapes dataset as the target domain. The columns correspond to the different classes. Our results are compared to previous articles solving the same task with the same network. For each work, including ours, their baseline only trained on the source domain is presented along with the result. All numbers are mean intersection over union. Our results are the average from three runs. Some earlier works only present their results for 13 of the 16 classes, so for a fair comparison we include results for this as well.

Method	Road	SW	Build	Wall*	Fence*	Pole*	TL	TS	Veg	Sky	Person	Rider	Car	Bus	MC	Bike	mIoU*	mIoU
Source	55.6	23.8	74.6	-	-	-	6.1	12.1	74.8	79.0	55.3	19.1	39.6	23.3	13.7	25.0	38.6	-
AdaptSegNet [67]	84.3	42.7	77.5	-	-	-	4.7	7.0	77.9	82.5	54.3	21.0	72.3	32.2	18.9	32.3	46.7	-
SIBAN [46]	82.5	24.0	79.4	-	-	-	16.5	12.7	79.2	82.8	58.3	18.0	79.3	25.3	17.6	25.9	46.3	-
CLAN [68]	81.3	37.0	80.1	-	-	-	16.1	13.7	78.2	81.5	53.4	21.2	73.0	32.9	22.6	30.7	47.8	-
APODA [69]	86.4	41.3	79.3	-	-	-	22.6	17.3	80.3	81.6	56.9	21.0	84.1	**49.1**	24.6	45.7	53.1	-
PatchAlign [48]	82.4	38.0	78.6	8.7	0.6	26.0	3.9	11.1	75.5	84.6	53.5	21.6	71.4	32.6	19.3	31.7	46.5	40.0
AdvEnt [70]	85.6	**42.2**	79.7	8.7	0.4	25.9	5.4	8.1	80.4	84.1	57.9	23.8	73.3	36.4	14.2	33.0	48.0	41.2
Source	64.3	21.3	73.1	2.4	1.1	31.4	7.0	27.7	63.1	67.6	42.2	19.9	73.1	15.3	10.5	38.9	40.3	34.9
CBST [50]	68.0	29.9	76.3	10.8	1.4	33.9	22.8	29.5	77.6	78.3	60.6	28.3	81.6	23.5	18.8	39.8	48.9	42.6
MRKLD [51]	67.7	32.2	73.9	10.7	1.6	**37.4**	22.2	31.2	80.8	80.5	60.8	29.1	82.8	25.0	19.4	45.3	50.1	43.8
CADASS [73]	82.5	**42.2**	81.3	-	-	-	18.3	15.9	80.6	83.5	61.4	33.2	72.9	39.3	26.6	43.9	52.4	-
Source	44.0	19.3	70.9	8.7	0.8	28.2	16.1	16.7	79.8	81.4	57.8	19.2	46.9	17.2	12.0	43.8	40.4	35.2
MRNet [74]	82.0	36.5	80.4	4.2	0.4	33.7	18.0	13.4	81.1	80.8	61.3	21.7	84.4	32.4	14.8	45.7	50.2	43.2
R-MRNet [52]	**87.6**	41.9	**83.1**	14.7	1.7	36.2	**31.3**	19.9	81.6	80.6	63.0	21.8	**86.2**	40.7	23.6	**53.1**	**54.9**	47.9
Source:	36.30	14.64	68.78	9.17	0.20	24.39	5.59	9.05	68.96	79.38	52.45	11.34	49.77	9.53	11.03	20.66	33.65	29.45
Ours:	80.56	25.12	81.90	**21.46**	**2.85**	37.20	22.67	**23.99**	**83.69**	**90.77**	**67.61**	**38.33**	82.92	38.90	**28.49**	47.58	54.81	**48.34**

5 Discussion

In this chapter, we discuss the results presented in the previous chapter. We do this with the help of an ablation study, in which we have tried our method with different settings than those described in section 3.2. We also go into greater depth discussing model output confidence and spatial consistency, both relevant aspects of our solution. Lastly, we also present suggestions for future research related to our work.

5.1 Discussion of results

5.1.1 Discussion of results for semi-supervised learning

Our results for semi-supervised learning sets new state-of-the-art results on the Cityscapes dataset, table 4.1. We note that this is both in terms of performance relative to the supervised baseline, as well in absolute terms for all but one data amount, despite the better baseline of some previous research.

For the Pascal VOC dataset we achieve better results than existing research for two data amounts, meaning that our method is competitive with the state of the art, as can be seen in table 4.2. The results are, however, not quite as strong as they are for Cityscapes. We attribute this mostly to two factors. Firstly, the variability of ClassMasks will be much smaller for the Pascal VOC images, since each image contains very few classes, often only the background class and one object. This means that the masks will be very similar for each image between training epochs. However, more important is that since there are so few classes it is not at all certain that approximately half of the pixels are transferred in the mixing, as is the case for Cityscapes. Instead, it is often the case that almost all pixels are transferred in the case where one object covers most of the image, or that almost no pixels are transferred in the case where the object is very small. This will lead to the strongly augmented image not being much different from the weakly augmented image, which the pseudo-label is based on, breaking the principle of augmentation anchoring.

Secondly, we believe that one reason for our method performing well on the Cityscapes dataset is due to images being quite similar within the dataset. This can be exemplified by spatial consistency, which is discussed further in section 5.1.3.

In our ablation study for semi-supervised learning, section 5.2, we find that additional improvements to the results can be obtained by cropping images, training for more iterations, and using additional augmentations. Since we did not include these in our main results, it is clear that the reported performance increase relative to previous research comes from our new mixed sample data augmentation strategy ClassMix paired with pseudo-labeling. It also means that we could have obtained even stronger results, had we further optimized the algorithm.

5.1.2 Discussion of results for unsupervised domain adaptation

Our unsupervised domain adaptation solution sets new state-of-the-art results in terms of mean intersection over union for both GTA5 to Cityscapes and SYNTHIA to Cityscapes, Tables 4.3 and 4.4. We note that for some classes our method is consistently performing worse than previous methods, though for most classes it is better. Among all comparisons only the methods CBST [50] and MRKLD [51] and their corresponding baseline (coming from the same authors), consistently predicted the ”train” class. For these results, the training time was kept longer than for SSL, which is in line with previous work [67]. We note that our baseline (the same network only trained on the source domain without adaptation) performs worse than in most other research, making our improvement from the baseline even larger.

