# IMAGE-BASED SPATIAL CHANGE DETECTION USING DEEP LEARNING

EVANGELOS BOUSIAS ALEXAKIS

A DISSERTATION SUBMITTED TO

THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

**DOCTOR OF PHILOSOPHY**

GRADUATE PROGRAM IN EARTH AND SPACE SCIENCE

YORK UNIVERSITY

TORONTO, ONTARIO

September 2023

# Abstract

Image change detection is an invaluable tool in monitoring and understanding the built and physical environments and supporting decision-making. Many recent research approaches for automatic change detection have been based on Deep Learning (DL) techniques and especially on variations of Convolutional Neural Network (CNN) architectures. CNNs have achieved notable success thanks to their great representational capacity, straightforward training, and state-of-the-art performance in visual tasks. Nevertheless, CNNs, like most DL approaches, still face limitations relating to their reliance on extensive labelled datasets, the localization accuracy and detail of the predicted outputs, and the notable model performance degradation when the target data have different characteristics from the training data.

This research contributes to the development and evaluation of novel DL methods and algorithms for automated image-based spatial change detection. It investigates novel architectures for enhanced model performance and ways to address the limited availability of labelled data and improve model generalizability. Two main approaches are investigated: (i) a direct approach, where both instances are fed into the DL model that is trained to output the prediction of the change map, and (ii) a post-classification approach, where change detection is performed based on each epoch's semantic segmentation predictions. In both cases, novel, enhanced CNN architectures have been proposed that leverage the semantic information of objects' boundaries to improve the accuracy of the model's predictions. Furthermore, training frameworks inspired by self-ensembling and the Mean Teacher method were developed for semi-supervised learning and domain adaptation, attenuating the models' reliance on large, labelled training datasets and improving their generalization performance. We evaluated the proposed approaches on multiple datasets using the precision, recall, F1 score, and Intersection over Union (IoU) metrics. The results indicate that both boundary-enhanced approaches lead to consistent, albeit marginal, benefits between 1 and 2 %. Notably, the proposed semi-supervised training framework for direct CD approximately matches the performance of the fully supervised approach while using only 20% of the available training labels. In domain adaptation, our post-classification approach significantly outperforms typical supervised methods, with the most

notable gains in recall rate (>22%) and IoU (12.6%). These findings highlight the effectiveness of our techniques.

*To Artemis, my partner, who has walked every step of this journey with me.*

*Στην Άρτεμη.*

# Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor, Professor Costas Armenakis, for his support, guidance, and invaluable insights throughout this research work. His expertise and mentorship have been instrumental in shaping the direction of this study.

I am also thankful to the members of my dissertation committee, Professor Gunho Sohn and Professor Jinjun Shan, for their questions, suggestions, and constructive feedback, which have significantly enriched the quality of this work. Furthermore, I would like to thank the members of the examination committee, Professors Baoxin Hu, Manos Papagelis, and Konstantinos Karantzalos for their time reviewing this dissertation and for their constructive comments.

I extend my appreciation to my colleagues and fellow researchers who provided stimulating discussions and a supportive academic environment. Your perspectives and insights have been crucial in refining the ideas presented in this thesis.

This endeavor would not have been possible without Artemis, my partner, whose unwavering support, love, and encouragement sustained me through the challenges and triumphs of this academic endeavor. Special thanks to my parents, George and Labrini, my aunt, Georgia, and my sister, Katerina, for their support, love, and belief in my abilities.

I would also like to thank all my close friends in Greece with whom I had to part for all these years. Especially my mates Panos and Yiagos, you know you are family to me even though I couldn't be there for your weddings!

Big thanks to Myrto for her friendship, support, and encouragement. She made our stay in Canada a lot more fun, and I consider ourselves very lucky to have made such a good friend! I would also like to thank Kyria Roula and Prof. Pagiatakis for treating us like family, and our friends Katerina, Miguel, Christina, Athina, Eleni, Sid and Shurabi for their friendship and support.

Lastly, I would like to acknowledge the financial support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), York University, and the Carswell Family Foundation, which made this research possible.

# Contents

# List of Tables

# List of Figures

# List of Algorithms

# 1

# Introduction

For more than half a century, the development of reliable Change Detection (CD) pipelines from multi-temporal aerial and satellite images has been an active area of research in the remote sensing community. This ongoing interest is mainly due to the fact that the development of a robust change detection framework can be an invaluable tool for mapping, monitoring, and understanding the spatio-temporal transformations of the built and natural environments. Spatial changes can reveal impacts, patterns, and relationships of the anthropogenic and natural activities on the Earth's environments. Change detection has a wide range of applications including, but not limited to, map updating, land cover monitoring, urban monitoring, forest or vegetation change (forest mortality, forest fire, defoliation, and damage assessment), drought and flood monitoring and monitoring changes in coastal zones environments[1].

According to Singh (1989), change detection refers to the process of identifying differences in the state of an object or a phenomenon by observing it at different times. In this work by "change" we refer to spatial land-cover changes. Although the process might sound simple, there is a semantic aspect that can be difficult to automate. Depending on the type of change that we are trying to detect, we need to focus on specific kinds of differences and ignore the apparent ones that are not of interest. For example, when trying to detect land cover changes, the algorithm must be able to ignore changes that are due to different weather or seasonal conditions between the two instances, such as snow or clouds (apparent changes). Thus, the change detection algorithm must be able to distinguish the semantically important variations (actual changes) from the total detected variations, including those irrelevant to the task. There is also a

[1] An extensive, albeit not recent, summary of research works relating to the different aspects of change detection can be found in (Lu et al., 2004).

need to address other common change detection challenges, such as differences in illumination, contrast, scale, sensor viewing geometry, resolution, occlusions, shadows, and image misregistration errors.

A plethora of change detection methods have been developed to address these issues. Tewkesbury et al. (2015) point out the difficulty of navigating the extensive and ever-expanding research literature on change detection and propose a taxonomy of the approaches according to (i) the unit of analysis and (ii) the comparison method. When using the unit of analysis as the classification criterion, CD methods can be classified, in the broadest sense[2], into pixel-based and object-based methods, with the object-based methods using pixel groupings, such as the segments extracted by image segmentation algorithms commonly applied in Object-Based Image Analysis (OBIA). The comparison methods are organized into layer arithmetic methods, post-classification CD, direct classification CD, transformation methods (e.g., using principal component analysis (PCA) and multivariate alteration detection (MAD)), change vector analysis (CVA), and hybrid methods.

Modern change detection methods rely heavily on machine learning and deep learning algorithms. Traditional machine learning methods, such as PCA and Support Vector Machines (SVM), have been extensively used in change detection applications (Bai et al., 2022; Lebedev et al., 2018). Nevertheless, these methods rely on hand-crafted sets of features carefully designed for a specific dataset/task and may struggle with complex, non-linear changes or require extensive data preprocessing. Deep learning approaches, on the other hand, can be used to extract high-level, abstract features, which are very hard to model with traditional representation learning algorithms, directly from the input data by recursively introducing representations that are expressed in terms of other, simpler representations (I. Goodfellow et al., 2016, Chapter 1). By learning directly from data using Convolutional Neural Networks (CNNs) – or Artificial Neural Networks (ANNs) in the more general case – the models can extract intricate patterns within the data and detect subtle changes in a more adaptive and nuanced manner, thus eliminating the need for explicit feature engineering.

[2] A more detailed taxonomy comprising six classes of methods is presented in (Tewkesbury et al., 2015) but is out of the scope of this introduction.

CNNs have been successfully applied in the detection, segmentation, and recognition of regions, objects, and instances from many different kinds of imagery and have produced state-of-the-art results in most vision-related tasks. CNNs are specialized to work with data having a grid-structured topology, such as images, and they are easier to train and generalize better than networks with fully connected layers (LeCun et al., 2015). The stacks of convolutional and pooling layers of a CNN, which are inspired by classic notions in visual neuroscience, allow them to learn useful context information from images by taking advantage of the hierarchical structure of an image's features.

Lately, the most successful approaches to automatic change detection from satellite images rely on CNN using an encoder-decoder architecture that directly produces a binary map of changed and unchanged regions given an initial image pair (Daudt et al., 2018; D. Peng et al., 2019; Caye Daudt et al., 2019; Papadomanolaki et al., 2020). This labeling process, where each pixel in an image is classified into one of a given number of classes, is called semantic segmentation in deep learning literature. The first successful deep CNN architecture to perform end-to-end semantic segmentation, meaning that the network produced a final prediction map for each object class without the need for any post-processing steps, was the Fully Convolutional Networks (FCN) (Long et al., 2015). Although FCN excelled at many semantic segmentation benchmarks, they suffered from blurry outputs due to the repeated application of down-sampling operators (either through max-pooling or striding) performed at consecutive layers of the encoder network, leading to reduced spatial resolution features. Another challenge pertained to the existence of objects at various scales and the fact that the receptive field of FCN grew slowly with respect to the depth of the network, thus limiting the ability of the network to fully model complex long-range relationships between image regions (X. Li et al., 2020). Many approaches have been proposed since then that aim to address these issues, such as UNet (Ronneberger et al., 2015), UNet++ (Z. Zhou et al., 2018), and Feature Pyramid Networks (FPN) (Lin et al., 2017) architectures that introduce skip connections to improve the spatial accuracy of the model representations, and SegNet (Badrinarayanan et al., 2017) which performs the non-linear upsampling in the network's decoder layers using pooling positional indices that have been estimated in the max-pooling step of the corresponding encoding layers. Also, the multiple versions of the DeepLab architectures (L.-C. Chen et al., 2017, 2018) improve the spatial accuracy and increase the receptive fields of convolutional layers by introducing dilated, also

known as atrous (à trous), convolutions and the Atrous Spatial Pyramid Pooling (ASPP) layer to capture objects and image context at multiple scales. End-to-end approaches are both straightforward and very fast when producing the final prediction maps (also known as heat maps), primarily when run on a modern GPU (Long et al., 2015; Ronneberger et al., 2015).

UNet is an encoder-decoder architecture consisting of a contractive path and a symmetrical expanding path. The contracting path is built on a sequence of convolutional and max-pooling layers that gradually reduce the spatial resolution of the input and capture contextual information. In contrast, the expanding path consists of sequential convolutional and upsampling layers that increase the spatial resolution back to the resolution of the original input image. Each max pooling step degrades the spatial accuracy of the output, which cannot be retrieved solely by using the upsampling step. Therefore, in order to take advantage of both the contextually rich semantic information of the coarser lower layers and the spatially accurate activations of the fine-grained higher layers, multiple skip connections have been introduced between contrastive and expanding blocks that share a matching resolution. Z. Zhou et al. (2018) suggest that there is a semantic gap between the respective contractive and expanding blocks that negatively affects the performance of the UNet architecture. In order to bridge this semantic gap, they developed the Nested UNet, also known as UNet++ architecture, by introducing multiple intermediate convolutional blocks at various levels of the architecture and increasing the number of skip connections between the different convolutional blocks. Experimental results obtained from medical imagery in the original paper, as well as from change detection imagery (D. Peng et al., 2019; Bousias Alexakis & Armenakis, 2020), suggest that UNet++ performs semantic segmentation tasks better than the original UNet architecture.

## 1.1. Deep Learning Change Detection Open Challenges

Despite the excellent performance and many advantages of CNN-based architectures compared to traditional machine learning approaches, there are still open challenges concerning the development and training of CNN-based change detection algorithms that need to be addressed.

First, the model's prediction quality directly relates to the annotated training data quality. As in all learn-by-example approaches, the model is only as good as the data on which it is trained. Hence, there is a need for high-quality annotation data for training and evaluating the models

(Foody et al., 2016). The change detection training data are usually obtained from time-consuming, labour-intensive, and costly processes such as human interpretation of remotely sensed datasets with the help of semi-automated CD pipelines. Therefore, the cost of acquiring and annotating data for training and evaluating such models is a significant factor in remote sensing applications since CNNs require considerable amounts of data in order to generalize effectively on high variability datasets. Thus, there is a need for methods that can help decrease the amount of labelled data needed to successfully train a CNN-based encoder-decoder architecture and simultaneously effectively use the substantial amount of available unlabelled data.

In addition, even when enough labelled training data are available, current state-of-the-art deep learning models exhibit significant drops in performance when faced with new test data (new domains) that have different characteristics (different distributions) from the training dataset (Bengio et al., 2021; B. Sun et al., 2016). On the contrary, humans can quickly – and mostly unconsciously – adapt to such changes (e.g., images collected using a different sensor/camera).

Finally, besides issues relating to the quantity and quality of the training and evaluation datasets and the out-of-distribution generalization capabilities of CNNs, most successful CNN approaches still face challenges with respect to semantic segmentation localization accuracy and detail, especially when it comes to the detection of small objects and the accurate labelling of pixels near the borders of changed regions (Marmanis et al., 2018; X. Li et al., 2020; Caye Daudt et al., 2018; D. Peng et al., 2019; Papadomanolaki et al., 2020).

## 1.2. Objectives - Contributions

The primary research directions of this work are shaped by the need to address the key challenges facing deep learning solutions for change detection, which were discussed in the previous section. The objectives of this dissertation are:

- To assess the change detection performance of existing CNN architectures and investigate the effects of various model and training design choices on the model's predictions' quality.

- To explore the benefits of incorporating boundary information into CNN architectures as a way to improve the detail and localization accuracy of the produced change maps.
- To attenuate CNN's reliance on extensive labelled training data by leveraging the information provided by readily available unlabelled imagery.
- To improve the CNN performance on test data originating in a domain other than the one used for training, thus enabling the utilization of readily available datasets for training the models.

### 1.2.1 Research Contributions

This research work contributes to the development of novel deep learning approaches and algorithms for automated image-based spatial change detection and the development, evaluation, and comparison of automatic, robust, and highly generalized change detection pipelines for multi-sensor multi-temporal imagery. We identified and investigated two main supervised deep learning approaches for change detection:

- A direct approach (DA), where both instances are fed into the DL model that is trained to output the prediction of the change map, and
- A post-classification or semantic-based approach where the deep learning networks are first trained to detect objects of interest through either semantic or instance segmentation separately for each image epoch. Then, the per epoch predictions are compared in order to detect the changes.

The direct change detection approach is inspired by traditional pixel-based change detection methods, is straightforward to implement and results in high model performance as existing CNN-based approaches that follow a direct change detection formulation have achieved state-of-the-art results (Daudt et al., 2018; Peng et al., 2019). Producing semantic segmentation maps for each epoch and comparing them to retrieve the changes between image pairs is another traditional approach (Singh, 1989) that allows the identification of the type of change besides the detection of changed areas in an image pair. According to Tewkesbury et al. (2015), the direct comparison of land cover maps produced from satellite images is "one of the most established and widely used change detection methods."

The contributions of this dissertation pertaining to the direct change detection approach comprise the:

- Adjustment of U-shaped state-of-the-art models to the change detection task and evaluation of their performance using different training techniques.
- Development and evaluation of a novel U-shaped encoder-decoder architecture that leverages boundary semantic Information to improve segmentation accuracy.
- Development and evaluation of a data augmentation technique that increases the model's robustness to misregistration noise.
- Development and evaluation of a semi-supervised training framework based on the Mean Teacher paradigm to increase the quality of segmentation in the absence of an adequate number of labelled data for supervised training.

The proposed post-classification change detection approach introduces a domain adaptation training framework inspired by the temporal-ensembling and Mean Teacher methods (Tarvainen & Valpola, 2017; S. Li et al., 2021; French et al., 2019). Our contribution addresses the lack of labelled training data for the direct change detection approach by shifting the training task of the deep learning models from change detection to land use/land cover pixel-wise classification, thus utilizing existing large datasets that are readily available.

The original research contributions of the proposed post-classification change detection pipeline can be summarized as follows:

- The development and evaluation of a ResUNet-based encoder-decoder CNN for building semantic segmentation, which leverages the buildings' boundaries information via body and edge decoupling in order to enhance the segmentation performance.
- The development and evaluation of a domain adaptation training framework inspired by the Mean Teacher (MT) training paradigm, which aims to bridge the models' performance gap between the training (or source) and the target datasets (or domains).
- The development and evaluation of a filtering process that removes prediction errors due to small misclassification and misregistration errors in the building prediction masks. This process helps to refine the change detection results, improving the overall accuracy of the change detection pipeline.

## 1.3. Thesis Outline

An overview of the dissertation's organization is presented in Figure 1.1 with chapters 3, 4 and 5 dedicated to the direct change detection approach and chapters 6 and 7 dedicated to the development and assessment of the post-classification change detection pipeline.

Chapter 2 provides a literature review of relevant research works pertaining to:

- Change detection applications employing deep learning techniques with an emphasis on CNN-based approaches.
- Edge and boundary detection CNN architectures and edge-enhanced CNN-based encoder-decoder approaches.
- Semi-supervised Learning and domain adaptation techniques that aim to alleviate the models' reliance on extensive datasets of labelled samples for training and improve the generalization performance on new datasets with different characteristics.

The chapter ends with a brief overview of the datasets used throughout this research.

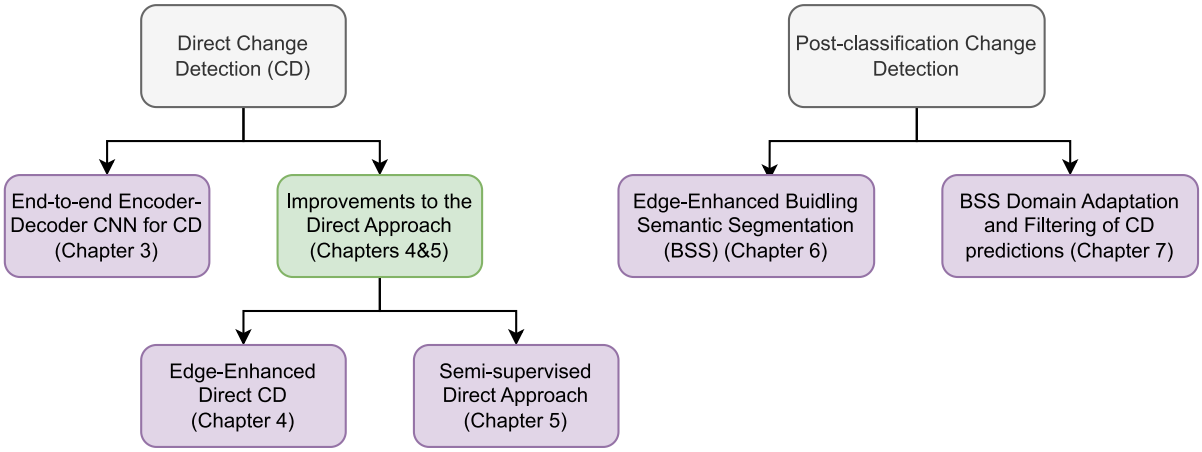



*Figure 1.1: Overview of the chapter organization.*

In Chapter 3, we introduce an early fusion CNN-based method that takes as input images of the same region collected at two different time epochs and directly predicts the change map. We train and evaluate two CNN architectures, UNet and UNet++, using Very High-Resolution (VHR) satellite images on the change detection task. We also examine the effects of different loss functions, data augmentation, and deep supervision techniques on the models' predicted change maps. Chapter 3 heavily reflects the content published in (Bousias Alexakis & Armenakis, 2020).