Some earlier reported state-of-the-art results in domain adaptation employ early stopping based on the validation set of Cityscapes, while simultaneously using it as test set. This is in our opinion bad practice, as noted in section 4.1.2, which is why we do not use it. When training, we did, however, evaluate our models regularly, even though these results were never used for model selection. Therefore we can, for reference, report that for GTA5 to Cityscapes we would have achieved an (average over 3 runs) mIoU of 53.84% instead of 52.14% had early stopping been used, and for SYNTHIA 49.10% instead of 48.34%. The baseline from GTA5 would go from 32.85% to 35.68%, and for SYNTHIA from 29.45% to 32.85%. The reason for this considerable increase in performance from early stopping is that performance varies a lot over the course of training, rather than that the model was overfitting the training data. Despite us not using early stopping we achieve the best overall performance compared to all other methods known to us. It is also worth noting that several works have altered the baselines considerably relative to Tsai et al., and tuned the hyperparameters based on the testing data. This was also something we wanted to avoid, as there is otherwise a risk of overfitting the test data. We have thus tried to keep hyperparameters in line with the original authors.

It is also interesting to note that where a lot of recent solutions have focused on offline self-training with multiple rounds of re-training, adding complicated modules for handling uncertainty and class-balanced sampling, our solution is by comparison much simpler and more general; we simply load batches of labeled synthetic data in parallel with unlabeled data in the target domain, and for each iteration process them together with ClassMix. This simplicity makes our state-of-the-art results particularly notable.

5.1.3 Spatial consistency

We hypothesise that mask-based mixed sample data augmentations, like the ones we have investigated in this thesis, owe a part of their success in urban scene seg-

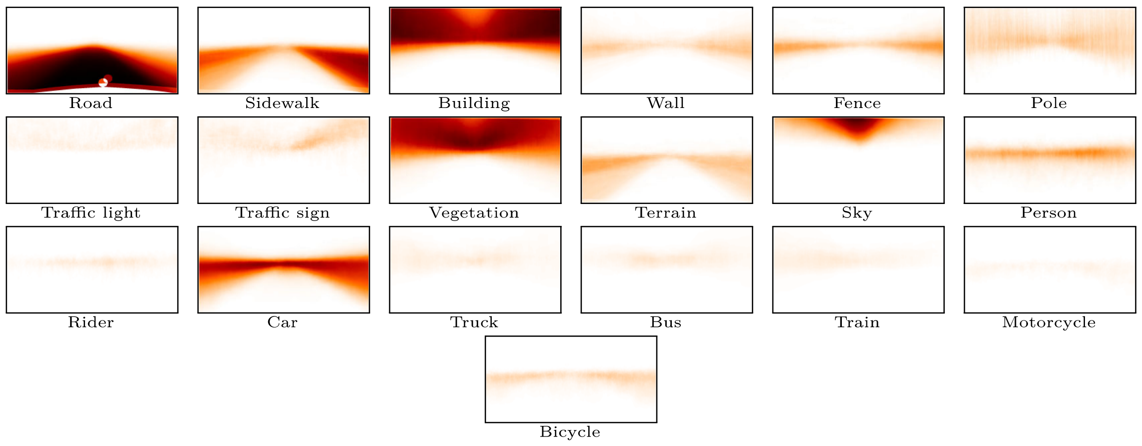


Figure 5.1: Spatial distribution of all classes in the Cityscapes training dataset. Dark pixels correspond to more frequent appearance. It is clear that most, if not all, classes are limited to appear in only a specific part of the images.

mentation to the spatial consistency of images. By this we mean that objects of a certain class are usually found in the same place in different images, road is always down and sky is always up and so on. This has as an effect that when pasting a part of one image over to another image, it is likely that the formed image is not too far from the original data distribution. This is likely true for all mixed sample data augmentations, but especially so for ClassMix because single objects are often pasted over, meaning that the context of the background image is very important to the realism of the resulting image.

Figure 5.1 shows the spatial distribution of classes in the Cityscapes dataset, which is what one can expect. It can be compared to the distributions in the GTA5 and SYNTHIA datasets shown in figure 5.2 and 5.3 respectively. It is easy to see that the spatial distributions of Cityscapes and GTA5 are very similar, whereas SYNTHIA differs more. This is not unexpected, as both Cityscapes and GTA5 images are first person views from a car, while SYNTHIA images have a variety of angles including bird's eye views which naturally have a different spatial distribution. Part of the reason why the results are higher for training using GTA5 data as opposed to SYNTHIA data is therefore likely the higher spatial consistency between Cityscapes and GTA5. For example, models trained using SYNTHIA struggle with discerning sidewalk from road, as can be seen in figure 4.3 and table 4.4. For example, in figure 4.3 the adapted SYNTHIA model mistakes a large piece of road for sidewalk for one example image, and completely misses a sidewalk in another. This could be because road and sidewalk can be very similar in texture, and whether it is one or the other has to be inferred from the surroundings. Since the spatial distribution of SYNTHIA deviates from Cityscapes, ClassMix will often mix road or sidewalk independently to Cityscapes causing its semantic meaning to change.

The spatial distributions can be compared to figure 3.9 showing the relative frequency of occurrence of classes. For example, as can be seen in that figure, there are very few bicycles in the GTA5 dataset, which can also be seen in the spatial distribution as the bicycle distribution is almost not visible.

The spatial distribution of the Pascal VOC dataset is not nearly as consistent

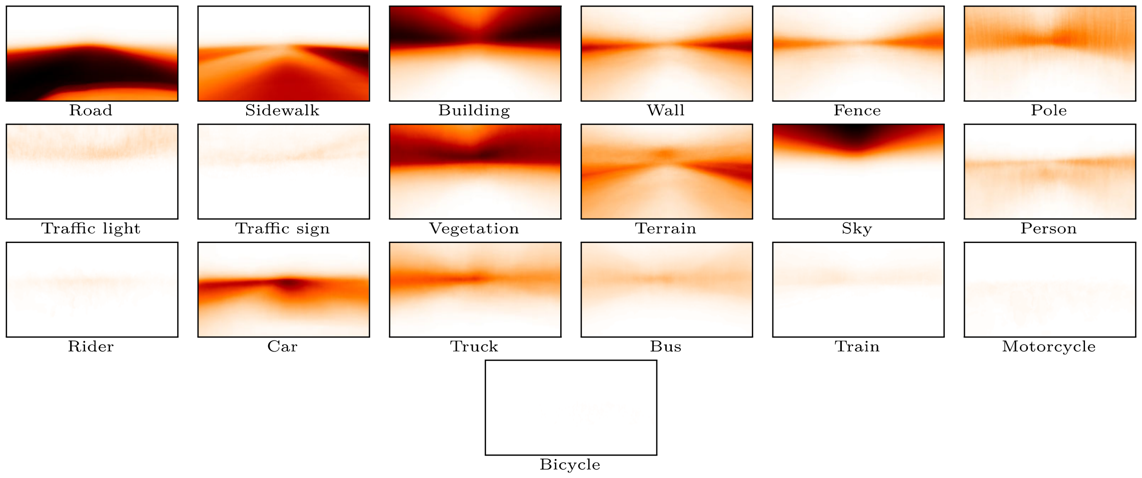


Figure 5.2: Spatial distribution of all classes in the GTA5 dataset. The classes are mostly confined to specific regions of the images, although not quite as much so as for the Cityscapes data.

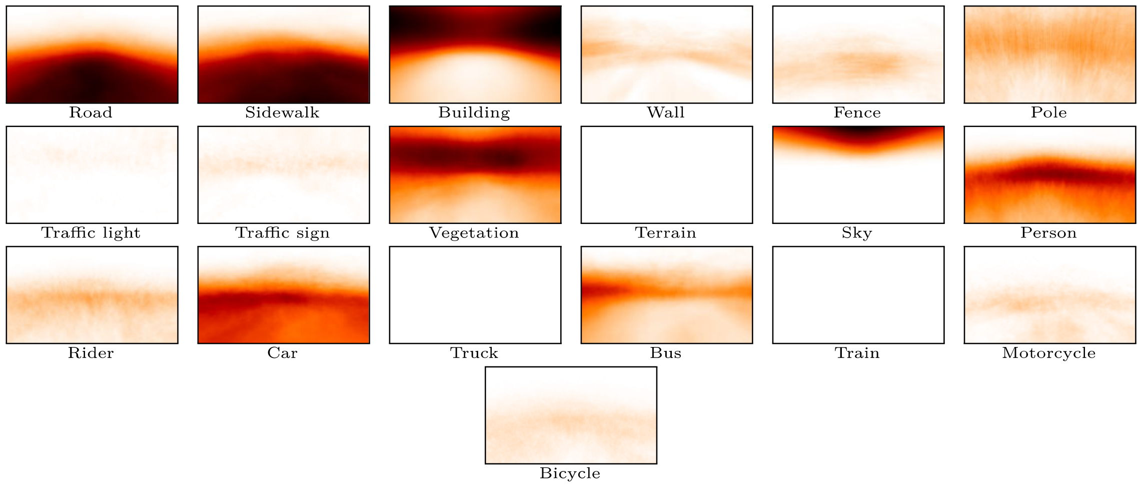


Figure 5.3: Spatial distribution of all classes in the SYNTHIA dataset. Dark pixels correspond to more frequent appearance. The classes Terrain, Truck and Train are not present in the SYNTHIA dataset, and their corresponding spatial distributions are therefore empty, but they are included here for easier comparison to Cityscapes and GTA5. The classes are not as confined to specific regions here as in the Cityscapes and GTA5 datasets.

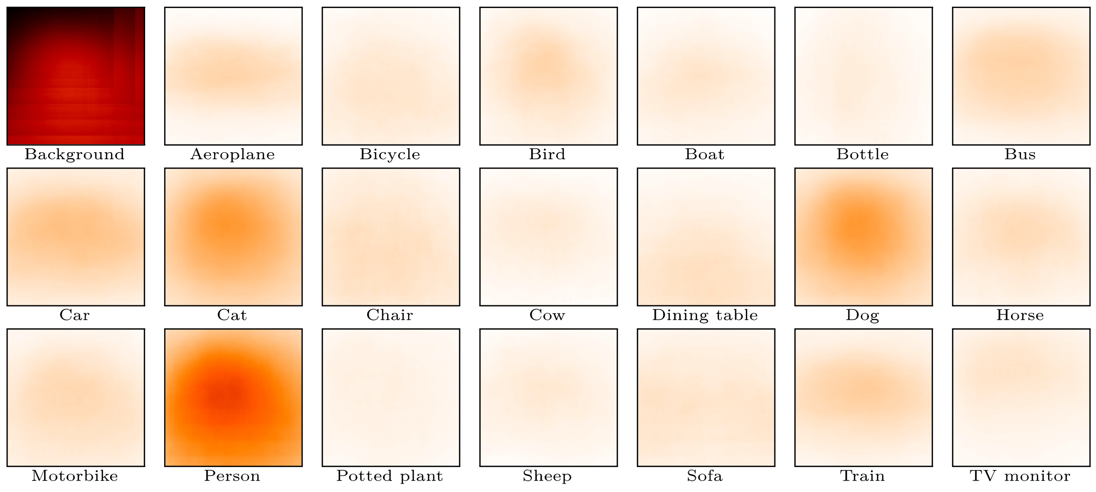


Figure 5.4: Spatial distribution of all classes in the Pascal VOC training dataset. Dark pixels correspond to more frequent appearance. All classes are very homogeneously spread out spatially.

as for the Cityscapes dataset. Figure 5.4 clearly shows that the classes are almost homogeneously spread out. This means that when using ClassMix it is not likely that pasted objects will end up in reasonable contexts, which was one of the original motivations for ClassMix as a method. Therefore, we believe that the mixed images are further from the real data distribution for Pascal VOC than for Cityscapes.

5.2 Ablation study

Here, we present ablation studies for SSL and UDA, meaning that we investigate our methods further to find what contribution the different aspects of the algorithm give to the results. The studies are performed such that for each setting that is tried, only that aspect of the algorithm changes, the rest is identical to our default algorithm described in section 3.2. All results are mean values from three runs. For SSL we use the Cityscapes dataset, and for UDA we use the GTA5 and Cityscapes datasets.