Chapter 4 presents a new edge-enhanced CNN architecture for change detection that incorporates semantically informed edges produced via the Dense Extreme Inception Network (DexiNeD) (Soria et al., 2020) into U-shaped encoder-decoder models and a simple yet effective approach for implicitly modeling the misregistration errors between the two instances of each image pair. Extensive experiments on a CD dataset consisting of very high-resolution RGB image pairs and their corresponding change masks (Lebedev et al., 2018) indicate that both proposed approaches lead to increased model performance on the precision, recall, F1 score, and Intersection over Union metrics. Chapter 4 heavily reflects the content published in (Bousias Alexakis & Armenakis, 2021b).

Chapter 5 describes a semi-supervised learning approach for change detection based on the Mean Teacher framework that aims to reduce the need for labelled training samples by synergistically utilizing unlabelled samples – which are easier and cheaper to acquire – together with labelled samples to train the models. The training approach is tested on two different change detection datasets, with the results on the more extensive and more reliable LEVIR-CD (H. Chen & Shi, 2020) dataset indicating significant improvements in the metric values when our semi-supervised training scheme is applied. Chapter 5 – excluding the experiments and analysis on the LEVIR-CD dataset – heavily reflects the content published in (Bousias Alexakis & Armenakis, 2021a).

Chapters 6 and 7 are dedicated to our proposed post-classification change detection approach. Chapter 6 focuses on the development of a U-shaped architecture – based on a ResNet50 encoder – designed to leverage the semantic information of the objects' boundaries so as to improve the quality and detail of the produced semantic segmentation predictions. The proposed approach is evaluated and

Chapter 6 – excluding the experiments and analysis on the LEVIR-CD dataset – heavily reflects the content published in (Bousias Alexakis & Armenakis, 2022).

Chapter 7 describes the remaining two components of the CD pipeline: a domain adaptation training framework inspired by the Mean Teacher self-ensembling technique and a filtering step designed to remove false positive changes that are due to partial building misclassifications and to misregistration errors between the instances of the image pair. The benefits of the proposed domain adaptation training framework are extensively investigated for multiple domain

adaptation scenarios between different building semantic segmentation datasets and for the primary post-classification change detection task on the LEVIR-CD building change detection dataset.

Finally, Chapter 8 discusses the conclusions of this research and suggests future work directions.

# 2

# Relevant Literature Review

Aerial and satellite images have been used extensively for the detection of spatial and semantic changes of the built and natural environments. Recent developments in the deep learning methods and techniques, and especially the use of CNN architectures, are significantly contributing to the use of machine learning approaches for change detection from high resolution images. In this chapter we present and discuss a) an overview of notable previous work on image-based change detection using deep learning algorithms, b) a review of methods aimed at improving the performance of deep CNN architectures for the semantic segmentation task, and c) a summary of prominent works in the fields of semi-supervised learning and domain adaptation.

## 2.1. DL - based methods for image-based Change Detection

This section reviews the state-of-the-art research on change detection based on deep learning. The researched Deep learning methods are organized into methods utilizing CNN architectures, which comprise the patch (scene) and semantic segmentation-based categories, transformers for semantic segmentation, and unsupervised change detection. Semi-supervised approaches are discussed in Section 2.3.

Since the publication of AlexNet in 2012 (Krizhevsky et al., 2012) and the first-place win, by a large margin, in the 2012 ImageNet Large Scale Visual Recognition Challenge, CNNs have become a popular approach for vision-related tasks and have been deployed in many real-world applications. A similar trend has been noted in the remote sensing research community, with many recent papers exploring the potential of CNN for land cover classification and change detection.

We focussed on research that applies CNN-based approaches for Change Detection applications. For a more general overview of recent Change Detection applications not limited to CNN-based

approaches, D. Peng et al. (2019) have provided a thorough summary. There are two main approaches for applying a CNN architecture to a pixel-based labelling task: a) using a CNN that takes as input patches of the initial image and produces a label for the central pixel of the patch, and b) using a CNN that directly performs semantic segmentation for the entire image.

### 2.1.1 Image patch-based approaches

Patch-based approaches run in a sliding window manner across the image and produce separate labels for each image pixel. These approaches can overcome a lack of training data, as a single training image pair can provide many training patches. On the other hand, patch-based methods have a generally smaller receptive field, and thus, they cannot take into account contextual information for the entirety of an image. Since they are applied in a sliding window manner, they are both very slow in terms of inference time and inefficient when compared to encoder-decoder architectures, as the same regions need to be visited multiple times.

There are examples of patch-based CNN approaches for change detection applications using different kinds of imagery (SAR, multispectral, hyperspectral, RGB). Wiratama et al. (2018) developed a dual dense CNN architecture for change detection in SAR images, where preceding feature maps are connected to all subsequent layers. Their model consists of two independent convolutional subnetworks: one for each instance of the image pair. The network works on 40 × 40 pixels patches of the image pair and is trained using a contrastive loss function. Daudt et al. (2018) detect changes in Sentinel-2 multitemporal instances by applying the CNN architectures described by Zagoruyko & Komodakis (2015). Zagoruyko & Komodakis (2015) experimented with the use of Siamese, pseudo-Siamese and 2-channel networks for estimating the similarity between an image pair. The pseudo-Siamese networks consist of two identical networks similar to the regular Siamese networks, but each subnetwork has its own trainable parameters (there is no parameter sharing). The 2-channel networks are typical CNN architectures that receive a multi-channel image, with each channel being a different instance, instead of having a single image as input. The method is easily extensible and can include more than one channel per image. Daudt et al. (2018) train multiple networks with a varying number of input channels per image, ranging from a common RGB image to an image comprising all 13 Sentinel 2 channels. On a similar note, Zhang & Lu (2019) worked on a Siamese model for change detection in multispectral image pairs. Image patches are fed into a Siamese network consisting of two

identical CNNs that share the same weights. The outputs of each CNN are unravelled onto a 1-dimensional feature vector, and the results are fused. The fused results are then passed through a Neural Network consisting of two hidden layers that classify the central pixel of the patch as changed or unchanged.

A CNN framework can be successfully applied to various types of datasets (RGB, multispectral, hyperspectral, PolSAR), provided that it is trained on that same type of data. Seydi et al. (2020) propose an end-to-end Multi-dimensional CNN framework for change detection. The model consists of 3 parallel channels: the first two channels are used to extract deep features from the bi-temporal images (one channel per image) and the third channel is used to create deep features of changes by stacking and differencing the deep features of the other two channels.

On another patch-based CD approach that highlights the potential of transfer learning, Cao et al. (2019) evaluated three different popular CNN architectures (AlexNet, GoogLeNet and VGGNet) on Land Use (LU) classification of images captured at multiple different epochs between 2004 and 2017. The authors also applied transfer learning, using the original weights of a model trained on a general-purpose deep learning dataset such as ImageNet. They concluded that using a pre-trained GoogleNet model as a feature extractor and having the final convolutional layer replaced by a Support Vector Machine (SVM) classifier led to higher classification accuracies (up to 98%). Finally, they used the multitemporal LU classification results to perform a land cover change analysis.

(Ru et al., 2021) develop a patch-based approach for simultaneous image classification and land use change detection – which in the paper is called scene-level change detection and is contrasted with pixel-level change detection. The proposed network, named CorrFusionNet, was designed to take advantage of the temporal correlation between the bi-temporal feature representations. The network first encodes each instance of the bi-temporal image pair and then projects the extracted feature representations into a lower dimensional space using multiple fully connected layers. The produced feature maps are then fused by the newly introduced CorrFusion module, which is based on Soft Deep Canonical Correlation Analysis (X. Chang et al., 2018).

### 2.1.2 Semantic Segmentation Encoder-Decoder-based Approaches

Over the past few years, multiple research efforts formulated the Change Detection problem as an end-to-end semantic segmentation task that can be addressed by some form of encoder-decoder CNN network architectures. The encoder learns discriminative feature representations at multiple scale levels from the low-resolution, hidden layers. At the same time, the decoder gradually reprojects the features to the original image resolution and produces pixel-wise classification predictions.

Caye Daudt et al. (2018) have used and compared multiple variations of Fully Convolutional Networks (FCNs) for change detection applications for bi-temporal satellite imagery. The models included a shortened version of the UNet Architecture and a hybrid Siamese encoder-decoder architecture that consisted of two identical contracting paths (encoders) and a single expanding path (decoder). Regarding the hybrid model, two different ways to pass the information from the contracting blocks to the corresponding expanding blocks were examined: either directly passing the output feature maps from both contractive blocks to the corresponding expanding blocks through a skip connection or fusing the two outputs (find their difference) first before passing that difference to the same level up-sampling block. In a following work, Caye Daudt et al. (2019) introduced a very high-resolution semantic change detection dataset that comprised 291 RGB image pairs accompanied by the corresponding pixel-wise change information and land cover information. They also proposed a multi-task framework that consists of multiple UNet architectures that, when given an image pair, can simultaneously predict change maps as well as provide a pixel-wise landcover labelling for each of the two image instances.

Another encoder-decoder architecture based on the Feature Pyramid Network (FPN) (Lin et al., 2017) and UNet was used by C. Zhang et al. (2019) in a change detection application. The common convolutional layers at each scale level of the network's encoder have been replaced by multi-rate atrous convolutions in order to increase the receptive field of each contractive block. The network is used to perform land cover pixel-wise classification on an input image and the output is then compared to a base map to produce the change map.

Su et al. (2020) performed building change detection with a UNet architecture. They input two orthoimages collected at different times into the network and experimented with additional

inputs that included height differences between the Digital Surface Models (DSMs) corresponding to each instance, and a map of the region referring to the initial image period. The experimental results suggest that both the DSM information and the map increase the performance of the model on the test set.

Papadomanolaki et al. (2020) integrated Long Short-Term Memory (LSTM) blocks into each contractive block of a UNet architecture in order to capture the temporal dynamics of features at different spatial resolutions. The proposed framework is modular, allowing the processing of more than two instances at the same time, while using an extra up-sampling subnetwork to produce an additional building semantic segmentation layer. Experimental results suggest that the proposed architecture outperforms the models used by Caye Daudt et al. (2018) on the F1score metric by over 2%.

Lebedev et al. (2018) used a Conditional Generative Adversarial Network (C-GAN) (Isola et al., 2017) to predict changes given a bitemporal image pair. The basic format of a GAN consists of two subnetworks: a generator G and a discriminator M that compete and learn the required representation (I. Goodfellow et al., 2020). The conditional GAN learns the mapping from an observed image and a random noise vector to produce a new image. In this case, the generator takes as input both instances of the image pair and tries to predict the change map that will trick the discriminator into classifying it as an original change map, while the discriminator takes as input both instances of the image pair, the ground truth changes, and the predicted changes and classifies the predicted changes produced by the generator as original or artificial. Theoretically, the training process ends when the discriminator cannot distinguish the original from the artificial change map (both classes have a probability of 0.5). The trained generator achieved a relatively high rate of prediction (ranging from 0.72 to 0.91) and recall rates (ranging from 0.65 to 0.87) when detecting changed objects between satellite multitemporal image pairs.

D. Peng et al. (2019) used the dataset created by Lebedev et al. (2018) to train a UNet++ network, which they later compared to other recent Deep Learning approaches for CD. Following training suggestions introduced in the original UNet++ paper (Z. Zhou et al., 2018) they showed that the use of a deep supervision training scheme and a combination of the binary cross entropy loss with the Dice coefficient as the loss function results in a better network performance. The comparison to other recent CNN methods for change detection concludes that

the UNet++ architecture trained using deep supervision outperforms other state-of-the-art approaches.

Cheng et al. (2022) propose a new deep learning architecture for remote sensing change detection applications named ISNet that was designed with a focus on feature refinement. The architecture uses a ResUNet encoder as a backbone and introduces multiple spatial and channel attention modules to better model the spatial and channel-wise relationships between the model's features as well as margin maximization modules, which were designed to better separate the features belonging to changed and unchanged regions.

Park et al. (2022) developed a dual task approach for change detection which simultaneously estimates the optical flow between the image pair though a "correspondence map decoder" and the corresponding change map via a "mis-correspondence (change) map decoder". They use a two stream feature pyramid architecture (Truong et al., 2020) that allows estimating the correspondence and mis-correspondence at multiple resolutions. To address the lack of labelled training data they propose an approach to create artificially synthesized training data using random image warping (in a way similar to (Melekhov et al., 2019)) and the cut-paste method (Dwibedi et al., 2017). Through extensive experiments they show that their architecture produces state of the art results on street view change detection datasets and can effectively handle image pairs with strong co-registration errors and/or captured from different camera viewpoints.

### 2.1.3 Deep Learning Architectures for Change Detection based on Transformers

The great success of multi-headed self-attention layers for natural language processing applications introduced in the Transformers architecture (Vaswani et al., 2017) has led to the development of many models whose aim was to leverage self-attention for computer vision tasks either by integrating attention layers into CNN-based architectures (Carion et al., 2020; Fu et al., 2019) or by developing pure transformer based models (Dosovitskiy et al., 2021; Z. Liu et al., 2021). During the past couple of years transformer-based approaches have also been adopted for change detection applications on remote sensing imagery.

#### 2.1.3.1 Integration of CNN and transformer architectures

H. Chen & Shi (2020) introduced STANet, a spatial-temporal attention neural network that predicts binary change maps from bitemporal remote sensing satellite imagery. The model follows a Siamese architecture that encodes each image instance separately using a ResNet18 encoder and feeds the embedded feature maps to a Pyramid spatial-temporal Attention Module (PAM). Drawing inspiration from the pyramid structure of the PSPNet (H. Zhao et al., 2017) the PAM module creates multiple scale feature maps from the input embeddings, which are processed individually by a multi-head self-attention module (Vaswani et al., 2017) and then aggregated into a tensor of the same shape as the one of the input feature maps. In addition, the authors proposed a weighted variation of the contrastive loss function, named batch-balanced contrastive loss, that uses the batch statistics to reweight the distances of the changed/unchanged pixels in the batch in order to address the class imbalance between changed and unchanged image regions.

Chen et al. (2021) incorporated a Dual Attention module (Fu et al., 2019), which can better model long-range channel and spatial relationships among features, to a Siamese architecture to improve the model's change detection performance. They also developed a weighted contrastive loss function to better address the imbalance in the frequency of changed and unchanged pixels in remote sensing datasets.

Another research work for remote sensing change detection that tries to enhance a CNN encoder-decoder architecture by incorporating into it attention layers is that of D. Peng et al. (2021), who replace all the up-sampling convolutional blocks of a UNet++ architecture by Up-Sampling Attention (UA) blocks. Each UA block comprises both channel and spatial attention layers to better model long-range relationships between different regions and channels of the input feature maps.

#### 2.1.3.2 Pure Transformer Methods

Zhang et al. (2022) introduced SwinSUNet: a pure transformer network for remote sensing image change detection that combines the structure of a Siamese encoder-decoder U-shaped architecture (such as the one used by Caye Daudt et al. (2018)) with the Swin transformer multi-head self-attention layers (Z. Liu et al., 2021). A similar approach that also relies on Swin

transformer multi-head self-attention layers and a Siamese encoder-decoder architecture for change detection from satellite images was introduced by Yan et al. (2023).

### 2.1.4 Unsupervised Change Detection using Deep Learning

In order to address the lack of training data, many unsupervised or semi-supervised approaches based on Neural Networks (NN) or CNN have been recently proposed. Some of them make use of autoencoders to automatically extract features from the image pairs and then apply complex algorithms like the Chan-Vese algorithm (X. Zhang et al., 2019) or a stacked mapping network and a clustering algorithm like fuzzy c-means (FCM) (L. Su et al., 2017). In the latter, the unsupervised method is mainly based on models which learn feature representations from images. A stacked denoising autoencoder is applied to two images for feature extraction. Then mapping functions are generated by a stacked mapping network to form relationships between the features of each class. The change detection is performed by comparing the features and, in the end, applying a clustering algorithm. More unsupervised approaches for CD are cited by Khelifi & Mignotte (2020), who provide a comprehensive review and meta-analysis of deep learning change detection methods for remote sensing images, but in most cases the proposed methods do not make end-to-end predictions and only incorporate the Deep Neural Network as a feature extractor in the CD pipeline. Another recent review paper on change detection approaches based on deep learning techniques for various types of change detection datasets (e.g. SAR, multispectral, hyperspectral, Very High Resolution RGB) is the one by (Shafique et al., 2022).

In the following sections, we focus on research works not confined to the field of change detection that explore ways to improve the semantic segmentation accuracy of CNN-based architectures with an emphasis on exploiting the semantic information from objects' boundaries. We will reattend to remote sensing and change detection approaches in the last section of this chapter when discussing semi-supervised learning and domain adaptation techniques.

## 2.2. Improving Model's Performance using Boundary Information

Methods and approaches for improving the performance of CNN for semantic segmentation were also investigated. These methods consider the edges and boundaries of image segments, such as

buildings, where an edge defines an abrupt variation in brightness or colour, while a boundary is a contour and indicates a change from one object to another.

### 2.2.1 Edge and Boundary Detection using CNNs

There are quite a few recent CNN-based approaches to edge and boundary detection. The introduction of a few prominent state-of-the-art approaches for semantic edge detection based on CNN architectures is deemed beneficial as chapters 4 and 6 introduce methods that aim to improve the performance of semantic segmentation networks for change and building detection by utilizing semantic boundary information.

An established approach that has been widely used in state-of-the-art research is the Holistically-Nested Edge Detection (HED) algorithm (Xie & Tu, 2015) that was developed to address, in a holistic manner, image training and prediction using multi-scale and multi-level feature learning. The term holistic is used to refer to an end-to-end process, meaning that the training and inference are performed in an image-to-image fashion. Nested refers to the nested multi-scale feature learning process, which is inspired by deeply supervised nets (Lee et al., 2015). The initial input image passes from sequential convolutional and max-pooling layers with gradually increasing strides and receptive fields so that shallow (and higher resolution) layers deal with more local image patterns while the deeper (and spatially coarser) layers can capture the object-level information. The network has five side outputs corresponding to edge predictions for the five different main scales of the network, with a class-balanced cross-entropy loss function being computed for each output/scale. There is also a learnable weighted-fusion layer that leads to a second loss function, a cross-entropy loss between the fused predictions and ground truth label maps. The complete objective function to be minimized consists of the sum of the loss functions of each output/scale plus the (overall) fused cross-entropy loss component.

Another recent approach is the Bi-Directional Cascade Network model (BDCN) (J. He et al., 2019). The main idea is similar to HED: the network consists of five main consecutive blocks that are based on the first five VGG16 blocks (Simonyan & Zisserman, 2015) but are enhanced using the proposed Scale Enhancement Module (SEM). SEM consists of multiple parallel convolutions with different dilation rates and serves the purpose of rescaling the output of each block to the original dimensions of the input image. The other main contribution of the BDCN is

the use of multiple and different loss functions for the predictions produced by each block. Two different loss functions are computed for each block, one considering the predictions for scales equal or higher to the block's scale, while the second one considers only the scales that are equal or lower to the block's scale. Like HED, they also include a layer that learns to fuse the results of the aforementioned loss functions, and, as a final objective function, they use the sum of the two loss functions produced for each block plus the computed fused loss. In this way, the network learns representations that are specific to the block's scale, unlike HED and other similar approaches. The latter uses the same ground truth mask for determining the predictions of different scales, a method that is not optimal as the edges should vary by scale. The results show that BDCN outperforms HED and other state-of-the-art architectures on the BSDS500 test set (Arbeláez et al., 2011).

Soria et al. (2020) developed the Dense Extreme Inception Network (DexiNed), inspired by both the HED (Xie & Tu, 2015) and Xception (Chollet, 2017) networks. Similarly to HED and BDCN, the model produces predictions at multiple resolutions that are then fused into a single prediction, while the way the information flows through the various levels of the network differs, much like the building blocks of the Xception and unlike the VGG model used in the previous two approaches. The network evaluation on different benchmarks, including a newly introduced edge detection dataset, results in improved F-measure values compared to other state-of-the-art algorithms. A more detailed description of DexiNed is provided in Section 4.2.1.

### 2.2.2 Edge-enhanced CNN-based encoder-decoder approaches

In this section, we review a few recent research works in the remote sensing and computer vision communities that studied semantic segmentation enhancement via the use of boundary information.

Chen et al. (2016) proposed a method that combines a CNN architecture based on the DeepLab model and Domain Transform (DT) filtering. This edge-preserving filtering method smooths images based on an edge reference map. Marmanis et al. (2018) experimented with enhancing the segmentation results for a classification task performed using a variation of the SegNet architecture by introducing semantically informed edge detection maps at different scale levels of the architecture. The idea is based on the hypothesis that by explicitly introducing information

relating to the boundaries between the different classes to the network, the network would end up learning to incorporate them into class predictions and thus improve the classification accuracy, especially near the object boundaries. In order to produce the boundary maps, the Holistically-Nested Edge Detection (HED) network architecture (Xie & Tu, 2015) is used as the first "building block" of the architecture and the boundary results, together with the image colour channels and a depth map, are introduced as input to the segmentation network. Both networks (SegNet and HED) are trained as a complete architecture, although a more complex training scheme is implemented at first with different parts of the architecture (HED and parts of SegNet pertaining to different spatial resolutions) being trained separately. The training and validation of the results were performed using the ISPRS Vaihingen and ISPRS Potsdam Datasets, which contain images of RGB and Infrared channels, as well as DSMs of a ground sampling distance of about 9cm (and the manually annotated ground truth labels). The synergy between the Edge Detection and the segmentation networks resulted in improved performance, with the segmentation labelling accuracy improved by up to 6% and the prediction of man-made surfaces having the most significant improvement. Alternatively, the annotation accuracy of the more fuzzy-bordered vegetation regions did not show any signs of improvement. Similarly, Jung et al. (2022) adopt HED, which extracts edge features at an encoder of a given architecture and in the proposed boundary enhancement module, an extracted edge and segmentation mask are combined, sharing mutual information.

Lyu et al. (2019) combine the outputs of an edge detection network based on MobileNet V2 (Sandler et al., 2018) and of a semantic segmentation network based on ESPNetV2 (Mehta et al., 2019) through a multilayer fusion module in order to perform real-time semantic segmentation on the Cityscapes dataset. Similarly, C. He et al. (2020) experimented with fusing the outputs of a Fully Convolutional Network (FCN) with edge information derived by a HED model and showed improved semantic segmentation performance compared to simply using an FCN network on the ESAR GID remote sensing datasets (Tong et al., 2020). H. He et al. (2021) propose an enhanced boundary learning approach in the very challenging scenario of glass-like object segmentation. They introduce a Refined Differential Module, which works with multiple resolution input features in a coarse-to-fine manner and learns to predict the edge, body and the complete glass object, and a Point-based Graph convolution network Module (PGM), which is used to improve the final prediction.

X. Li et al. (2020)developed a framework that explicitly models the body and edges of objects during semantic segmentation. They integrate their framework into a Deeplabv3+ architecture and use the principles of label relaxation (Y. Zhu et al., 2019) to train their models. Their approach, named decoupled body and edge supervision, achieved state-of-the-art results on road scene semantic segmentation benchmarks.

#### 2.2.2.1 Building Extraction using CNN models

Building extraction has been widely studied by the remote sensing community, with many CNN-based approaches being proposed (Chatterjee & Poullis, 2019; Ji et al., 2019; Shao et al., 2020; Xu et al., 2018; Yuan, 2018) and certain among them focusing on the refinement of the buildings' boundaries (Girard et al., 2021; K. Zhao et al., 2018, 2020). Zhao et al. (2020) propose a two-step method for a refined extraction of building boundaries that first uses a variation of mask R-CNN for building instance segmentation and then refines the noisy building information using a Graph Convolutional Network (GCN) that learns the geometric shapes of building polygons. Girard et al. (2021a) introduced an additional frame field output, besides the building interiors and building boundary outputs, to an encoder-decoder network for building extraction. The additional frame field information improves the segmentation quality and is also useful for building boundaries polygonization via a newly introduced algorithm that extends the concept of the Active Contour Model (ACM).

## 2.3. Semi-supervised Learning and Domain Adaptation

As a large number of labelling data is a necessary element in supervised learning, we review state-of-the-art algorithms for semi-supervised learning and domain adaptation that address the lack or small number of training data. Relevant works on classification, semantic segmentation, object detection and change detection were studied. As mentioned in Chapter 1, the terms semi-supervised learning and domain adaptation refer to very similar concepts. Domain adaptation approaches aim to compensate for the degradation in performance due to domain shift between a training (source) and test (target) domain and can be supervised, semi-supervised, or unsupervised depending on the availability of labelled data in the target domain (B. Sun et al., 2016). Semi-supervised learning approaches aim to leverage unlabelled data (which are usually much easier to acquire) and (a usually much smaller amount) of labelled data to improve a model's performance on a given task. Although in the context of semi-supervised learning,

usually both the labelled and unlabelled components are considered to belong to the same dataset (i.e., same distribution) (Ouali et al., 2020a) in some research works, the term semi-supervised approaches has been loosely used to refer to domain adaptation approaches (e.g. (Peláez-Vegas et al., 2023)). In our work, we sometimes also loosely use the term semi-supervised approaches to refer to methods based on consistency regularization compared to adversarial methods.

In a comprehensive literature review of semi-supervised learning for semantic segmentation applications, Peláez-Vegas et al. (2023) classify semi-supervised methods into five main categories:

- Adversarial methods, which apply a GAN-like structure and adversarial training,
- Methods based on consistency regularization, which introduce an auxiliary component to the loss function that ensures consistent predictions under various perturbations,
- Self-training methods based on pseudo-labels created using models trained on the labelled data (e.g. (Yang et al., 2022)),
- Methods based on contrastive learning and
- Hybrid methods that incorporate components from more than one category.

In our review, we mainly focus on two of the most prominent categories: the methods that rely on generative adversarial networks and the methods based on consistency regularization, which is also the category under which our method described in Chapter 7 falls.

### 2.3.1 Semi-supervised Learning methods and their adoption for Domain Adaptation

The presented methods are organized according to the end task the semi-supervised frameworks were designed to solve: classification, semantic segmentation, and object detection.

#### 2.3.1.1 Classification

Laine & Aila (2017) extend the idea of a neural networks' ensemble by proposing self-ensembling: a training framework that combines the outputs of different instances of the same network corresponding to different training epochs, different regularization constraints and using inputs that have been subjected to different augmentations. In the lack of labelled training data, the ensemble predictions are more likely to be closer to the missing ground truth compared to the

predictions of the current model and thus can be used as pseudo-labels for training. The authors explore two ways to implement self-ensembling: the Π model, which is based on self-ensembling based on the application of random dropout and input augmentations and a temporal ensembling model, which aggregates the predictions of models collected during different training epochs. At the time, both approaches achieved state-of-the-art results on semi-supervised classification tasks. Finally, another interesting finding of a side experiment of the paper was that when used in conjunction with supervised learning, the self-ensembling approach helps the models to develop a higher tolerance to incorrect labels in the training set compared to a typical fully supervised approach.

Tarvainen and Valpola (2017) built on the work of Laine and Aila (2016) and proposed Mean Teacher, a method that computes the Exponential Moving Average (EMA) of model weights instead of averaging over the model's predictions. This way, the EMA (also known as the teacher model or Mean Teacher) is updated after each iteration and not after each epoch, which significantly increases the pace at which the training information is incorporated into the training process. The results indicate a significant improvement in training accuracy and enable the models to learn using a smaller number of labelled samples compared to the approach proposed by Laine and Aila (2016).

Miyato et al. (2019) propose a regularization method for supervised and semi-supervised learning based on adversarial examples (I. J. Goodfellow et al., 2015) and a virtual adversarial loss, which is a measure of the model's local smoothness around each input data point. The method looks for the direction of the perturbation that can cause the greatest change to the output distribution (the model's output), and unlike (I. J. Goodfellow et al., 2015), it does not require labelled examples to compute the adversarial component of the loss function as it relies on "virtual" examples (model predictions) and thus is named virtual adversarial training (VAT).

Ke et al. (2019) introduce the Dual Student training framework for semi-supervised learning, which replaces the exponential moving average teacher model with another student network. The two student models share the same network architecture but are initialized and optimized separately in order to ensure that their weights are not tightly coupled. The two models are trained using a novel stability constraint in conjunction with a consistency regularization loss and a supervised classification loss component. The stability constraint ensures that each sample

satisfies the smoothness assumption for each model and that the corresponding models' prediction probability is safely above the decision boundary. A variation of the method using multiple student models (4 instead of 2) that are randomly divided into Dual Student pairs on each iteration and updated separately achieved even further performance improvements on semi-supervised learning benchmarks for image classification.

Verma et al. (2022) propose Interpolation Consistency Training (ICT), an efficient semi-supervised learning approach for classification which applies a Mean Teacher training paradigm on the interpolations of unlabelled samples computed using MixUp (H. Zhang et al., 2018). They argue that in classification tasks, ICT increases the probability of the smoothness assumption being satisfied and show that ICT improves the networks' performance on two semi-supervised benchmark datasets.

#### 2.3.1.2 Semantic Segmentation

Li et al. (2021) propose a semi-supervised semantic segmentation approach for medical imagery that incorporates both a supervised and an unsupervised component in the loss function used for training the network. For the unlabelled samples of the dataset, the algorithm learns to make consistent predictions by utilizing a regularization term that tries to minimize the difference between predictions of the same input that has been subjected to different perturbations (gaussian noise, dropout, geometric augmentations). The model also makes use of a Mean Teacher and student scheme (Tarvainen & Valpola, 2017) when computing the consistency regularization term, where the weights of the teacher are an exponential moving average of the student's weights on different training epochs. The proposed approach was validated in three different medical image segmentation tasks and achieved state-of-the-art results.

S. Li et al. (2021) propose a similar variation of the Mean Teacher approach for the segmentation of 3D left atrium MR medical images, which they enhance by applying multi-scale deep supervision for both the supervised and the consistency regularization components of the loss function.

French et al. (2019) show that the use of strong augmentations such as CutMix (Yun et al., 2019) and CutOut (DeVries & Taylor, 2017) on the input images can improve the performance of semi-supervised training frameworks based on consistency regularization.

Ouali et al. (2020) present another consistency-inspired semi-supervised approach for semantic segmentation based on a novel cross-consistency training framework. Unlike other prominent consistency-based approaches (French et al., 2019; X. Li et al., 2021; Tarvainen & Valpola, 2017), this method applies various perturbations to the encoder outputs, not the image input. The proposed architecture consists of a shared encoder, a primary decoder that is used for supervised training, and a set of multiple auxiliary decoders that take as inputs perturbed versions of the encoder's output and are trained to produce similar outputs via a consistency regularization constraint. The encoder is trained using all labelled and unlabelled samples. In contrast, the primary decoder is trained solely by the supervised component of the loss function, and the auxiliary decoders are solely trained by the unsupervised components of the loss function, respectively.

Liu et al. (2022) propose a consistency learning semi-supervised approach for semantic segmentation that combines image, feature, and network perturbation. They extend the Mean Teacher framework by introducing a second teacher model and propose a variation of the virtual adversarial training (VAT) loss for applying feature perturbations, which are computed using the more stable teacher models, to the student models.

Another application of the Mean Teacher training scheme for semantic segmentation, this time in a remote sensing setting, was reported by Hobley et al. (2021), where the Mean Teacher method is used to train a Fully Convolutional Network for seagrass monitoring from Remotely Piloted Aircraft (RPA) Very High Resolution (VHR) imagery. The method was compared to a fully supervised training setup as well as to an Object-based Image Analysis (OBIA) approach, resulting in improved results compared to the fully supervised setting but still not as good as the results achieved using OBIA.

#### 2.3.1.3 Object Detection

Mi et al. (2022) developed an active sampling training scheme that subjects the unlabelled image inputs to gradually stronger augmentations based on each sample's usefulness for training a semi-supervised object detection (SSOD) network. They quantify the usefulness of each sample by introducing the difficulty, information, and diversity metrics that respectively reflect the entropy of the probability distribution predicted by the model, the amount of information for SSOD, and the distribution of object categories in an image. The proposed active sampling

algorithm is incorporated in a Mean Teacher training framework (Tarvainen & Valpola, 2017), which the authors name Active Teacher.

Zhou et al. (2023) apply the Mean Teacher paradigm on a YOLOv5 detector for domain adaptive object detection. They enhance the Mean Teacher process by introducing additional pseudo-images produced using a pre-trained style transfer network (they use Contrastive Unpaired Translation (CUT) (T. Park et al., 2020)) on the source and target domain.

### 2.3.2 Domain Adaptation based on Generative Adversarial Networks

Many successful research approaches have been based on generative models and adversarial training for domain adaptation applications.

#### 2.3.2.1 Classification

Tzeng et al. (2017) developed the *Adversarial Discriminative Domain Adaptation (ADDA)* method, which they applied to bridge the domain shift between classification datasets in an unsupervised manner. The approach consists of three distinct stages:

- First, a CNN classifier is trained in a supervised manner using the source dataset.
- Next, the method makes use of the classifier's encoder to train a target encoder through a discriminator that predicts whether an image belongs to the source or the target domain. This way, and via a loss function commonly used in GAN, the target encoder learns to produce a feature representation similar to the one of the source's encoder.
- During the final stage, the method uses the trained target encoder from stage 2 to map the target images to the shared feature space in conjunction with the classifier from stage 1 to predict the classes of images from the target domain.

Bousmalis et al. (2017) introduce another unsupervised domain adaptation approach based on generative adversarial training. Their proposed framework consists of:

- a generator that takes as input an image from the target domain and a noise vector and generates a new fake image,
- a discriminator that tries to discriminate fake images (images produced by the generator) from real ones (coming from the source domain),
- a task-specific classifier that assigns task–specific labels to an image.

The model is trained using a typical GAN minmax objective function augmented by a task-specific loss component.

Liu et al. (2017) propose an unsupervised image-to-image translation framework based on a shared latent space assumption that builds on the Coupled GAN (M.-Y. Liu & Tuzel, 2016) framework. The method combines variational autoencoders and GAN and imposes weight-sharing constraints on certain layers of the source and target encoders as well as the GAN generators in order to enforce the shared-latent space. The approach was successfully applied to a domain adaptation task for image classification.

#### 2.3.2.2 Semantic Segmentation

Hoffman et al. (2017) introduced Cycle-Consistent Adversarial Domain Adaptation (CyCADA), an approach that aims to adapt representations both at the feature and the pixel level. The approach builds on the cycle-consistent image-to-image translation technique by incorporating a semantic loss component that enforces semantic consistency in addition to the cycle-consistency loss that encourages the preservation of local structural information by the cross-domain transformation. CyCADA was successfully applied for the adaptation of an FCN trained on synthetic street view data (GTA5) to the CityScapes semantic segmentation dataset.

Murez et al. (2018) propose a deep learning framework similar to (Tzeng et al., 2017) for domain adaptation, which they apply among other tasks for the semantic segmentation adaptation between real world and synthetic images (Cityscapes and GTA5 datasets). Their framework consists of multiple sub-networks – an encoder and a decoder network for each domain and multiple discriminators – and incorporates multiple loss functions, including a task-specific supervised loss, an identity loss, a typical GAN discriminator loss and multiple translation adversarial and cycle consistency loss terms.

Vu et al. (2019) propose an entropy-based unsupervised domain adaptation approach for semantic segmentation. They try to bridge the domain shift between the source and target domain by introducing additional loss terms based on the entropy of the pixel-wise predictions, which they apply (a) directly as an additional loss term and (b) using an adversarial training framework. The latter approach results in better performance as it exploits the structural consistency between

the source and target domains. It leads to state-of-the-art results in synthetic-to-real domain adaptation for road scene semantic segmentation.

#### 2.3.2.3 Domain Adaptation in Remote Sensing and Change Detection Applications

During the past few years, there have been research works that have adapted or further developed methods based on generative adversarial training and the cycle consistency loss for bridging the domain gap between different remote sensing datasets mainly for pixel-wise classification applications.