5.2.1 Ablation for semi-supervised learning

Apart from the supervised baseline and our default SSL solution, 11 different settings have been tried, results are shown in table 5.1. We have tried two additional mixing algorithms, CutMix [14] and CowMix [15]. The unsupervised loss weighting has been experimented with, trying a sigmoid ramp up and pixel-wise threshold, as well as keeping the weight constant, see section 2.5.4. We have tried using squared error as our unsupervised loss function L_u, instead of cross-entropy. Pseudo-labels have been replaced with using the unprocessed output distributions as targets. We have tried applying random cropping to images. The extra augmentations color jitter and Gaussian blurring have been applied in the strong augmentation scheme, and we tried training models for 80k iterations instead of 40k. A detailed analysis of the

Table 5.1: Ablation study for SSL on the Cityscapes dataset. Apart from the SL baseline and our default SSL solution, we try two different mixed sample data augmentations, three different weight functions for the unsupervised loss, sum of squared errors as unsupervised loss function, not using pseudo-labels, random cropping of images, applying extra augmentations for the unsupervised images and training for 80k iterations instead of 40k.

Settings	mIoU
Baseline	54.84%
Default SSL	61.35%
CowMix	60.37%
CutMix	59.12%
Sigmoid ramp up	60.58%
Pixelwise threshold	58.61%
Constant unsupervised weight	60.58%
Squared error loss	58.74%
No Pseudolabel	60.15%
Random crop Baseline	56.42%
Random crop	62.16%
Extra augmentations	61.85%
80k iterations	62.92%

ablation results now follows, all results are for 1/8 (372) of the labeled samples and are averaged over three independent runs.

As is evident from table 5.1, ClassMix is the best performing mixed sample data augmentation, followed by CowMix and lastly CutMix, with approximately 1% gaps. The reason for CutMix performing worst is believed to be because of the limitations to its variability, as noted in section 2.5.2. It only has four degrees of freedom, the two side lengths w and h and the two locational coordinates of the rectangle, giving approximately $w^2 \cdot h^2 \sim 10^{11}$ possible combinations for images of size 512×1024. This may seem like a high number, but many mixes will be very similar to each other. This can be put in contrast with CowMix and ClassMix which both, in principle, have as much potential variability as the image size allows, each pixel could vary independently of its neighbours, giving $2^{h \cdot w} \sim 10^{157000}$ possible combinations, which is obviously vastly bigger than that of CutMix. In practice the variability is not only bounded by the image size however, meaning the number is not quite this high. For CowMix, the Gaussian field is generated using a σ that we draw from a bounded distribution, meaning that the features are within a certain range of sizes, and for ClassMix the pixels selected for the mask likely belong to the same classes, meaning that they are clustered together. The variability is still much higher for these two, however.

Since the boundaries of the mask are always straight and orthogonal to the image in CutMix, it becomes much easier for the network to discriminate between the two mixed images. This likely also hurts performance, as strong augmentations have been shown to be beneficial for consistency regularization [13, 11, 15]. The fact that ClassMix performs this much better than CutMix seems likely to be the main reason for our results being stronger than those of French et al., as they are using CutMix [15].

That ClassMix performs better than CowMix, we attribute to the fact that ClassMasks will, to a high degree, follow the boundaries of objects in the images. Like we argued in section 3.1, this means that mixed images will look more like real images, and will therefore provide a more useful training signal.

Another important parameter is the weight factor determining the relative size between the supervised and unsupervised loss components as a function of training time, with results varying a lot between different variants. Table 5.1 shows results from using a sigmoid ramp up, pixel-wise threshold, and no loss weight function at all, see section 2.5.4. All options use a scaling factor $\lambda_{u0} = 1$.

The sigmoid is defined in equation 2.13, with slope parameter $c = 5$ as in [4, 5], and $k_{max} = 10000$ iterations, this forms the curve shown in figure 2.9. This gives an mIoU score slightly below that of using a threshold-proportional weight. The reason for this could be the general shape of the curve, the weight is lower for the sigmoid in the beginning of training, and higher from approximately 10k iterations and on, why this would be better or worse is not clear though. If this is indeed why the results are better, it is quite likely that they could be improved further by fine-tuning the weight, since the scaling of the weight has not been experimented with at all, and the relatively small difference between the two curves in figure 2.9 gives such a large change in performance.

More likely is that the reason for the sigmoid's inferior performance is that the threshold-proportional weight is self-adjusting, meaning that each batch is weighted differently depending on the network's certainty on that particular batch. The variance between batches is relatively large, as can be seen in figure 2.9, with weight values varying between approximately 0.7 and 0.9 after the first few thousand iterations. This means that hard samples give a smaller training signal than easier samples, yielding something similar to curriculum learning, where a model first is trained on easy samples and then on harder samples as training progresses, which has proven useful in other tasks [77, 78]. If this is the prime reason for the better performance of threshold-proportional weight relative to sigmoid ramp up, it is possible that the results would improve further by basing the weights on individual samples rather than on whole batches, like we are doing now, since the batch size is only two for the Cityscapes experiments this would most likely not make a huge difference, however.

The pixel-wise thresholded loss in table 5.1 is the one described in section 2.5.4. It is performing significantly worse than the default threshold-proportional weight. This is likely because many pixels, around 15% according to figure 2.9, are not used for training at all. These pixels are mainly the ones close to class boundaries, as argued in section 2.5.4. This fact is illustrated in figure 5.5, where the certainties of a prediction are illustrated, as well as which pixels are above a threshold τ. We can clearly see that the boundary pixels are indeed mainly the ones with certainties below this threshold, meaning they are getting more or less entirely excluded from the unsupervised loss. In addition to losing signal at boundaries, small or difficult classes can be masked out as well, which could potentially be alleviated by setting different thresholds depending on classes which would also lead to more balanced

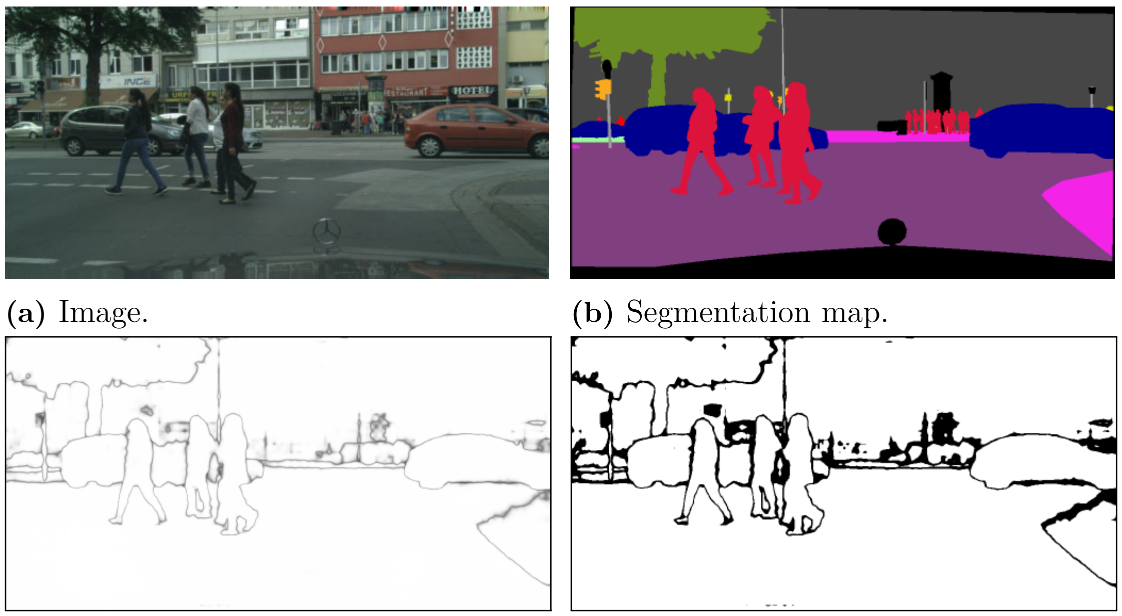

(a) Image.
(b) Segmentation map.
(c) Confidence map.
(d) Thresholded confidence.

Figure 5.5: An illustration of the certainty of the output of the network. **(a)** shows the image, and **(b)** shows the corresponding semantic map. In **(c)** the max probabilities in the output for each pixel is shown. It is clear that the network gives less certain predictions close to class boundaries. In **(d)** a binary mask is shown, where white pixels are pixels where the certainty of the output of the network is above a threshold $\tau = 0.968$ (as used in [6]), and black pixels are below the threshold. We can see that if thresholding the loss like this, boundary pixels would to a high degree be left out.

sampling [50], [62]. This was not further investigated in our thesis, however.

Having no ramp up of the unsupervised loss weight at all has also been investigated. We set the weight to be a constant one. The performance is here the same as it is for using sigmoid ramp up, see table 5.1. The conclusion from this is that it does not seem to matter much that the weight is incrementally increasing over the course of training. The sigmoid ramp up is identical to the constant weight after 10k iterations, having a sigmoid that converges later might give a different result.

We have also tried changing our default cross-entropy unsupervised loss function with the sum of squared error from equation 2.1. This resulted in approximately 2.5% lower performance, as can be seen in table 5.1. This is expected, as cross entropy, in general, is better suited for classification tasks than squared loss, which is also why cross entropy is more widely used. It is worth noting that the unsupervised weight λ_{u0} was kept at one, the same as when using cross-entropy loss, and no additional tuning of it was made. This caused the unsupervised loss to be roughly half as large when using squared error loss as when using cross-entropy, which might have had a large impact on the results. It is also worth mentioning that French et al. use squared error loss with an unsupervised weight of one [15].

Not using pseudo-labeling, but instead training the model against the softmax output distribution from the weakly augmented unlabeled samples, produced results approximately one percent lower than when using pseudo-labels. This strengthens the point made in section 2.5.3, that it is advantageous to minimize entropy in the targets. We hypothesise that it is especially useful when using mixed sample data augmentations in general, and ClassMix in particular. The reason for this is that the context of classes changes, and hence the uncertainty between classes close to class boundaries can become unwarranted, see section 3.1 for a more detailed description of the issue.

For our experiments with semi-supervised learning on the Cityscapes dataset, we did not do any random cropping. We did, however, try performing random crops of 512×512 on the 512×1024 images, as described in section 3.4, for both a supervised baseline as well as for semi-supervised. This did not only speed up training considerably but also improved upon the baseline as well as the SSL performance, as can be seen in Table 5.1. The delta, however, is lower. The reason for the smaller delta could be that the cropped images do not respect spatial consistency as well as full images, see section 5.1.3, yielding a smaller relative improvement from the mixing. The reason could also simply be that the stronger supervised baseline leaves less room for improvement by semi-supervised learning.

It is worth noting that we also experimented with cropped images for other amounts of labeled data amounts, and the performance of models relative to not cropping the images decreased as the amount of labeled data increased. The reason that we are not using cropped images in our main results is that we want to stay as close as possible to the hyperparameters used by Hung et al. [24].

In our default SSL solution, we are not using any augmentations other than the mixing of images using ClassMix. Here, we have tried adding Gaussian blurring and Color jittering, as described in section 3.4. Both augmentations have a 50% chance of being applied for each unsupervised batch and are applied after the mixing.

The performance is increased by 0.5% when extra augmentations are applied, compared to the default solution, as can be seen in table 5.1. The reason for this could stem from the fact that after mixing images the boundaries between the mixed images will look more or less unnatural, even in the case of ClassMix where the resulting images look more realistic. This is unwanted because more realistic images lie closer to the underlying data distribution, which helps the training of models, further discussed in section 3.1. However, when blurring and color jittering is applied after the mix, the image will be less clear and boundaries will hence be less visible.

Another reason for this could just be the fact that the strong augmentation scheme when using blurring and color jittering is stronger than when using just mixing. This would provide the network with more varied data, which in general is positive, so long as the variations are not too large. This part of the ablation study was performed after the main results had been generated, if not, this would have been included in the default solution, hopefully granting stronger results for all amounts of labeled data. We point out however that these augmentations were used in our experiments on unsupervised domain adaptation.