Wittich & Rottensteiner (2021) propose an appearance adaptation method for the semantic segmentation of aerial images which combines an unsupervised adversarial training scheme of an appearance adaptation framework with the supervised training of a fully convolutional network for semantic segmentation, similar to (Murez et al., 2018). The authors also introduce a new criterion for selecting the optimal model in the absence of labelled validation data, which is based on an entropy-based confidence measure.

Wittich & Rottensteiner (2019) adjust the Adversarial Discriminative Domain Adaptation (ADDA) (Tzeng et al., 2017) framework to semantic segmentation tasks, which they apply using a U-Net-based encoder-decoder architecture for the pixel-wise classification of remote sensing imagery.

Wittich (2020) proposes a semi-supervised domain adaptation framework based on the entropy minimization loss introduced by (Vu et al., 2019) and a novel weighting strategy, which they apply for the automated pixel-wise classification of aerial images.

Noa et al. (2021) apply the Adversarial Discriminative Domain Adaptation (ADDA) (Tzeng et al., 2017) framework in conjunction with a margin-based regularization constraint for detecting deforestation on Landsat satellite images from three different forested areas in Brazil.

A few research works focused on the application of domain adaptation specifically for change detection applications. Soto Vega et al. (2021) propose an unsupervised appearance adaptation method for change detection based on the Cycle-Consistent Generative Adversarial Network (CycleGAN) (J.-Y. Zhu et al., 2017) which they apply for deforestation detection in the Amazon. In addition to the GAN adversarial loss, the cycle consistency loss and the identity loss terms

that are used in the original CycleGAN implementation, the authors propose the introduction of an extra loss term that aims to preserve the magnitude and orientation of the change vectors computed between the different instances of an image pair.

Soto et al. (2020) apply the CycleGAN framework for deforestation detection in the Amazon Forest and show that the method can lead to improved results by reducing the domain shift between satellite images captured at different time periods.

Noh et al. (2022) propose an unsupervised change detection method that is based on the application of a generative adversarial network and an image reconstruction loss function. The proposed approach builds on an insight found in reconstruction-based anomaly detection methods, which suggests that anomalies cannot be reconstructed by a generative model (like a GAN) and will lead to high reconstruction errors. Thus, instead of training a model to detect changes between bi-temporal images, the authors propose to photometrically transform an image and train a GAN that takes as input the transformed image and tries to reconstruct the original one. The changed regions can be detected at inference time by finding the image regions that exhibit high reconstruction loss (based on a predefined threshold).

Zheng et al. (2021) propose a single-temporal supervised learning approach named STAR. STAR consists of a deep CNN model that performs semantic segmentation on each instance of the bi-temporal image pair separately and a *"ChangeMixin"* module which takes as input features from the decoder of the semantic segmentation model for both instances and learns to predict the changed regions between the image pair. The "*ChangeMixin*" module is trained in an unsupervised way using pseudo-bi-temporal images that were created by pairing up two random images from the dataset and using as change map the disjunctive union of the corresponding building masks.

Overall, the presented literature review highlights the active research interest in improving the performance of deep learning and CNN-based approaches on change detection and other computer vision and remote sensing tasks. It also demonstrates the need for novel strategies that will reduce the reliance of DL algorithms on extensive annotated training datasets and improve the models' generalizability.

After providing a brief outline of the employed datasets in this chapter's next and final section, we begin our analysis by introducing the direct change detection approach in Chapter 3.

## 2.4. Overview of used datasets

We used multiple datasets for training and evaluating various approaches throughout this research. All datasets primarily comprised RGB satellite, aerial, and orthorectified images, ensuring minimal relief displacements and nadir viewing, which are welcome features for land-cover change detection. For each set of experiments, we chose the dataset with the highest number of samples available at the time of conducting the experiments. Table 2.1 provides an overview of these datasets. Detailed descriptions for each dataset are available in the respective chapters where they were first utilized.

*Table 2.1: Overview of used datasets.*

| Dataset | Description |
|---|---|
| **Change Detection Dataset** (Lebedev et al., 2018)<br><br>Used for training and evaluation of models for direct change detection applying supervised and semi-supervised training frameworks (Chapters 3,4, and 5)<br><br>Described in Section 3.2.1 | - 10,000 training, 3,000 validation, and 3,000 test image pairs (Size: 256×256 pixels)<br>- Images taken from Google Earth (Digital Globe)<br>- Ground resolution ranges from 30 cm to 100 cm<br>- Samples include both natural and synthetic changes<br>- Changes focus on the appearance/ disappearance of objects and ignore seasonal variations (most image pairs were collected at different seasons) |
| **Vaihingen Dataset** (Cramer, 2010) for land cover semantic segmentation<br><br>Used for training models for boundary detection (Chapter 4)<br><br>Described in Section 4.2.3 | - 33 patches of different sizes (≈ 2500 × 2000 pixels)<br>- Extracted from true orthophoto mosaic<br>- Ground resolution ≈ 9 cm<br>- Pixel-wise classification into 6 categories: impervious surfaces, buildings, low vegetation, trees, cars, and clutter/background |
| **LEVIR-CD** (Chen & Shi, 2020) for building change detection<br><br>Used for training and evaluation of models for both direct and post-classification change detection approaches applying supervised, semi-supervised, and domain adaptation training frameworks (Chapters 5 and 7)<br><br>Described in Section 5.3.2.1 | - 637 co-registered bitemporal RGB image pairs and building change maps<br>- Image Size: 1024 × 1024 pixels<br>- Image Spatial Resolution ≈ 50 cm<br>- Capture Time ranges from 2002 to 2018 and may vary<br>- Designed to address significant land-use changes, variations in illumination & seasonal conditions |
| **Inria Aerial Image Labelling dataset** (Maggiori et al., 2017)<br><br>Used for training and evaluation of models for building semantic segmentation using supervised and domain adaptation training frameworks (Chapters 6 and 7)<br><br>Described in Section 6.2.4 | - Aerial Orthorectified RGB imagery and ground truth masks for building semantic segmentation<br>- Covers an area of 810 $km^2$ of dissimilar urban settlements captured with varying illumination conditions and at different seasons<br>- Ground Resolution ≈ 30 cm |
| **SpaceNetV2** (Van Etten et al., 2019)<br><br>Used for training and evaluation of models for building semantic segmentation using supervised and domain adaptation training frameworks (Chapter 7)<br><br>Described in Section 7.2.3.1 | - Developed for the extraction of building footprints<br>- Image tiles of size 650 × 650 pixels (200m × 200m)<br>- Ground resolution ranges from 30cm to 1.24 m (depending on the band/composite)<br>- Covers four cities around the world (Vegas, Shanghai, Paris, and Khartoum) |

# 3

# UNet & UNet++ Architectures in High-Resolution Image-based Change Detection Applications

## 3.1. Methodology

In this chapter we introduce and evaluate an early fusion direct change detection pipeline based on UNet-shaped encoder-decoder architectures. We chose to work with UNet and UNet++ architectures because of their simple, yet effective design. UNet is an encoder-decoder architecture consisting of a contrastive and symmetrical expanding path and skip connections connecting convolutional blocks that share the same spatial resolution. This ensures that both the finer-grained spatial information of the contractive blocks and the more context-rich information of the expanding blocks are retained during the network's learning. The UNet++ architecture is an extension of UNet that was developed to bridge the semantic gap between the contractive and expansive blocks of the network by incorporating more intermediate blocks and more skip connections between the blocks that share the same spatial resolution (Z. Zhou et al., 2018).

Our goal is to examine the effects of different UNet and UNet++ encoder-decoder architectural choices in combination with different loss functions on the performance of the trained networks for change detection applications. In addition, we investigate how the use of deep supervision and data augmentation affects the performance of the various networks.

The methodology consists of 4 sub-sections:

- In Section 3.1.1, we present the two network architectures.

- Section 3.1.2 discusses the different loss functions used to train the networks.
- In Section 3.1.3, we introduce the concept of deep supervision.
- Finally, Section 3.1.4 briefly discusses the importance of data augmentation techniques.

### 3.1.1 Network Architectures

We start our analysis with a more detailed discussion of UNet and UNet++ architectures.

#### 3.1.1.1 UNet

UNet (Ronneberger et al., 2015) was built based on the idea of the Fully Convolutional Network (FCN) architecture as introduced by (Long et al., 2015) with a goal to create a new type of architecture that can be trained using fewer training samples and which can produce higher segmentation accuracy. The main innovation of UNet that differentiates the architecture from FCNs is the introduction of more convolutional filters in the up-sampling path, creating in this way an up-sampling (or expanding) subnetwork, which is in general terms symmetric to the down-sampling (or contracting) part of the network. Thus, the final network has a U-shaped form, a fact highlighted by the given name of the architecture. Like the traditional FCN architecture, an essential part of the network is the “skip connections” that pass information from the down-sampling blocks to the corresponding up-sampling convolutional blocks. This flow of information is achieved by concatenating the high-resolution features of the down-sampling blocks to the first layer of the corresponding up-sampling block. The total set of features is then passed to the subsequent convolutional layers. In this way, the previously learned contextual information flows into the convolutional layers without losing the localization accuracy caused by the down-sampling operation of the max pooling steps. This flow of information, combined with the larger number of filters in the up-sampling path, helps ensure that the network retains both high localization accuracy and context-rich feature representations.

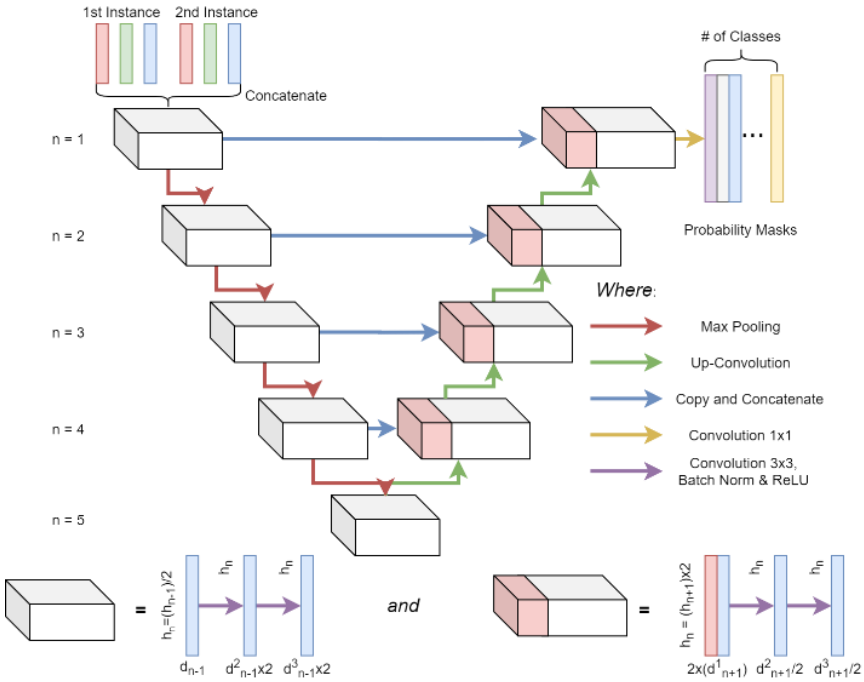



*Figure 3.1. UNet archritecture. The notation $h_n$ refers to the height of the convolutional layers at block level n and $d_n^i$ refers to the number of channels of the $i^{th}$ convolutional layer in level n.The width of each convolutional layer has an identical behavior as the height and for this reason it has been omitted. We can see that the channel depth of a convolutional layer at layer n is twice as large as the corresponding depth of the corresponding convolutional layer of block level n-1.*

More specifically, each down-sampling block consists of 2 successive convolutional layers with a 3 × 3 filter size and a padding of 1 pixel. The first element of each block is the down-sampled result of the previously applied max pooling layer, while in the first block the input consists of the concatenated RGB channels of both image instances. All max pooling layers use a 2 × 2 kernel shape and a stride of 2. Batch Normalization is applied to each convolutional layer, which was not included in the original UNet implementation, but helps the network learn faster by allowing the use of higher learning rates, reducing the need for careful initializations and having a regularization effect on training that helps reduce overfitting[3] (Ioffe & Szegedy, 2015). The number of feature channels of each convolutional layer of every down-sampled block is doubled with respect to the number of channels in the convolutional layers of the previous level, starting with 32 filters for the first two convolutional layers of spatial resolution level $n = 1$.

With regards to the expanding part of the network, the blocks are almost identical to the down-sampling blocks, with the main differences being the input's origin and the extra feature channels

[3] Batch normalization is discussed in more detail in Appendix A.

coming from the skip connections. The input to each block is the output of the directly previous coarser block in the sequence. The up-sampling operation can either be a transpose convolution operation, which has learnable weights and biases, or a simple upsampling operation like a bilinear interpolation. The convolutional layers of each expanding block have the same number of feature channels as the convolutional layers of the corresponding contracting block. When applying the skip connection, the features from the contracting block do not need to be cropped as in the original paper since we have padded the convolutional layers and kept the feature width and height constant within each block. An overview of the architecture is presented in Figure 3.1.

### 3.1.1.2 Nested UNet (UNet++)

As mentioned earlier, Nested UNet, also commonly known as UNet++, is an extension of the UNet architecture that was introduced by Z. Zhou et al. (2018) in an attempt to improve the segmentation accuracy of the UNet architecture. The authors maintain the encoder-decoder architecture of UNet and argue that a gradual enrichment of the feature maps of higher resolution before aggregating them with the decoder results would help the network capture more high-resolution details thanks to the higher semantic similarity between the concatenated features. The central hypothesis for the introduction of the UNet++ architecture is that adding more intermediate (nested) convolutional blocks and densifying the skip connections between blocks would cause the concatenated results of each up-sampling block to be more semantically similar than the results obtained by the original UNet architecture, which would ultimately result in an easier optimization problem and thus more accurate results.

Using the same definitions regarding the contracting and expanding blocks as described in the plain UNet architecture and in Figure 3.1, the general overview of the UNet++ architecture is presented in Figure 3.2. The nested convolutional blocks $X^{n,m}$ that have been introduced to bridge the semantic gap between the contracting and the expanding blocks of the same level $n$ in the pyramid are connected through skip connections with every convolutional block of the same level $n$ with $m' > m$. More specifically $X^{n,m}$ can be defined as the output of Equation 3.1 where $ConvBlock(x)$ is the output of a convolutional block given an input $x$, $\langle X^{i,j} \rangle_{k=1}^{f}$ is the concatenation operation for elements $X^{i,1}, \dots, X^{i,f}$ and $\langle x, y \rangle$ stands for the concatenation operation of elements $x$ and $y$.

$$X^{n,m} = \begin{cases} ConvBlock(\langle X^{n-1,m} \rangle), & for\ m = 1 \\ ConvBlock\left(\langle \langle X^{n,k} \rangle_{k=1}^{m-1}, X^{n+1,m-1} \rangle\right), & for\ m > 1 \end{cases} \tag{3.1}$$

Thus, this formulation leads to an architecture that consists of multiple, nested encoder-decoder architectures (like for example: $\{X^{1,1}, X^{2,1}, X^{1,2}\}$ and $\{X^{1,1}, X^{2,1}, X^{3,1}, X^{2,2}, X^{1,3}\}$), with each intermediate block receiving information from multiple feature maps with both skip connections and upsampling blocks. According to Z. Zhou et al. (2018), this more gradual flow of information results in more semantically similar feature maps, which make the optimization problem easier to solve and lead to higher semantic segmentation accuracy than the plain UNet architecture.

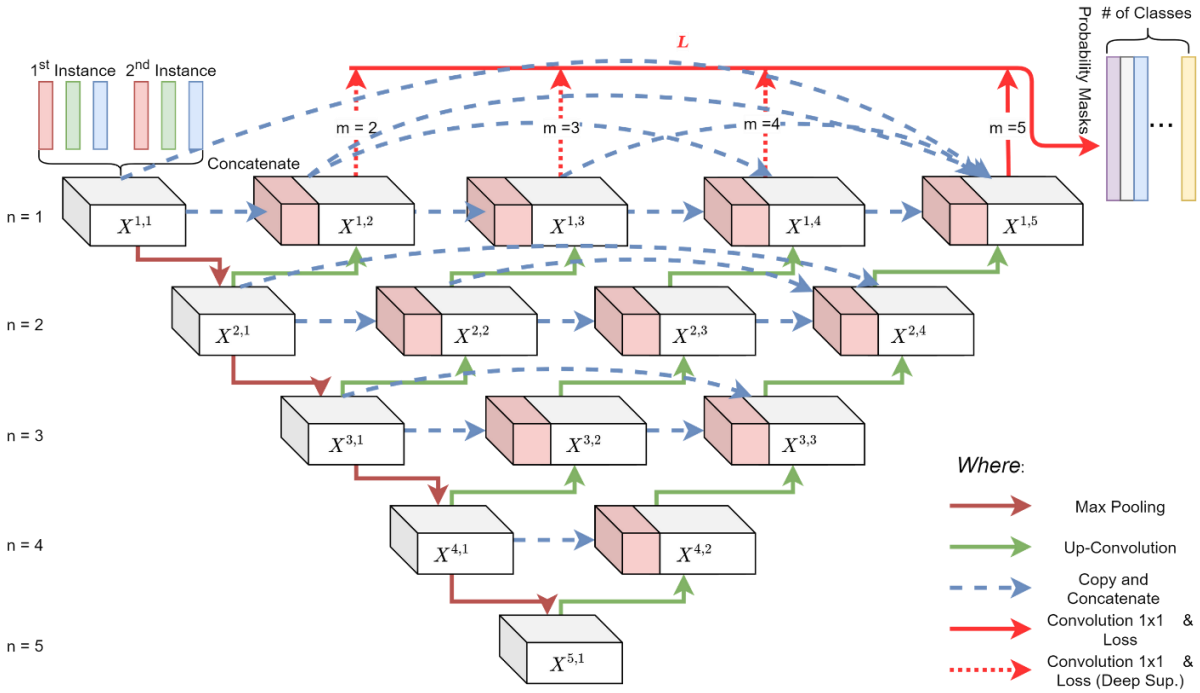



*Figure 3.2: Nested UNet Architecture.*

## 3.1.2 Loss Functions

We have investigated and compared two different loss functions: the first one is a combination of Binary Cross-Entropy loss with the Dice coefficient function (BCE-Dice loss) (Equation 3.2), and the second one is the Lovász Hinge loss (Berman et al., 2018), which is a tractable surrogate for the optimization of the intersection over union measure. The combination of the BCE-Dice loss was used in the original UNet++ paper (Z. Zhou et al., 2018) as well as by (D. Peng et al., 2019) and produced state-of-the-art results.

$$L(Y,\hat{Y}) = -\frac{1}{N}\sum_{b=1}^{N}\left(\lambda \cdot (Y_b \cdot \log(\widehat{Y_b}) + (1 - \widehat{Y_b}) \cdot \log(1 - Y_b)) + \frac{2 \cdot Y_b \cdot \widehat{Y_b}}{Y_b + \widehat{Y_b}}\right) \quad (3.2)$$

The parameter $\lambda$ in Equation 3.2 is set to 0.5, following the setup of Z. Zhou et al. (2018) in the original UNet++ paper. This value was also experimentally shown to produce the best results on a change detection application developed by D. Peng et al. (2019). $Y_b$ and $\widehat{Y_b}$ are the flattened ground truth and predicted probability maps respectively of image $b$ and $N$ is the batch size (number of samples per batch).

Regarding Lovász Hinge loss, Berman et al. (2018) are introducing a tractable surrogate of the Jaccard index, which is defined as the Intersection over Union $J_C(y,\hat{y})$ (Equation 3.3) with the convention that $\frac{0}{0} = 1$. The parameter $c$ is the class value for each pixel and given a set $A$ the operator $|A|$ returns the total number of elements in the set.

$$J_C(y,\hat{y}) = \frac{|\{y = c\} \cap \{\hat{y} = c\}|}{|\{y = c\} \cup \{\hat{y} = c\}|} \quad (3.3)$$

In our case we are dealing with a binary problem and so the parameter $c$ of the Jaccard index will be equal to 1 (Equation 3.4) since we are only considering the foreground object, meaning pixels that are mapped as changed.

$$J_1(y,\hat{y}) = \frac{|\{y = 1\} \cap \{\hat{y} = 1\}|}{|\{y = 1\} \cup \{\hat{y} = 1\}|} \quad (3.4)$$

The corresponding loss is given by (Equation 3.5), which is not differentiable as the parameters $y$ and $\hat{y}$ can only take binary values $\in \{-1, 1\}$. Berman et al., (2018) use the Lovász extension of a set function and develop an algorithm that computes an optimization surrogate of (Equation 3.5).

$$\Delta_{J_1}(y,\hat{y}) = 1 - J_1(y,\hat{y}) \quad (3.5)$$

A SoftMax version of the extension has been used by Rakhlin et al. (2018) in combination with the UNet architecture for a land cover classification task and produced promising results.

### 3.1.3 Training - Deep Supervision

UNet++ can also be trained using Deep Supervision (Lee et al., 2015), where the overall loss is computed by aggregating the loss of the output layer (the output of the convolutional block $X^{1,5}$after applying a 1x1 convolutional layer) with the "companion"[4] losses of all the intermediate layers of the first level of the pyramid $X^{1,k}$ for $k = 1,..,4$. The companion losses are computed by applying the loss function to each output $X^{1,k}$. The overall loss is then computed as the average of all five losses. In (Z. Zhou et al., 2018) the use of deep supervision gave both better and slightly worst results depending on the dataset being used (all datasets related to medical segmentation tasks) compared to using only the output layer loss of the network.

### 3.1.4 Data Augmentation

Data augmentation techniques have been shown to improve the performance of similar networks (D. Peng et al., 2019; Ronneberger et al., 2015; Z. Zhou et al., 2018) especially when training using relatively small datasets and to reduce overfitting to the training data. Thus, we evaluate and quantify the effect of data augmentation on the training of different architectures with different loss functions and in combination to deep supervision. We have augmented the original dataset by occasionally (with a 50% probability) performing a horizontal flip to the input training images, a technique that even though very simple proved to be quite effective as will be shown in the results section.

## 3.2. Experiments

The experiments were designed to evaluate the performance of UNet and UNet++ architectures on detecting changes from satellite images and to study the effects of data augmentation, deep supervision, and the two different loss functions, BCE-Dice loss and Lovász Hinge loss, on the quality of the trained models' predictions. Thus, this set of experiments will help us identify the combination of training design parameters that leads to the best performance on the test set.

[4] The term companion loss is used by (Lee et al., 2015)

The section begins with a brief introduction of the change detection dataset, followed by the training process implementation details, and closes with a description of the selected evaluation metrics.

### 3.2.1 Test Dataset

The dataset we used to train and evaluate the performance of each approach on the change detection task was created by Lebedev et al., (2018) and consists of 10,000 training samples, 3,000 validation samples, and 3,000 test samples. Each sample comprises two RGB satellite images covering the same region and a ground truth mask, where the changed regions are annotated. The satellite images are 256×256 pixels each, taken from Google Earth (DigitalGlobe), and sampled from 11 larger images, from varying seasons, with a ground resolution that varies from 30cm to up to 100cm per pixel (7 pairs of images that are 4725×2700 pixels for the creation of manual ground truth masks; 4 pairs of images that are1900×1000 pixels and include natural and synthetic changes). The masks focus only on changes that relate to the appearance or disappearance of objects between the two instances of the pair without taking into account any seasonal variations, which allows us to evaluate whether the models can learn to detect certain types of changes and ignore others based on the semantics of the training samples. In most cases, the two instances of each image pair were captured during different seasons, while some artificial changes were introduced to the dataset by the manual addition of objects.

### 3.2.2 Training Implementation

For the training, we have trained all networks for 260 epochs, applying the Adam optimization algorithm in all cases using the default parameters for the coefficients of the running average (0.9 and 0.999) and with no weight decay. The batch size consisted of eight images, and the learning rates used were set to 0.0003 when training using the BCE-Dice Loss function and 0.0005, which was gradually reduced to 0.0001 when training using the Lovász-Hinge loss function. The training was performed using the PyTorch framework[5] on an NVIDIA GeForce GTX 1080Ti GPU.

[5] We have built on the UNet++ PyTorch implementation available at: https://github.com/4uiiurz1/pytorch-nested-unet

### 3.2.3 Evaluation Metrics

The evaluation metrics that have been used are Precision, Recall, F1 Score, and Accuracy (Equations 3.6 to 3.9), where TP (True Positive) is the number of pixels that were correctly classified as changes, TN (True Negative) is the number of pixels that were correctly classified as unchanged, FP (False Positive) is the number of pixels that were classified as changed while they were not actually changed, and FN (False Negative) is the number of pixels that were mistakenly classified as unchanged (Figure 3.3). Change Detection algorithms must deal with highly unbalanced data with respect to the proportion of changed regions compared to the unchanged area of the scene. In many cases, the changed area can cover less than 5% of a change map, which means that a Network that never detects any change and classifies the whole image as unchanged would score an accuracy higher than 95%. Thus, the measure of accuracy can be widely misleading in a Change Detection task since it does not distinguish between changed and unchanged regions, and the Precision, Recall, and F1 score measures represent much more realistic evaluation metrics.

$$Precision = \frac{TP}{TP + FP} \tag{3.6}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.7}$$

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall} \tag{3.8}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.9}$$

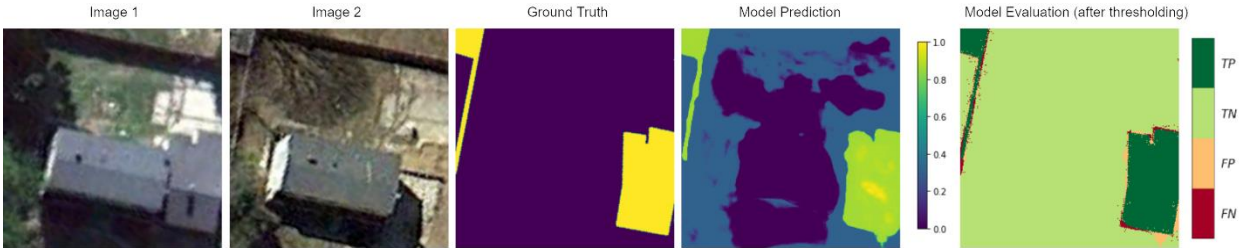



*Figure 3.3: Model prediction and visualization of the TP, TN, FP and FN values for an image sample given the two image instances and the ground truth change map.*

## 3.3. Results

The evaluation metrics over the test set for UNet and UNet++ architectures trained using either the BCE - Dice Loss or the Lovász Hinge Loss functions are summarized in Table 3.1. The table also presents the perceived changes in the performance on the test set caused by the incorporation of data augmentation and deep supervision (when applicable) during training. The comparison of the results suggests that the UNet++ architecture, when trained using the BCE-Dice Loss function with data augmentation, produces the best results over the test set. Figures 3.4 to 3.9 illustrate the effects of the different design and training choices on the performance of each trained network on the test set.

*Table 3.1: Results Summary. Evaluation metrics for various choices regarding the architecture of the network, the loss function being used for training and the use of deep supervision and data augmentation during training.*

| **Design and Training Choices** | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Architecture** | UNet++ | UNet++ | UNet++ | UNet++ | UNet++ | UNet++ | UNet++ | UNet++ | UNet | UNet | UNet | UNet |
| **Loss Function** | Lovász | Lovász | Lovász | Lovász | BCE Dice | BCE Dice | BCE Dice | BCE Dice | Lovász | BCE Dice | Lovász | BCE Dice |
| **Deep Supervision** | ✓ | × | ✓ | × | ✓ | × | ✓ | × | × | × | × | × |
| **Data Augmentation** | × | ✓ | ✓ | × | × | ✓ | ✓ | × | ✓ | ✓ | × | × |
| **Evaluation Metrics** | | | | | | | | | | | | |
| **Precision** | 0.8504 | 0.8683 | 0.8770 | 0.8602 | 0.9565 | **0.9668** | 0.9575 | 0.9599 | 0.8455 | 0.9572 | 0.8667 | 0.9512 |
| **Recall** | 0.8807 | 0.9031 | **0.9076** | 0.8804 | 0.8831 | 0.9034 | 0.8977 | 0.8781 | 0.8967 | 0.8989 | 0.8604 | 0.8656 |
| **F1** | 0.8654 | 0.885342 | 0.8920 | 0.8702 | 0.9183 | **0.9340** | 0.9267 | 0.9172 | 0.8704 | 0.9271 | 0.8635 | 0.9064 |
| **Accuracy** | 0.9638 | 0.9680 | 0.9710 | 0.9661 | 0.9842 | **0.9870** | 0.9856 | 0.9841 | 0.9638 | 0.9858 | 0.9700 | 0.9822 |

Figure 3.4 shows the effect of data augmentation and deep supervision on the predictions of the UNet++ architecture trained using the Lovász Hinge loss function. The use of data augmentation has a positive effect on all the metrics, while the use of solely deep supervision has a negative effect on Precision (of about 1%) and on F1 score. However, the best results were obtained when using both deep supervision and data augmentation. Similarly, Figure 3.5 presents the effect of data augmentation and deep supervision on the prediction of the UNet++ architecture trained using the combination of the Binary Cross Entropy loss with the Dice loss (BCE-Dice loss)

function. The results suggest that deep supervision and data augmentation produce higher metric values than the baseline scenario, where none of the two techniques was used, with the use of data augmentation having a more significant positive effect. The best results for all metrics are retrieved when applying only data augmentation (and no deep supervision) when training the network.

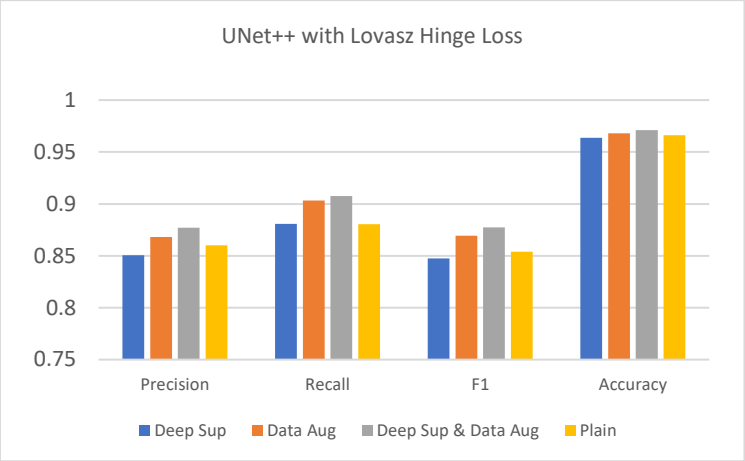



*Figure 3.4: Effect of Data Augmentation (Data Aug) and Deep Supervision (Deep Sup) on UNet++ architecture when training using the Lovász Hinge Loss.*

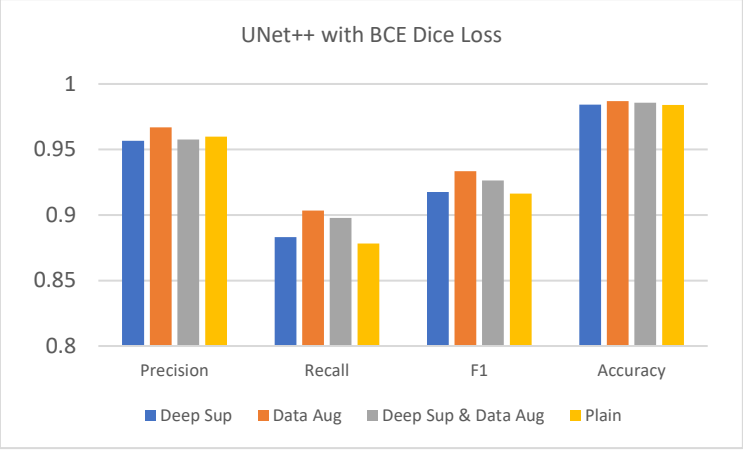



*Figure 3.5: Effect of Data Augmentation (Data Aug) and Deep Supervision (Deep Sup) on UNet++ architecture when training using the BCE Dice Loss.*

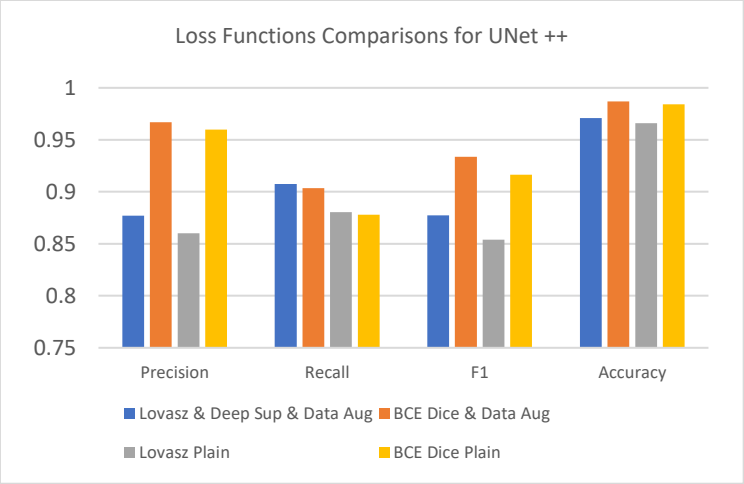



*Figure 3.6: Comparison of the effects of using different loss functions on the UNet++ test results.*

Figure 3.6 illustrates a direct comparison of the retrieved metrics when using different loss functions on the same UNet++ architecture. The combination of the BCE loss and Dice coefficient achieves better precision (by more than 8%) and slightly worse recall than the use of the Lovász Hinge loss. This means that the network trained with the BCE Dice loss produces fewer pixels falsely classified as changed, but at the same time, it detects slightly fewer changes than the network trained with the Lovász Hinge loss. The F1 score, as the geometric mean of the precision and recall rates, is 7% higher for the BCE Dice loss. Likewise, the comparison of the effects of the different loss functions on the UNet architecture is presented in Figure 3.7, where the effects are similar to the ones derived from Figure 3.6, and the precision gap is even larger (higher than 11%).

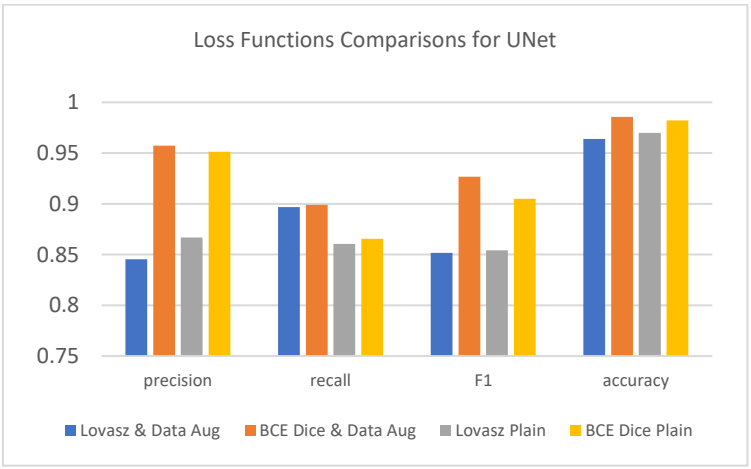



*Figure 3.7: Comparison of the effects of using different loss functions on the UNet test results.*

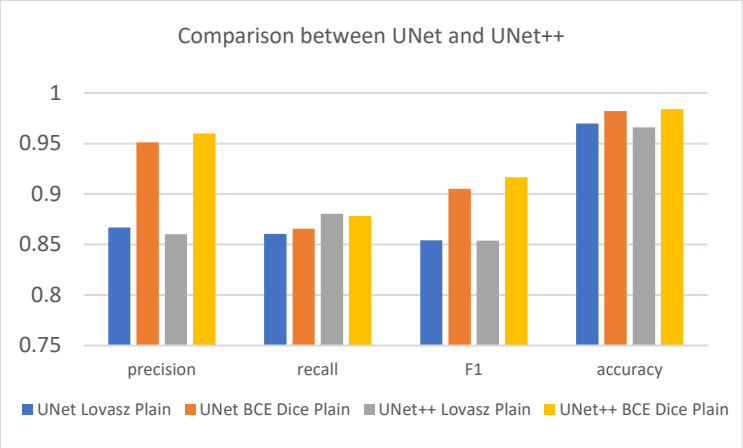



*Figure 3.8: Comparison of the effects of different architectures and different loss functions.*

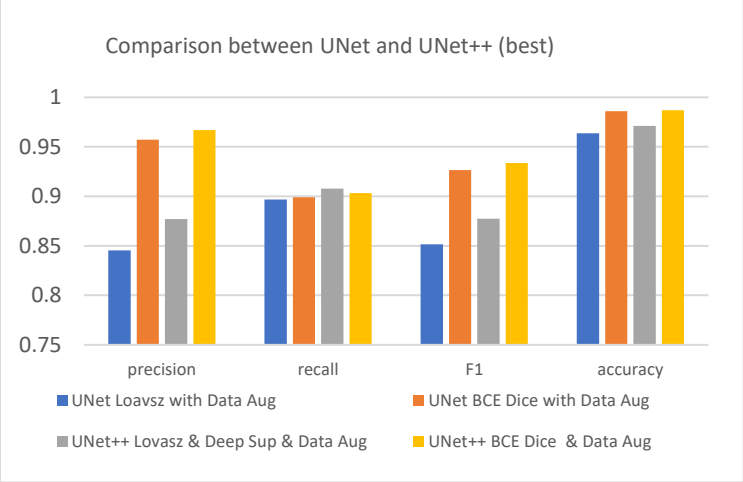



*Figure 3.9: Comparison of the best results achieved with each architecture and each loss function.*

The effects of using different architectures and loss functions without data augmentation or deep supervision are presented in Figure 3.8. The choice of the loss function affects the performance of the network more than the choice of UNet or UNet++ architecture, with the BCE Dice loss yielding higher metric values. Also, using the UNet++ architecture produces slightly better results than UNet given the same loss function, with the differences being less than 1% for all metrics. Similarly, the comparison of the best results achieved for each architecture and each loss function is shown in Figure 3.9. Once again, the choice of the loss function seems to affect the results more than the choice of architecture, with the best results being produced by the UNet++ trained with BCE Dice Loss and data augmentation followed closely (less than 1% difference on every metric) by the UNet network trained with BCE Dice loss and data augmentation.