We also tried training models for more iterations, when training a model for 80k iterations instead of the default 40k iterations, all other things being identical, performance climbs approximately 1.5%. This is expected, as overfitting has not seemed to be an issue in our method, possibly because of the large variation in the mixed images. It is likely that training for an even longer time would yield even stronger results, however with diminishing return. The reason that we did not train for longer than 40k iterations for the main results is that most previous works on semi-supervised semantic segmentation have trained for 40k iterations for the Cityscapes dataset [24, 25, 15]. It would also have increased the requirement of computational resources, something we could not meet. If we had been training for longer it is likely that both our results and our Delta in table 4.1 would be larger, further increasing the improvement from previous research. The fact that semi-supervised learning benefits from longer training times is consistent with state-of-the-art semi-supervised learning [13] as well as the related task of contrastive learning [54]. This also makes sense for ClassMix as it artificially expands the dataset. It is worth noting that larger batch sizes of the unlabeled data have also been shown to be important, which makes sense as pseudo-labels are noisy, but their average is likely more correct. We were, however, unable to try this due to computational- and time constraints.

5.2.2 Ablation for unsupervised domain adaptation

For unsupervised domain adaptation, our ablation study concerns the effect of mixing images across domains, InterMix, in contrast to using the same pipeline as for SSL, where only the target domain is mixed, IntraMix. We also attempt using distribution alignment for the case of IntraMix. The GTA5 dataset is the source domain and the Cityscapes dataset is the target domain.

Table 5.2: Ablation study for unsupervised domain adaptation on the GTA5 dataset as source domain and Cityscapes as target domain. Apart from the model only trained on the source domain and our default solution which we here call InterMix, we illustrate the effects of not mixing across domains (just within the target domain) which we call IntraMix. We also include the results of deploying distribution alignment on IntraMix with the target class distribution based on the ground truth labels of the target domain.

Method	Road	SW	Build	Wall	Fence	Pole	TL	TS	Veg	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	MC	Bike	mIoU
Source:	63.31	15.65	59.39	8.56	15.17	18.31	26.94	15.00	80.46	15.25	72.97	51.04	17.67	59.68	28.19	33.07	3.53	23.21	16.73	32.85
InterMix:	89.90	39.66	87.87	30.71	39.52	38.52	46.43	52.79	87.98	43.96	88.76	67.20	35.78	84.45	45.73	50.19	0.00	27.25	33.96	52.14
IntraMix:	84.78	0.00	82.81	0.34	0.05	10.56	47.96	58.86	86.87	8.08	90.99	56.09	0.00	86.92	40.45	11.38	0.00	0.45	0.00	35.08
Distribution Aligned IntraMix:	85.05	28.88	86.79	16.92	36.89	30.38	49.73	53.91	85.61	32.20	92.78	66.61	23.53	84.00	34.81	27.70	0.20	16.65	60.08	48.04

As described in section 3.2 we mix images from the source and target domains, what we call InterMix, rather than mixing images only from the target domain which is what we do in our SSL solution. The reason that we mix in a different way here is to avoid entropy collapse, section 2.5.3, where larger classes would otherwise dominate the transfer, merging with smaller classes. We here present results mixing only within the target domain, meaning the unsupervised pipeline is identical to that in figure 3.3. We call this method IntraMix.

The results are shown in table 5.2, where the performance of IntraMix is seen to be considerably reduced relative to InterMix, due to several classes not being predicted at all. We hypothesise that the domain difference causes the network to discriminate between labeled and unlabeled samples. Pseudo-labeling, section 2.5.3, is a way to enforce entropy-minimization [18]. This is done indirectly in our case, since the gradient does not pass through the teacher network and the pseudo-labels are generated on unperturbed samples and enforced on mixed samples. With that said we hypothesise that since the network only sees pseudo-labels in the target-domain it possibly learns to assign itself easy to predict pseudo-labels to minimize its objective. The lower the entropy in the pseudo-label, the lower the loss, thus instead of learning to adapt to the target domain by being consistent, the network learns to adapt by collapsing the entropy in the pseudo-labels. The predictions in the source-domain for IntraMix was observed to not collapse into low-entropy, giving credence to the idea that the network learns to assign itself low-entropy pseudo-labels in the target domain by discriminating between domains. This idea is also strengthened by the fact that IntraMix is identical to our semi-supervised solution, with the labeled data representing the source domain and the unlabeled data the target domain (though with no domain difference), and in those experiments no minimal entropy-collapse in the pseudo-labels was observed.

To test this hypothesis further, and make sure that the superior performance of InterMix comes from the fact that entropy is injected into the pseudo-labels rather than some other factor, we investigated whether similar results could be achieved with IntraMix but with entropy forced into the labels in another way. We accomplished this by assuming that we know the class distribution of the target domain, and use it as the target distribution in distribution alignment, described in section 2.5.6. In our implementation of distribution alignment, we use an exponential moving average (with parameter $\alpha = 0.99$) of the predictions (pseudo-labels) as the current distribution. The target distribution is calculated prior to training using the labels of the target domain. Since in a real scenario these labels do not exist, the distribution would have to be estimated in some way for it to be a practical solution. It is worth noting that basing the target distribution on the source domain is often not a viable solution considering the differences in class distribution can vary substantially, see figure 3.9. Here the ground-truth labels were used to calculate the target distribution in order to showcase that the issues of IntraMix can be alleviated by distribution alignment. We note that the performance of Distribution Aligned IntraMix is similar to that of InterMix, showing that alternative ways of forcing entropy into the pseudo-labels are possible.

5.3 Future work

We here give a few suggestions for future areas of research related to ours, that we did not have the opportunity to pursue.

5.3.1 Improved mixing

For the ClassMix algorithm, a mask is generated by selecting half of the classes belonging to one of the images. There are a lot of possible variations of this that could be experimented with, such as combining more than two images, oversampling more difficult classes to be pasted, mix correlated classes such as rider (distinct from the person class in Cityscapes) together with their respective vehicles, or not to base it on entire classes but on instances of classes instead. One thing that stands out is that no consideration is taken to the image pasted upon. This could be important to do to produce more realistic images, as well as handling context in a better way. For example, the difference between what is a sidewalk and what is a road in the case of urban scene segmentation can often be impossible to tell in isolation based purely on texture, but instead needs to be inferred from its surroundings.

Specifically for unsupervised domain adaptation, the mix happens across domains, so an interesting direction could be to attempt style transfer from the source- to the target domain prior to pasting.

5.3.2 Offline self-training

In our solution, mini-batches of labeled and unlabeled data are processed in parallel during training. This is similar to many previous works in semi-supervised learning [13, 12, 24, 25, 15] and domain adaptation [67]. For us, the pseudo-labels are assigned by the teacher network. A different way to do it is to use offline training; where pseudo-labels are generated for the entire unlabeled dataset multiple times and the network is retrained with these generated pseudo-labels included. This has been done for both semi-supervised learning [62, 56] and domain adaptation [50, 51, 52]. The main advantage of this is that pseudo-labels can then use more sophisticated sampling techniques. For example, as semantic segmentation tasks often have a large class imbalance, there can be benefit found in oversampling less frequent classes. For ClassMix this could also have the additional benefit of allowing selection of which images to mix conditioned on the semantic maps predicted. Context could be better respected if images with similar semantics are combined. As by the previous example of road and sidewalk; images could be combined such that a sidewalk from one image is pasted appropriately with respect to the road of the other image. A simple heuristic could be that images where the predicted semantic maps are similar should primarily be combined. This issue could be especially important to address for other segmentation tasks not included in this thesis where semantic objects are unable to move independently, no longer invariant to their backgrounds.

5.3.3 Supervised Learning

For this thesis, we limited ourselves to two applications; semi-supervised learning and unsupervised domain adaptation. It would be interesting to investigate if ideas from this thesis could lead to performance gains in a purely supervised setting as well. Existing mixed sample data augmentation strategies were originally introduced for supervised learning [14, 79]. Though, in our opinion, it seems as if they are especially helpful for consistency based semi-supervised learning. The reasons for this were outside the scope of this thesis. We also note that strategies similar to ClassMix have been used to synthesize new data for supervised learning [59, 60, 61].

5.3.4 Active learning

Perturbing unlabeled images can be used in other settings than for enforcing consistency, for example to determine what samples to label in a setting called active learning [80]. As consistency regularization for semi-supervised learning forces the model to be invariant to perturbations, the images where the network fails to become invariant could be those where labels are most efficient for learning. This combination of active learning and semi-supervised learning has been explored in prior research [81, 82]. It could therefore be an interesting research direction to investigate whether similar solutions can be adapted to semantic segmentation, and in particular in combination with ClassMix, where inconsistent predictions of objects can be interpreted as those not recognised in different contexts or occlusions.

6
Conclusion

The need for data-efficient solutions is becoming increasingly important as the amount of available data grows and computational resources improve. This is especially true for semantic segmentation, where labels are expensive to procure. In this thesis, we have investigated semi-supervised learning and unsupervised domain adaptation, two approaches to training machine learning models with fewer labeled samples. We have found that using appropriate techniques, one can obtain results much higher than if only using labeled samples. In particular, we have introduced a novel mixed sample data augmentation, which we call ClassMix, that is based on the semantic maps of the images it is mixing. It selects some of the classes present in one image and transfers these to a second image, hence creating a mixed image that takes semantic information into consideration, forming a more realistic-looking sample than if mixing images with randomly generated masks. Another contribution of ours is the combination of entropy minimization and consistency regularization for semantic segmentation.

Using the ClassMix augmentation scheme, we achieve state-of-the-art results for semi-supervised semantic segmentation. Our results are higher than those of previous research with a margin of up to 3% for the Cityscapes dataset, and numbers are competitive for the Pascal VOC 2012 dataset.

We also apply a version of the ClassMix augmentation on unsupervised domain adaptation from the GTA5/SYNTHIA datasets to Cityscapes. Instead of mixing two unlabeled samples based on the network's predictions, we paste objects based on the ground truth segmentation from source domain data on top of target domain data, thereby injecting entropy into samples, as well as creating new, varied, samples. For this task we obtain state-of-the-art results for both datasets, with an increase from previous results with up to 2%.

Bibliography

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt 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.

[3] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *ArXiv e-prints*, mar 2016.

[4] 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.

[5] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[6] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *International Conference on Learning Representations*, number 6, 2018.

[7] Viktor Olsson, Wilhelm Tranheden, Juliano Pinto, and Lennart Svensson. Classmix: Segmentation-based data augmentation for semi-supervised learning. *arXiv preprint arXiv:2007.07936*, 2020.

[8] Wilhelm Tranheden, Viktor Olsson, Juliano Pinto, and Lennart Svensson. Dacs: Domain adaptation via cross-domain mixed sampling. *arXiv preprint arXiv:2007.08702*, 2020.

[9] Chen Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *International Conference on Learning Representations*, 2015.

[10] 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.

[11] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

[12] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised

learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059, 2019.

[13] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

[14] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031. IEEE.

[15] Geoff French, Samuli Laine, Timo Aila, and Michal Mackiewicz. Semi-supervised semantic segmentation needs strong, varied perturbations. In *29th British Machine Vision Conference, BMVC 2020*, 2019.

[16] Y Chen, Y Lin, M Yang, and J Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1791–1800, 2019.

[17] Qianyu Zhou, Zhengyang Feng, Guangliang Cheng, Xin Tan, Jianping Shi, and Lizhuang Ma. Uncertainty-aware consistency regularization for cross-domain semantic segmentation. *arXiv preprint arXiv:2004.08878*, 2020.

[18] Dong-Hyun Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[19] D. Randall Wilson and Tony R. Martinez. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(10):1429 – 1451, 2003.

[20] Boris Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4:1–17, 12 1964.

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[22] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015.

[23] Chen Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *International Conference on Learning Representations*, 2015.

[24] Wei Chih Hung, Yi Hsuan Tsai, Yan Ting Liou, Yen Yu Lin, and Ming Hsuan Yang. Adversarial learning for semi-supervised semantic segmentation. In *29th British Machine Vision Conference, BMVC 2018*, 2019.

[25] 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.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

[27] M. Holschneider, R. Kronland-Martinet, J. Morlet, and Ph. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform.

In Jean-Michel Combes, Alexander Grossmann, and Philippe Tchamitchian, editors, *Wavelets*, pages 286–297, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.

[28] George Papandreou, Iasonas Kokkinos, and Pierre-André Savalle. *Untangling Local and Global Deformations in Deep Convolutional Networks for Image Classification and Sliding Window Detection*. PhD thesis, Toyota Technological Institute at Chicago; Ecole Centrale Paris; Inria . . . , 2014.

[29] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 06 2014.

[31] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *ArXiv*, abs/1911.09785, 2019.