*Figure 3.10: Change Detection Prediction Example Using UNet++.*

An example of change detection prediction using UNet++ is shown in Figure 3.10. A set of exemplary results retrieved from different combinations of architectures, loss functions, and the use of data augmentation and deep supervision are presented in Figure 3.11. By visually examining the results, TP (change) and TN (no change), we can argue that all networks perform reasonably well on the CD task and can learn to ignore seasonal effects like snow and seasonal vegetation changes.

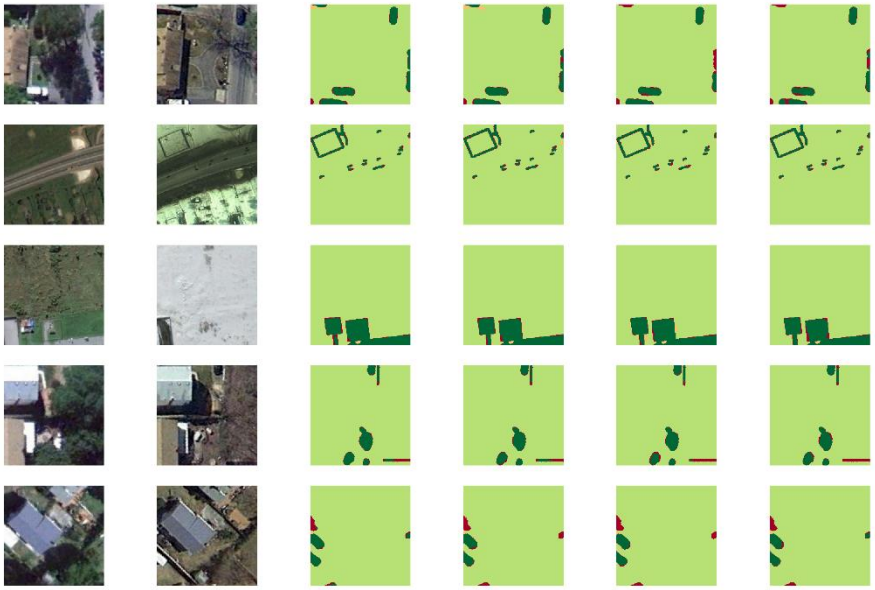


*Figure 3.11: Examples of change masks for various models. The color scale is the same as in Figure 3.10. From left to right the models are: (a) the UNet++ trained with BCE Dice loss and data augmentation, (b) the UNet++ trained with Lovasz Hinge loss deep supervision and data augmentation, (c) the UNet trained with BCE Dice loss and data augmentation and (d) the UNet trained with Lovasz Hinge loss and data augmentation.*

## 3.4. Summary

In this chapter, we have investigated and experimented with two encoder-decoder CNN architectures for change detection applications using high-resolution satellite images. We have also compared two different loss functions for the training of the CNNs and evaluated the contribution of data augmentation and deep supervision techniques on the performance of the

networks. All networks produced state-of-the-art results with the network using the UNet++ architecture and being trained with the BCE Dice Loss and data augmentation performing the best on the test data.

In the next chapter, we are going to examine two ways to enhance the semantic segmentation performance of the direct approach by:

- Incorporating object boundary information into the U-shaped encoder-decoder architectures
- Introducing additional geometric augmentations to the images of the input sample pairs.

# 4

# Performance improvement of U-shaped CNN architectures using boundary information and geometric augmentations[6]

## 4.1. Introduction

This chapter investigates ways to improve the state-of-the-art Convolutional Neural Network architectures designed and trained to detect land cover changes from a pair of satellite images. To achieve that, we enhance the encoder-decoder-based CNN direct change detection approach with semantically informed edges produced by a second network that explicitly performs boundary detection on each of the two instances of the image pairs and then introduces those edges into our Change Detection network. We show that incorporating boundary information into the encoder-decoder architecture leads to improved network semantic segmentation performance, especially near the location of object transitions involving objects with well-defined boundaries, like buildings and roads. We use the DexiNed model (Soria et al., 2020), a deep convolutional neural network architecture that detects image edges in an end-to-end manner, for the edge detection component of the architecture. We experiment with both UNet and UNet++

[6] The contents of this chapter are heavily based on (Bousias Alexakis & Armenakis, 2021b). Permission to use the material from the paper has been granted by the T&F publisher.

architectures for the change detection semantic segmentation component and evaluate the potential benefits in each case.

Furthermore, we devise a way to make the models more robust to the impact of misregistration errors between multi-temporal images by training on image pairs where the co-registration has been deliberately distorted. We use rotations and affine transformations to distort the alignment between image pairs of the training data. The results of our experiments suggest that this training technique, compared to the results obtained from the original models, increases the performance of the models significantly when dealing with significant mis-registrations and even results in minor performance improvements in cases where there are no mis-registrations introduced between the two image instances.

We begin this chapter with the introduction of the proposed model and training enhancements as well as a description of the datasets we used for training and evaluation purposes in Section 4.2.

## 4.2. Methodology

This section presents the proposed edge-enhanced encoder-decoder architecture and then introduces the Change Detection datasets under consideration. Next, we introduce our approach for dealing with high misregistration errors between the multitemporal images of each sample, and finally, in section 3.4, we present the parameters and loss functions used for training each model.

### 4.2.1 Edge-Enhanced Encoder-Decoder for Change Detection

The two encoder-decoder architectures that we use as our model's backbone architecture are the UNet (Figure 4.1) and UNet++ architectures (Figure 4.2), which were introduced in Section 3.1.1 and have been shown to produce high-quality results on Change Detection tasks (Daudt et al., 2018b; Daudt et al., 2019; D. Peng et al., 2019; Bousias Alexakis & Armenakis, 2020). Even though both networks perform very well on the change detection task, there is still room to improve the segmentation performance, especially near the boundaries of different objects in the images. To do this, we introduced two new feature maps (edge maps), computed using the DexiNeD model, into our backbone UNet and UNet++ networks. The boundaries for each instance of an image pair are computed using the DexiNeD architecture and then fed into the backbone architecture, together with the two RGB instances.

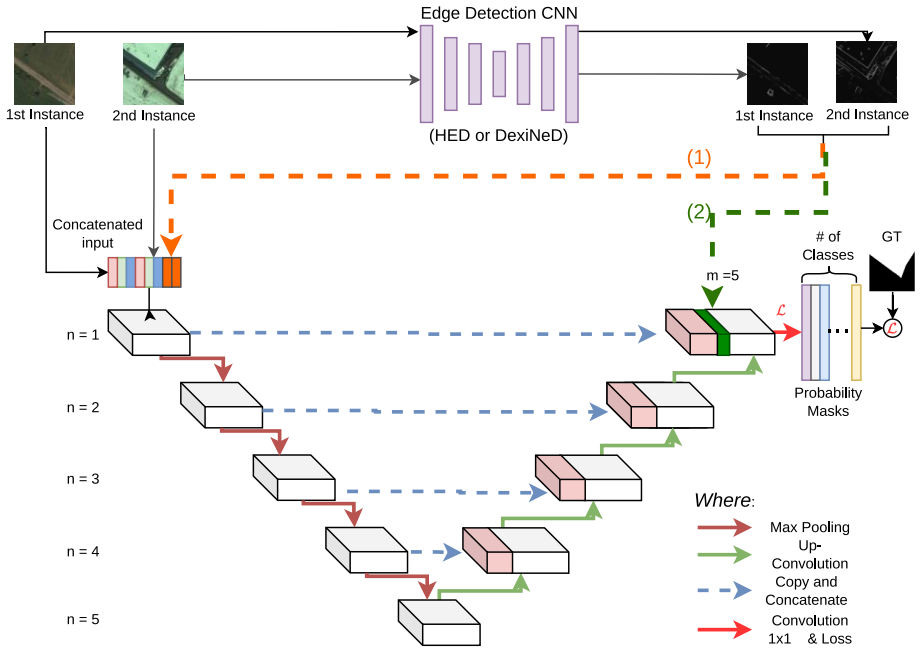



*Figure 4.1: UNet Architecture and Edge Enhanced UNet Architecture for change detection. On the first and last convolutional blocks of level n=1 of the network, we insert boundary information produced by the DexiNeD model. The structure of each convolutional block is presented at the bottom of the figure, where $h_n$ and $d_n$ are the height and depth of the feature map at level n, and "x" is used to denote multiplication.*

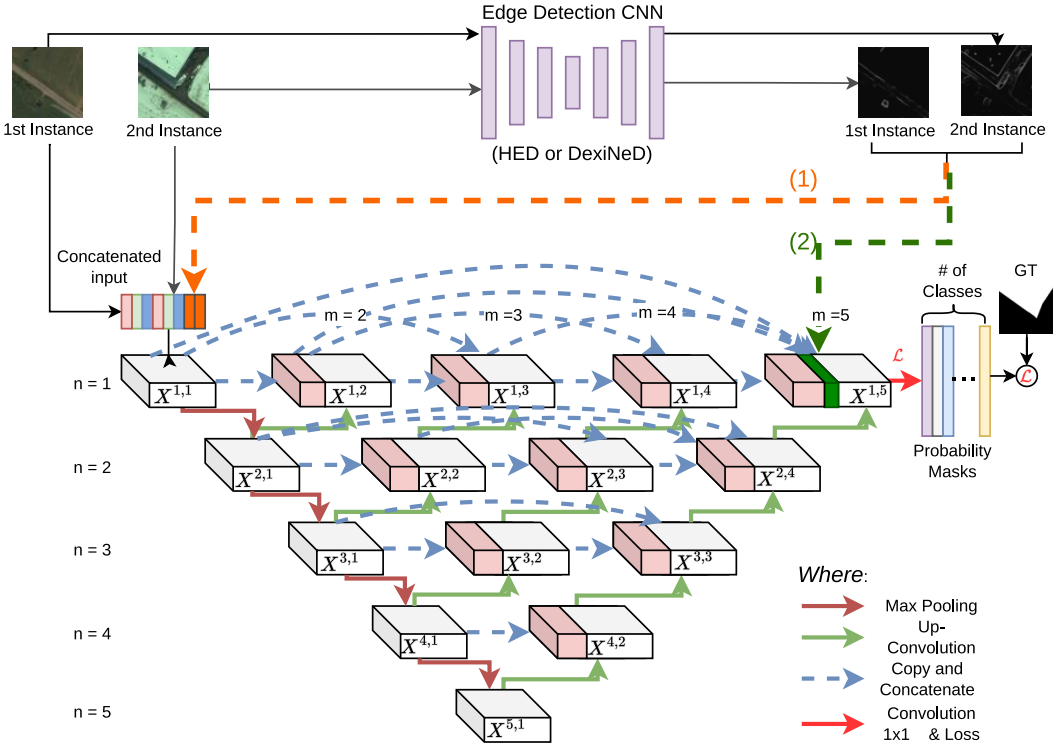



*Figure 4.2: UNet++ Architecture and Edge Enhanced UNet++ architecture for change detection. On the first and last convolutional blocks, $X^{1,1}$ and $X^{1,5}$, of level 1 of the network, we insert boundary information produced by the DexiNeD model.*

DexiNed architecture is presented in Figure 4.3. The main idea of DexiNed is similar to other networks presented in this paper (UNet, UNet++, HED, BDCN). The network consists of two main components: the first gradually decreases the spatial resolution of the input image and increases the number of filter channels to learn features at multiple scale levels. The second component is used to upsample the activations at different depths of the network and produce edge maps at scale levels that share the same spatial resolution as the original input image.

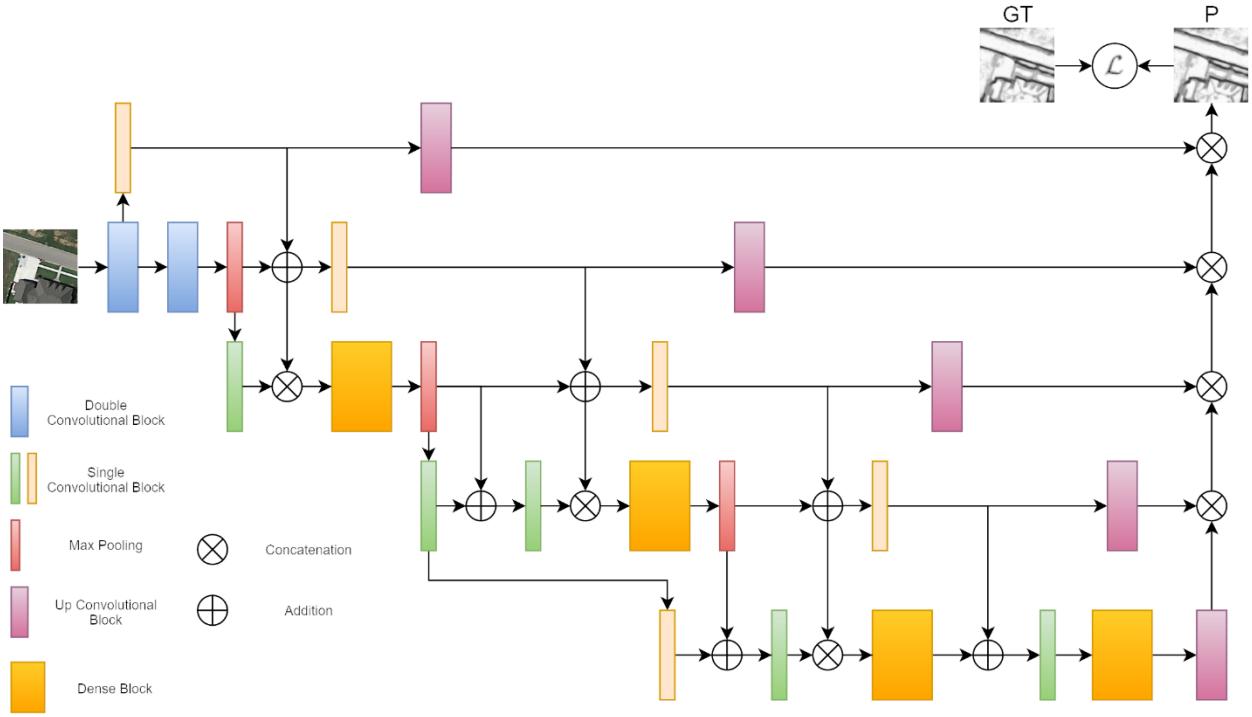



*Figure 4.3: DexiNed architecture. The diagram is a more detailed way to represent the same architecture as Soria et al. (2020) presented in their original paper, with an emphasis on the flow of information through the network based on the pytorch github code referenced by Soria et al. (2020)*[7].

A detail not immediately evident in the presentation of DexiNed (Soria et al., 2020) is how the activations of certain single convolutional blocks could be added to the activations of other blocks that did not (at first glance) share the same spatial resolution. It turns out that DexiNed uses two different ways to halve the width and height of the feature maps. The first one is the typical max pooling layer with a stride of 2, and the second passes the output of the previous layer through a single convolutional layer with a kernel size of 1 and a stride of 2. Thus, the single convolutional layers depicted in Figure 4.3 might either halve the spatial dimensions of

[7] https://github.com/xavysp/DexiNed/tree/master/DexiNed-Pytorch

their input (light yellow single convolutional blocks) or retain the spatial dimensions of the input and double the number of filters using a Kernel size of 1 (green single convolutional blocks).

The upscaling is performed through the Up-convolutional blocks (Figure 4.4). Each block consists of two sub-blocks. The first is only used when the ratio between the final scale and the current scale equals 2, whereas the second one is used in all other cases (when ratios are greater than 2). The input to the block is iteratively passed through the second upsampling sub-block until the scale of the output is half the scale of the original image, at which time it is passed through subblock 1 and then concatenated to the network's output. The main difference between the two sub-blocks is the number of convolutional filters of each layer (1 for sub-block 1 and 16 for sub-block 2).

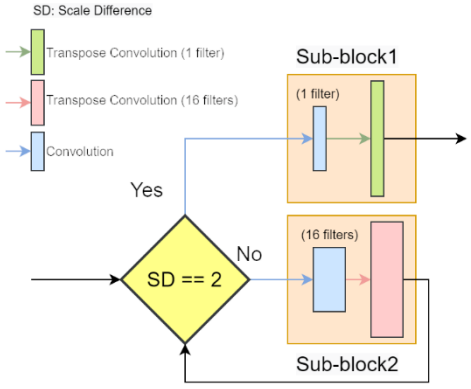



*Figure 4.4: DexiNed upsampling block.*

The edges produced by DexiNeD were inserted into two different blocks: (a) together with the RGB channels of the two instances as initial input to the network and (b) at the final convolutional block of the network. In the second case, the edges were inserted either at the beginning of the last block (Figure 5(a)), together with the feature maps from the skip connections and the upsampled feature maps, or at the final layer of the block and before the last convolution that produces the final predictions (Figure 5(b)). We have experimented with multiple different ways of inserting the edges into the UNet/ Nested UNet models:

- only at the end of the models (following Figure 4.5(a) or Figure 4.5(b)),
- only at the first convolutional block of the models

- both at the first and last convolutional block, in which case we tried both proposed approaches for the last convolutional block (Figure 4.5)

Preliminary experiments indicated that the best results were retrieved when inserting the edges both at the first convolutional block as well as the first layer of the last convolutional block of the network (Figure 4.5(a)).