[32] Xiaomeng Li, Lequan Yu, Hao Chen, Chi-Wing Fu, and Pheng-Ann Heng. Semi-supervised skin lesion segmentation via transformation consistent self-ensembling model. *arXiv preprint arXiv:1808.03887*, 2018.

[33] Christian S. Perone and Julien Cohen-Adad. Deep semi-supervised segmentation with weight-averaged consistency targets. *Lecture Notes in Computer Science*, page 12–19, 2018.

[34] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, 57-64 (2005)*, 01 2005.

[35] Yassine Ouali, Céline Hudelot, and Myriam Tami. Semi-supervised semantic segmentation with cross-consistency training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12674–12684, 2020.

[36] Geoff French, Avital Oliver, and Tim Salimans. Milking cowmask for semi-supervised image classification. *arXiv preprint arXiv:2003.12022*, 2020.

[37] Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[38] Xiang Wang, Shaodi You, Xi Li, and Huimin Ma. Weakly-supervised semantic segmentation by iteratively mining common object features. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1354–1362, 2018.

[39] Mostafa S Ibrahim, Arash Vahdat, Mani Ranjbar, and William G Macready. Semi-supervised semantic image segmentation with self-correcting networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12715–12725, 2020.

[40] Karl Mason, Sadegh Vejdan, and Santiago Grijalva. An “on the fly” framework for efficiently generating synthetic big data sets. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 3379–3387. IEEE, 2019.

[41] Sergey Nikolenko. Synthetic data in deep learning. In *School-conference “Approximation and Data Analysis 2019”*, page 21, 2019.

[42] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017.

[43] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018.

[44] Zuxuan Wu, Xin Wang, Joseph E Gonzalez, Tom Goldstein, and Larry S Davis. Ace: adapting to changing environments for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2121–2130, 2019.

[45] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2100–2110, 2019.

[46] Yawei Luo, Ping Liu, Tao Guan, Junqing Yu, and Yi Yang. Significance-aware information bottleneck for domain adaptive semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6778–6787, 2019.

[47] Lei Zhang, Shanshan Wang, Guang-Bin Huang, Wangmeng Zuo, Jian Yang, and David Zhang. Manifold criterion guided transfer learning via intermediate domain generation. *IEEE Transactions on Neural Networks and Learning Systems*, 30(12):3759–3773, 2019.

[48] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative patch representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1456–1465, 2019.

[49] Shanshan Wang, Lei Zhang, Wangmeng Zuo, and Bob Zhang. Class-specific reconstruction transfer learning for visual recognition across domains. *IEEE Transactions on Image Processing*, PP:1–1, 11 2019.

[50] Yang Zou, Zhiding Yu, BVK Kumar, and Jinsong Wang. Domain adaptation for semantic segmentation via class-balanced self-training. *arXiv preprint arXiv:1810.07911*, 2018.

[51] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5982–5991, 2019.

[52] Zhedong Zheng and Yi Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *arXiv preprint arXiv:2003.03773*, 2020.

[53] Jongmok Kim, Jooyoung Jang, and Hyunwoo Park. Structured consistency loss for semi-supervised semantic segmentation. *ArXiv*, abs/2001.04647, 2020.

[54] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[55] Ethan Harris, Antonia Marcu, Matthew Painter, Mahesan Niranjan, Adam Prügel-Bennett, and Jonathon Hare. Understanding and enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047*, 2020.

[56] Liang-Chieh Chen, Raphael Gontijo Lopes, Bowen Cheng, Maxwell D Collins, Ekin D Cubuk, Barret Zoph, Hartwig Adam, and Jonathon Shlens. Semi-supervised learning in video sequences for urban scene segmentation. *arXiv preprint arXiv:2005.10266*, 2020.

[57] John S. Bridle, Anthony J. R. Heading, and David J. C. MacKay. Unsupervised classifiers, mutual information and 'phantom targets. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 1096–1101. Morgan-Kaufmann, 1992.

[58] A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[59] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1301–1310, 2017.

[60] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324, 2016.

[61] Georgios Georgakis, Arsalan Mousavian, Alexander C Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes. *arXiv preprint arXiv:1702.07836*, 2017.

[62] Zhengyang Feng, Qianyu Zhou, Guangliang Cheng, Xin Tan, Jianping Shi, and Lizhuang Ma. Semi-supervised semantic segmentation via dynamic self-training and class-balanced curriculum. *arXiv preprint arXiv:2004.08514*, 2020.

[63] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[64] Bharath Hariharan, Pablo Arbelaez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision (ICCV)*, 2011.

[65] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *European Conference on Computer Vision (ECCV)*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing, 2016.

[66] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016.

[67] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7472–7481, 2018.

[68] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2507–2516, 2019.

[69] Jihan Yang, Ruijia Xu, Ruiyu Li, Xiaojuan Qi, Xiaoyong Shen, Guanbin Li, and Liang Lin. An adversarial perturbation oriented domain adaptation approach for semantic segmentation. *arXiv preprint arXiv:1912.08954*, 2019.

[70] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2517–2526, 2019.

[71] Yiheng Zhang, Zhaofan Qiu, Ting Yao, Dong Liu, and Tao Mei. Fully convolutional adaptation networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6810–6818, 2018.

[72] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6936–6945, 2019.

[73] Jinyu Yang, Weizhi An, Chaochao Yan, Peilin Zhao, and Junzhou Huang. Context-aware domain adaptation in semantic segmentation. *arXiv preprint arXiv:2003.04010*, 2020.

[74] Zhedong Zheng and Yi Yang. Unsupervised scene adaptation with memory regularization in vivo. *arXiv preprint arXiv:1912.11164*, 2019.

[75] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[76] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[77] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery.

[78] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, pages 2535–2544, 2019.

[79] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[80] Dani Kiyasseh, Tingting Zhu, and David A Clifton. Alps: Active learning via perturbations. *arXiv preprint arXiv:2004.09557*, 2020.

[81] Mingfei Gao, Zizhao Zhang, Guo Yu, Sercan O Arik, Larry S Davis, and Tomas Pfister. Consistency-based semi-supervised active learning: Towards minimizing labeling cost. *arXiv preprint arXiv:1910.07153*, 2019.

[82] Keze Wang, Xiaopeng Yan, Dongyu Zhang, Lei Zhang, and Liang Lin. Towards human-machine cooperation: Self-supervised sample mining for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1605–1613, 2018.