*Figure 4.5: Different ways to introduce the edge feature maps into the last convolutional block of the backbone architecture.*

### 4.2.2 Datasets

In this work, we used two main datasets to train and evaluate our models and generated two additional datasets, which we will call transformed datasets:

- The Change Detection Dataset (Lebedev et al., 2018) introduced in Chapter 3, whose samples consist of a bitemporal set of RGB satellite images and a binary mask annotating the changes between them.

- An adjusted version of the ISPRS Vaihingen dataset, used only to train the DexiNeD network on extracting semantically enriched edges of aerial images. This dataset will be described in more detail in section 4.2.3.

- A rotated version of the Change Detection Dataset used for modelling the misregistration errors described in section 4.2.5.

- An affine-transformed version of the Change Detection Dataset used for modelling the misregistration errors described in section 4.2.5.

### 4.2.3 DexiNeD Training

Our initial approach was to use the pretrained version of DexiNeD to extract the features to be introduced into our Change Detection architecture. The network was shown to generalize well with different datasets (Soria et al., 2020), but was never used to predict edges from satellite imagery. The initial results of the network were discouraging, as the results retrieved from the edge-enhanced architectures using the pretrained version of DexiNeD were relatively similar or worse than the results retrieved from the original UNet and UNet++ architectures. There was also a stronger tendency for the network to overfit to the training data, especially when no data augmentation was used. Upon closer examination of DexiNeD's results, we identified one potential cause for this behavior: the network was producing strong responses when dealing with shadows (non-semantic edges), which numerically have a strong gradient response but carry no useful semantic information for the Change Detection task (Figure 4.6). This example illustrates the usefulness of using edges that carry task-specific semantic information, such as boundaries between objects belonging to different classes, that can be later used to identify land cover changes relative to the change detection task.

*Figure 4.6: Example of strong edge responses due to shadows. The image does not convey the required semantic information. Responses near edges caused by shadows are stronger than those near the boundaries of buildings or roads.*

Therefore, we retrained our Edge Detector using the ISPRS Vaihingen 2D semantic labeling benchmark Dataset[8] (Cramer, 2010) so that the network could learn useful semantic information relating to land cover semantic segmentation and consequently produce semantically meaningful

[8] https://www.isprs.org/education/benchmarks/UrbanSemLab/2d-sem-label-vaihingen.aspx

edges. The training subset of the dataset consists of 16 patches of different sizes (about 2500 × 2000 pixels) extracted from a true orthophoto mosaic generated from aerial digital images with an approximate ground resolution of 9 cm. The ground truth labeling classifies the image regions into 6 categories: impervious surfaces, buildings, low vegetation, trees, cars, and clutter/background (a tile and its ground truth labels are presented as an example in Figures 4.7 and 4.8). There is also a Digital Surface Model (DSM) available, which was not used in this study. Using the ground truth labels of the dataset, we can extract an edge map with semantically rich edges which correspond to object boundaries (Figure 4.9), by first identifying the homogeneous regions in the mask and then extracting the boundaries of each region and creating a new image that contains only the boundaries between neighboring regions. We can thus create a semantically informed edge detection dataset that only considers the boundaries between different classes and not all image edges to train the DexiNeD model on boundary detection from satellite imagery.

Our target was to create training samples that have the same dimensions as the images used in the Change Detection Dataset (256 × 256 pixels). The network also needed to be able to generalize to multiple scales, so we created additional images (and labelled masks) at scales= [0.6, 0.8] of our original scales and added those images to the dataset. We then divided each image into tiles and randomly picked 5 image centres from each tile of every image, excluding a zone of about 50 pixels so that our new images would not include any blank regions resulting from the rotation in the following step. Each of the five images in every tile was rotated (along with their corresponding labelled masks) by an angle randomly chosen from the corresponding intervals [0º,72 º), [72 º,144 º), [144 º,216 º), [216 º,288 º), [288 º,360 º) to ensure significant diversity in the training set. Using this data augmentation approach, we ended up with a dataset consisting of 6856 images (Figures 4.10 to 4.12) that we used to train DexiNeD.

Figure 4.13 presents multiple image samples of the constructed boundary detection dataset, chosen to showcase the use of different scales and orientations when sampling image patches from the images of the original Vaihingen dataset.

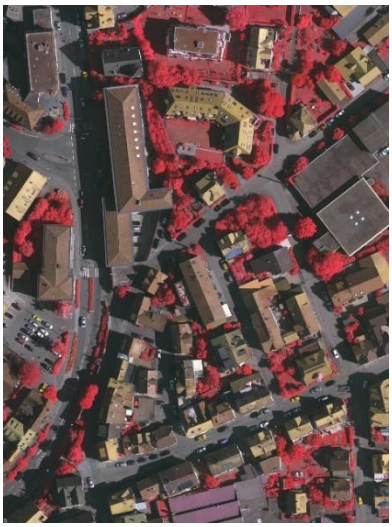


*Figure 4.7 Example image of the ISPRS Vaihingen 2D semantic labeling dataset.*

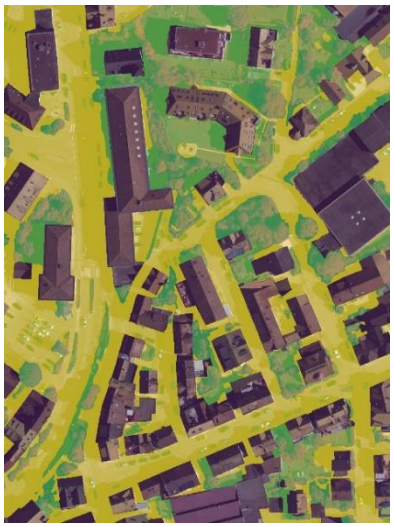


*Figure 4.8: Example image overlaid by labelled regions.*

*Figure 4.9: Edge Extraction based on labelled regions boundaries.*

*Figure 4.10: 256×256 pixels training sample.*

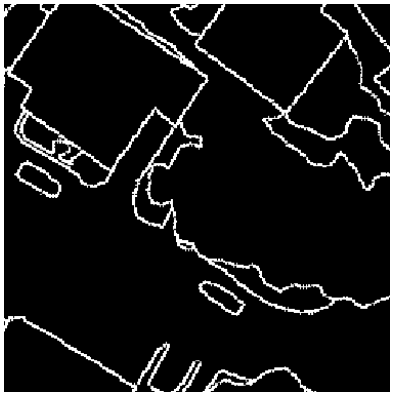


*Figure 4.11: Edges for the Figure 10 sample.*

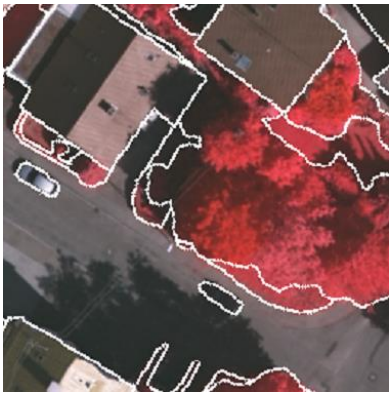


*Figure 4.12: Edges superimposed over the image.*

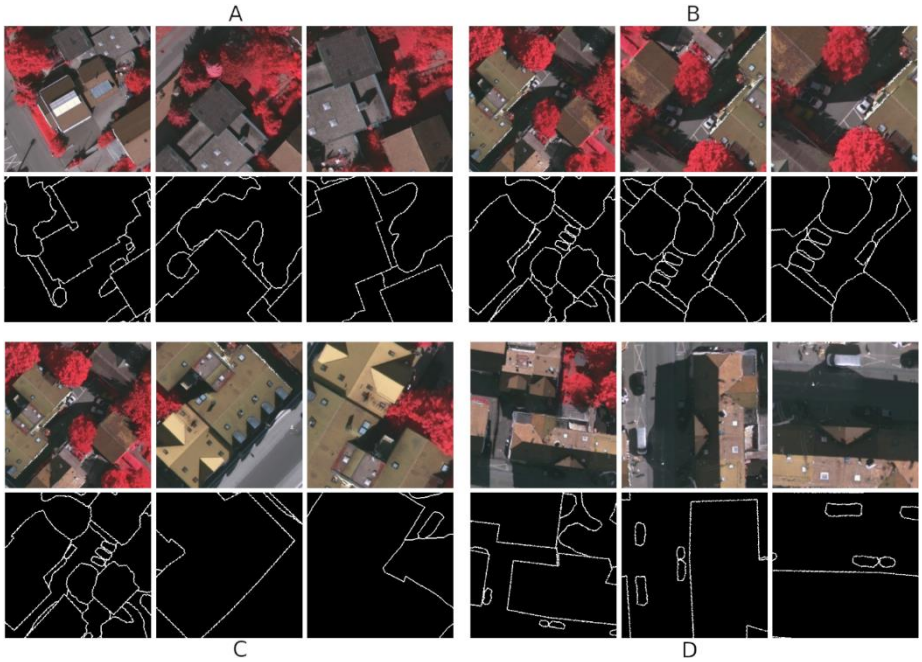



*Figure 4.13: Training samples from the boundary detection dataset. The samples have been chosen to showcase the use of different scales and orientations (for 4 different regions - quadrants A, B, C, D) when sampling from the orthophotos of the Vaihingen dataset.*

The Vaihingen dataset has a different spatial resolution (9cm) compared to our CD dataset (30 to 100cm) which raises the question of whether an airborne dataset of a different resolution should be used to train a CNN to detect edges on a separate dataset consisting of satellite images. In our view, there are two main reasons supporting the use of a dataset of different spatial resolution to train our edge detection network:

- Firstly, DexiNeD, as with all the CNN approaches for edge detection discussed in the paper, extracts feature maps at multiple resolutions that are then merged into the final predicted edge map. As shown in Figure 3, max-pooling layers gradually reduce the resolution to capture semantic information (and edges) of different granularity. Thus, as long as the resolution of the dataset used for training is higher than the resolution of the CD images, one of the deeper blocks of the network will produce activations at a resolution similar to the change detection dataset (giving the necessary “resolution overlap”).

- Secondly, while producing the edge detection dataset from the Vaihingen imagery and masks, we have used three different scales for our images (1.0, 0.8, and 0.6 of the original scale) to try and make our edge detection model more robust to scale differences.

To support our hypothesis, we have compared the CD results retrieved using a version of DexiNeD pre-trained on the general-purpose Barcelona Images for Perceptual Edge Detection (BIPED) dataset (Soria et al., 2020), to the ones retrieved when using edges from a DexiNeD model trained on our modification of the Vaihingen dataset. The results are presented in Appendix B and indicate a small improvement when using semantic edges.

### 4.2.4 Modelling of Misregistration Errors in Image Change Detection

Following these steps, we introduce a process to improve the robustness of an encoder-decoder model when dealing with misregistration errors between the instances of a bitemporal image pair. By introducing misregistration errors into training image pairs while keeping the corresponding ground truth mask the same, the model begins to implicitly learn to ignore such errors and, using the multi-resolution feature maps of the CNN-based architecture, relies more on a wider neighbourhood around each pixel when predicting each pixel's class ("Change" or "No Change"). We introduce misregistration errors into the image pair by applying a random transformation, either a pure rotation or an affine transformation, on one of the two instances of the image pair. More specifically, we keep the first instance of each image pair constant and add misregistration errors to the second instance of the pair while keeping the original "change"- "no change" mask as is. Depending on the type of the applied transformation (affine or rotation), we create two distinct training sets and their corresponding validation and test sets.

### 4.2.5 Creation of the affine and the rotated dataset (transformed datasets)

In order to apply a transformation to one of the two instances of an image pair, we need to address the blank regions created near the corners of the transformed image during image warping. One way to overcome this would be to crop the image and then rescale it, along with the second image of the pair and the ground truth mask. However, this would also affect the scale of the image pair and impede efforts to keep all other parameters constant (as much as possible) in order to be able to confidently extract conclusions on the effects of misregistration errors. However, as the dataset we are using (Lebedev et al., 2018) was created by sampling regions (let us call them subset images) from a set of bigger satellite image pairs, we could use the parent images to capture a larger region around each subset image, perform a random transformation on

that region, and then crop it to the original image size while maintaining the same image centre. Finding the parent image does present some challenges as the subset images have also been submitted to a Euclidean transformation and Lebedev et al. (2018) do not provide information on the parent image of each sample nor its transformation matrix. In order to retrieve the necessary information, we developed an algorithm to match each sampled image to its parent and estimate the transformation between the two images. Given the transformation from image $C$ to its parent $P$, $A_{CP}$, we can find the initial region in the parent image corresponding to $C$ and expand it in all directions equally to maintain the same centre. Then we apply a random transformation, $N$, to this expanded image, apply the inverse transformation $A_{PC} = A_{CP}^{-1}$ to get the subset image on its initial orientation and finally keep only a 256×256 pixels cropped window with the same centre as the original subset image. Here it should be noted that the $A_{CP}$ and $A_{PC}$ transformations should also be adjusted for a change in the coordinate system of the image, as we want the transformation $N$ to be applied relative to the centre of the subset image and not the original one.

```
## Find Parent Image and Transformation
for each image in parent_images:
    detect salient points and descriptors using ORB
    store them in Parent_Image_Features (on an image to image basis)

for each image in dataset_images:
    detect salient points and descriptors using ORB

    for each image in Parent_Image_Features:
        find matches between the two images (Use Ratio Test as refinement)
        use RANSAC to compute Euclidean transform and refine matches

    select transform that has more than 5 inliers and the lowest mean residual

    if mean residual < 5 pixel:
        save transform
```

*Algorithm 4.1: Pseudocode for image matching and estimation of the transformation between the subset image and its parent image.*

In order to match the images and retrieve the affine transformation that connects each training image to its parent image, we have developed a pipeline that first uses the ORB[9] algorithm (Rublee et al., 2011) to detect salient points and their descriptors in both the sample images and

[9] Oriented FAST and Rotated BRIEF where FAST (Features from Accelerated Segment Test) is a method for finding keypoints in real time systems (Rosten & Drummond, 2006) and BRIEF (Binary Robust Independent Elementary Features) is a feature descriptor algorithm (Calonder et al., 2010).

the parent images. It then tries to match the retrieved keypoints for each sample image to the ones of its parent image. This way, we get both the parent image and the affine transformation that links each subset image in the training/validation/test set to its parent. We use the Random Sample Consensus (RANSAC) algorithm (Fischler & Bolles, 1981) to estimate the transformation for each subset image and consider a subset image matched to its parent image when the transformation between the two has the smallest residual error amongst all the other parent images and RANSAC returns more than 5 inlier points and mean residual error of less than 5 pixels. A pseudocode of the algorithm that was used is shown in Algorithm 4.1. Using this code, we were able to match 3678 images out of the 10000 images of the training set, 1064 images in the validation set and 1037 images in the test set.

We used the produced matches on each part of the dataset (training/validation/test) to produce two new datasets, one with a random rotation of up to 10 degrees and one with an affine transformation that can be decomposed into two scales $(s_x, s_y)$, a rotation angle $\theta$, a translation vector $(t_x, t_y)$ and a skew angle $\varphi$. For the affine transformed data, we randomly selected values from an interval of [0.95, 1.05] for the scales $(s_x, s_y)$, from an interval of $[-6^o, +6^o]$ for the angle $\theta$, from an interval of $[-5^o, +5^o]$ for the skew angle $\varphi$ and from an interval of $[-2, +2]$ pixel for the translation vector $(t_x, t_y)$.

### 4.2.6 Training Description

Section 4.2.6.1 presents the loss functions used to train the Change Detection networks, UNet and UNet++, and the edge detection network DexiNeD. Section 4.2.6.2 provides information about the settings of certain training hyperparameters and the use of specific training techniques. It should be noted that the training details corresponding to the CD networks apply to the training of models on all three change detection datasets (original, rotated and affine). When specific settings apply only to the rotated and affine datasets, it is explicitly noted in the text.

#### 4.2.6.1 Loss Functions

Similar to Chapter 3, for training the models, we experimented with two different loss functions when training **the Change Detection models UNet and UNet++:** a combination of the Binary Cross Entropy loss with the Dice Coefficient (BCE Dice Loss) (Equation 3.2), which was also used in the original UNet++ paper (Z. Zhou et al., 2018), and the Lovász Hinge Loss function

(Berman et al., 2018). Both functions were designed for semantic segmentation tasks, and both BCE Dice (Z. Zhou et al., 2018; D. Peng et al., 2019; Bousias Alexakis & Armenakis, 2020) and Lovász Hinge loss (Rakhlin et al., 2018; Bousias Alexakis & Armenakis, 2020) can produce high-quality results.

Regarding the **Edge Detection component of the Network (DexiNeD)**, Soria et al. (2020) apply the loss function used on the HED network, which is a weighted cross entropy loss (Equation 4.1) for each side output $l_i$ that corresponds to a different hierarchical level of the network. In Equation 5, $y$ is the class of each image pixel, $\hat{y}_{c=1}$ is the output of the network describing the probability that a pixel belongs to class1 (edge), and $\hat{y}_{c=0}$ is the output of the network describing the probability that a pixel belongs to class=0 (not an edge). $Y^+$is the set of all class=1 pixels (edges) and $Y^-$ is the set of all class=0 pixels, while $\beta$ is the ratio of the number of pixels belonging to class=1 over the total number of pixels in the ground truth mask. Following the deep supervision paradigm, the complete loss function is a fusion of all the side outputs (Equation 4.2), where $m_i$ are trainable parameters. Deep Supervision is a training method that introduces multiple "companion" loss functions, each linked to outputs at a different point of the neural network that are then integrated together with the final output of the network into one main loss function used for training (Lee et al., 2015).

$$l_i(y, \hat{y}) = -\beta \sum_{y \in Y^+} \log\left(\hat{y}_{c=1}\right) - (1-\beta) \sum_{y \in Y^-} \log\left(\hat{y}_{c=0}\right) \quad (4.1)$$

$$L = \sum_{i=1}^{5} m_i l_i \quad (4.2)$$

#### 4.2.6.2 Training details

The backbone architectures have been trained on the original training dataset for 260 epochs each with a 0.0003 learning rate for the first 200 epochs and 0.0001 learning rate for the rest 60 epochs. For all Change Detection models, the training was performed using the Adam optimizer with the default parameters (0.9 and 0.999) for the coefficients of the running average and without weight decay. We have also used a simple form of data augmentation (image flips), which proved to be very effective and significantly increased the performance of the edge-

enhanced architectures[10]. Data augmentation also helped address any overfitting on the training dataset (Figure 4.14). Differences between the training and validation Intersection over Union (IoU) values are very small, which indicates an almost negligible overfitting effect. Thus, we have not used any other method to reduce overfitting, such as Dropout or an additional regularization term on the loss function.

We have also made use of the Deep Supervision (DS) concept when training the plain (with no added boundaries) version of the UNet++ models to compare with the rest of the UNet++ models. In the UNet++ case, companion loss functions (in our case either BCE Dice Loss or Lovasz Hinge Loss) are computed for the outputs of the convolutional blocks $X_{11}, X_{12}, X_{13}, X_{14}$ and, together with $X_{15}$, they are then aggregated into the final loss function that is used to train the neural network.

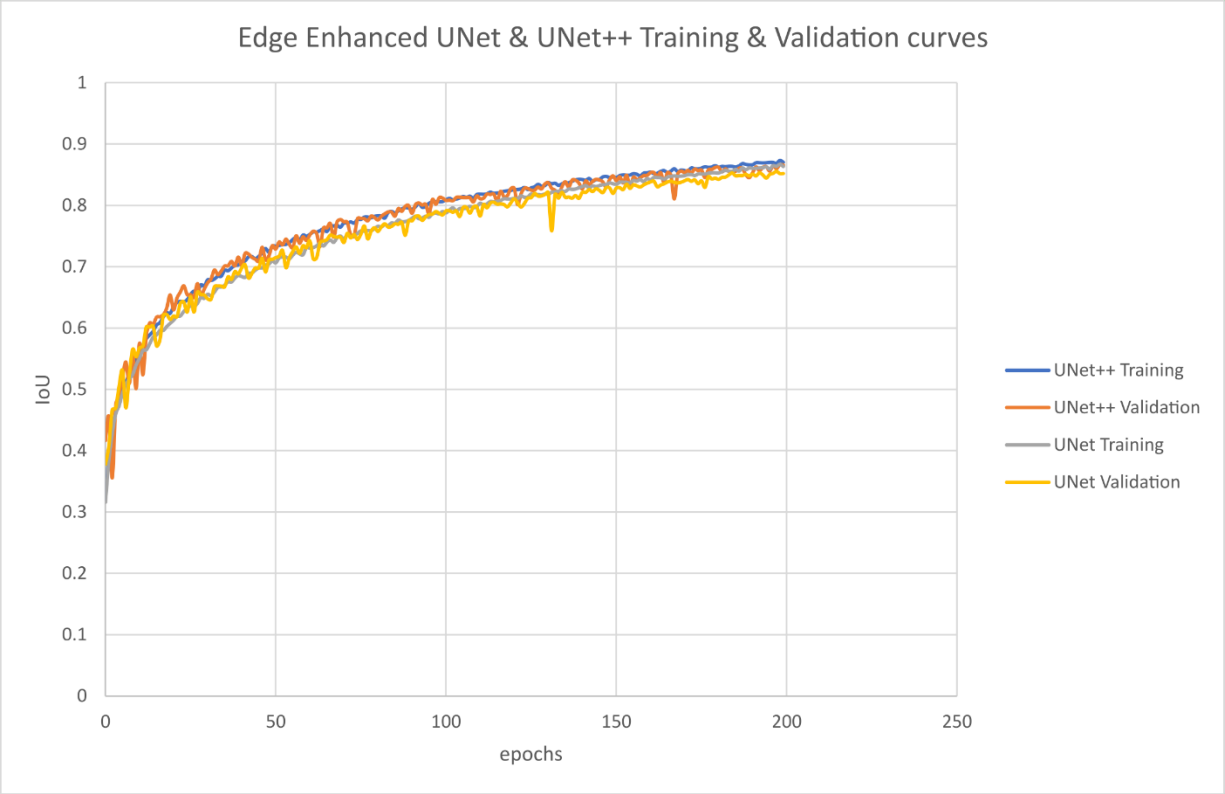



*Figure 4.14: IoU training curves for edge-enhanced UNet and UNet++ networks for the first 200 training epochs. The differences between the training and validation IoU values are less than 1%. Similar training behaviour was observed for all training models when trained using data augmentation.*

[10] We have used parts of https://github.com/4uiiurz1/pytorch-nested-unet and https://github.com/bermanmaxim/LovaszSoftmax repositories to develop our code.

The DexiNeD network has been trained for 30 epochs on our modified version of the Vaihingen ISPRS network using the original paper's code[11] and its default training parameters, with minor modifications when importing our dataset into the network and removing image rescaling to maintain images of size 256×256 pixels. We followed a transfer learning scheme and used the DexiNeD pre-trained model on the Barcelona Images for Perceptual Edge Detection (BIPED) dataset (Soria et al., 2020), a general-purpose edge detection dataset that consists of 250 outdoor images and their manually annotated edge masks of size 1280×720 pixels. The Edge Enhanced Networks have been trained from scratch on the Change Detection Dataset using the pretrained DexiNeD model. It should be noted that only the backbone network parameters (UNet or UNet++) were updated during the training phase of the Edge Enhanced Networks.

For the misregistration change detection model, training was performed using the rotated and affine dataset using networks that were already trained for 200 epochs on the original dataset. The models were trained for an extra 150 epochs on the rotated or affine dataset using the Adam optimizer. During the training for all the aforementioned models, we applied an extra convergence condition to stop training if the loss function on the validation set was not reduced after more than 20 epochs.

## 4.3. Results and discussion

We evaluate the results based on the average precision, recall, F1 score (Equations 3.6 to 3.8) and Intersection over Union (IoU) (Equation 4.3) metrics over the entire test set. We have also included accuracy (Equation 3.9) for the sake of completeness. However, as accuracy can be a misleading metric in datasets with highly imbalanced data such as ours, where the number of pixels classified as changed is much lower than the number of pixels classified as unchanged, we are not considering it in our analysis. The formulas of precision, recall, F1 and IoU do not include the number of True Negative pixels, which makes them good performance indicators of image segmentation for an imbalanced dataset where the unchanged areas (TN and FN) are more common than the changed ones (TP and FP). The batch size used for testing was set to 1.

[11] https://github.com/xavysp/DexiNed/tree/master/DexiNed-Pytorch

$$IoU = \frac{TP}{TP + FN + FP} \tag{4.3}$$

Figure 4.15 provides a high-level overview of the training and evaluation pipeline followed for each model, with an emphasis given on the use of different datasets (training, validation, test) at different process stages. The presented example focuses on networks with incorporated boundary information, but the same process has been followed when evaluating each of the backbone architectures. In this case, the greyed module on top of the chart indicates simply a choice between plain UNet and UNet++ architectures that are then fed into the following modules (green) for training.

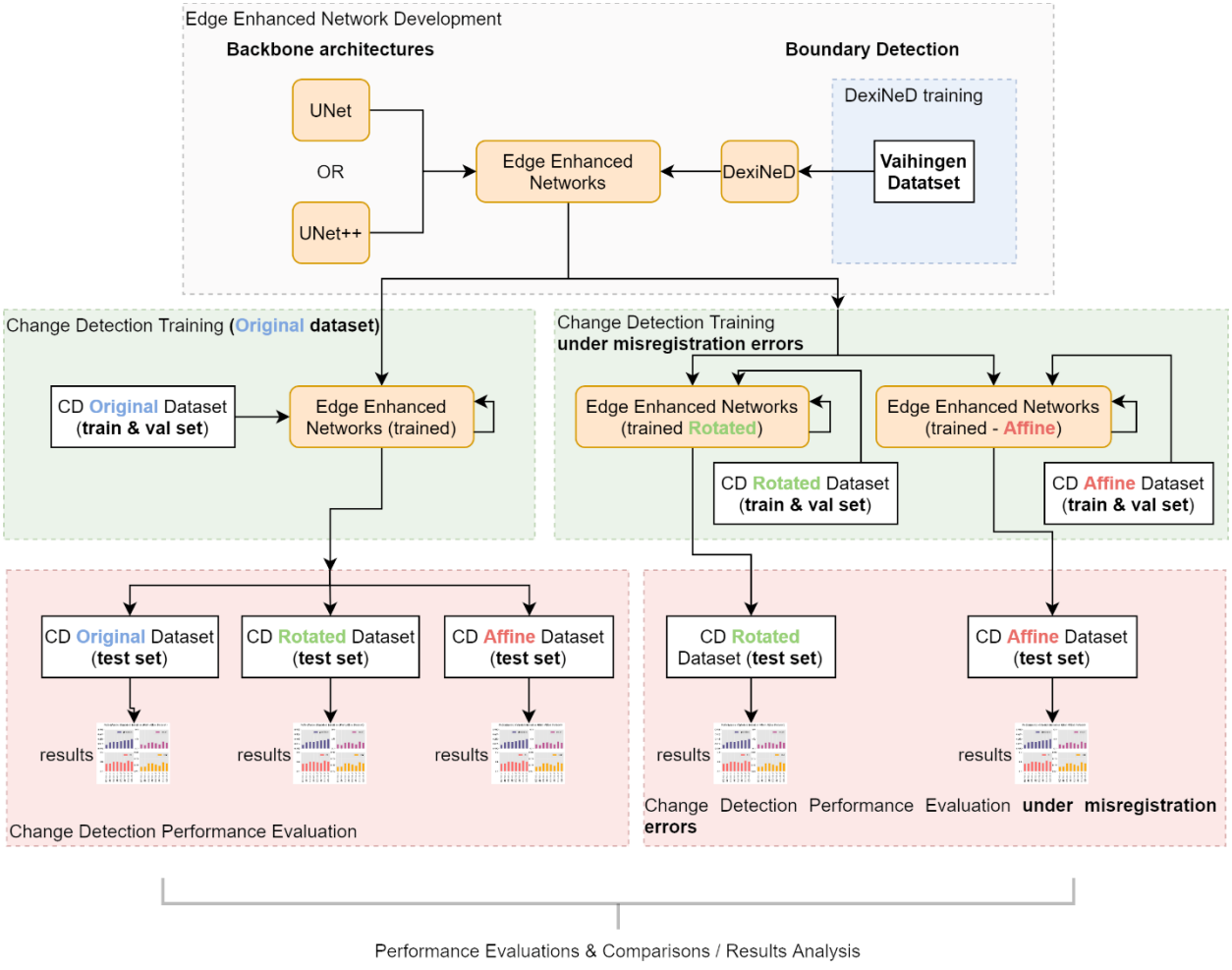



*Figure 4.15: High-level methodology overview with an emphasis on clarifying the use of each dataset for training and evaluation of the models summarizing the different training phases of the models and the use of the different networks.*

In order to evaluate the potential improvements of our approach on the models, we have organized the results into two main sections. In the first, we focus on the effects of the introduction of edges on each model (including models trained and/or tested on the transformed datasets). In the second, we focus on the effects of the training scheme using added misregistration errors on the performance of each network.

### 4.3.1 Effects of the introduction of edges into the Encoder-Decoder Architectures

We examine the effects of incorporating edges into both the original dataset, as well as the datasets that include mis-registrations between the instances of image pairs (affine and rotated datasets). The rotated and affine datasets (transformed datasets) are presented in Section 3.3 and contain the same number of training, validation and test samples as the original dataset (10,000, 3,000 and 3,000, respectively). All metrics presented in the figures and tables in Section 4 are based on the test set component of each dataset. In all figures, the following nomenclature is used to distinguish between models trained using different hyperparameters:

- For models trained with BCE Dice Loss, we include the BCE abbreviation in their name, while LOV is used with models trained using the Lovász Hinge Loss.
- For models trained with added boundaries information, the AB abbreviation has been used, while the NB abbreviation indicates models trained using the original architectures.
- We use UNet and UNetPP (for UNet++) to define the model's backbone architecture.
- When testing the models using the test set of the rotated or affine dataset, we distinguish between models trained on the affine training set by including "af" at the end of the model's name, "rot" when the model was trained on the rotated training set, and "or" when the model was trained on the original dataset.
- Models trained using Deep Supervision are indicated with the letters DS.
- When a hyperparameter is implied by the figure's title or caption, we omit it from the model's name for the sake of brevity.

Figure 4.16 (a) and (b), as well as Table 4.1, present the performance of models with or without the introduction of boundary features into the network for the UNet++ and the UNet architectures. In addition, we present results from experiments using both loss functions (BCE and LOV). For the NB models using the UNet++ architecture, we also used Deep Supervision, which incorporates outputs from different parts of the network into the loss function resulting sometimes in small performance improvements (Lee et al., 2015; D. Peng et al., 2019; Z. Zhou et al., 2018). The best precision, F1 score and IoU values were retrieved when using the UNet++ architecture enhanced with boundary information from DexiNeD and trained using the Lovász Hinge Loss function. An improvement in all 5 metrics was found with the UNet++ architecture when edges were introduced, with the highest increases noted on the recall rate (~2%) and the IoU metric (around 2.5% for BCE Dice Loss and 2.2% for Lovász Hinge Loss). Models based on the UNet architecture show similar improvements on all metrics, with the benefits on recall, F1 score, and IoU being close to or higher than 2% for both loss functions.

*Table 4.1: Performance metrics on the original dataset test set with and without added boundary information.*

| Arch | Loss Function | Boundaries | Deep Supervision | Train Dataset | Test Dataset | Accuracy | Precision | Recall | F1 | IoU |
|---|---|---|---|---|---|---|---|---|---|---|
| UNet++ | BCE DICE | ✗ | ✗ | original | original | 0.9872 | 0.9406 | 0.8129 | 0.8561 | 0.7788 |
| UNet++ | BCE DICE | ✗ | ✓ | original | original | 0.9857 | 0.9259 | 0.8054 | 0.8446 | 0.7626 |
| UNet++ | BCE DICE | ✓ | ✗ | original | original | 0.9893 | 0.9509 | **0.8332** | 0.8737 | 0.8035 |
| UNet++ | LOVÁSZ | ✗ | ✗ | original | original | 0.9874 | 0.9519 | 0.8124 | 0.8586 | 0.7837 |
| UNet++ | LOVÁSZ | ✗ | ✓ | original | original | 0.9872 | 0.9428 | 0.8191 | 0.8620 | 0.7858 |
| UNet++ | LOVÁSZ | ✓ | ✗ | original | original | 0.9892 | **0.9560** | 0.8329 | **0.8758** | **0.8059** |
| UNet | BCE DICE | ✗ | ✗ | original | original | 0.9859 | 0.9273 | 0.7956 | 0.8377 | 0.7549 |
| UNet | BCE DICE | ✓ | ✗ | original | original | 0.9877 | 0.9423 | 0.8142 | 0.8566 | 0.7803 |
| UNet | LOVÁSZ | ✗ | ✗ | original | original | 0.9857 | 0.9391 | 0.7898 | 0.8388 | 0.7567 |
| UNet | LOVÁSZ | ✓ | ✗ | original | original | 0.9877 | 0.9435 | 0.8166 | 0.8583 | 0.7829 |

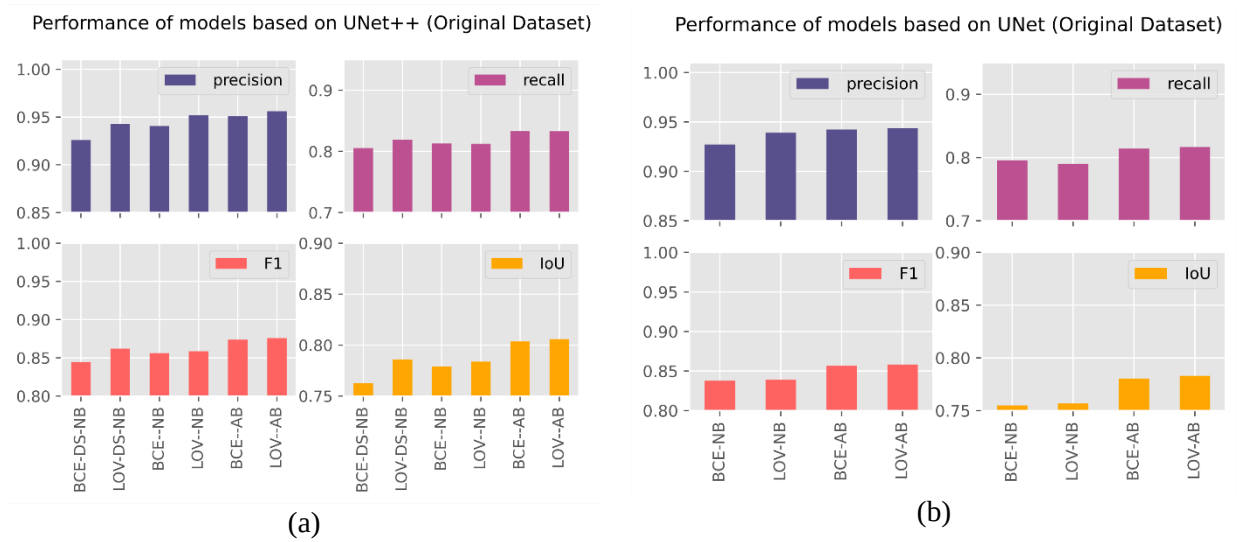



*Figure 4.16: Performance metrics for models trained using UNet++ (a) and UNet (b) as our backbone architecture (where BCE refers to BCE-Dice Loss; LOV to Lovász Hinge Loss; DS to Deep Supervision; AB to Added Boundary Information; NB to No Boundary Input).*

Similarly, in Figure 4.17 (a) and (b), and in Table 4.2, we present the performance metrics values produced on the affine test set for the same models when trained on the affine or the original datasets. The best performance in all four metrics was found by using UNet++ with BCE Dice Loss and Boundaries and training with the affine training set. When considering only the models trained using UNet, the best results on recall, F1 score and IoU are retrieved once again with the BCE Dice Loss and Boundaries with training on the affine training set (precision~91%, recall~79%, F1~83% and IoU~74.5%). A careful examination of the results has led us to the following two main observations/conclusions. First, when comparing the same models with all hyperparameters constant and varying only whether semantically informed edges are used, we observe an improvement on all metric values of the boundary-enhanced models compared to the corresponding original models, except for UNet ++ when trained using the Lovász Hinge Loss on the affine dataset. An almost negligible reduction of 0.1% on recall was obtained in this case. For all the other models, there is an average improvement of about 2% on the IoU, 1.7% on the recall and F1 score and 1% on the precision metric. Second, when the models are trained on the affine training set, they perform better than the same models trained on the original training set. This point will be further investigated in the next part of this analysis.

*Table 4.2: Performance metrics on the affine dataset test set for models with and without added boundary information, trained on either the original or affine training set.*

| Arch | Loss Function | Boundaries | Train Dataset | Test Dataset | accuracy | precision | recall | F1 | IoU |
|---|---|---|---|---|---|---|---|---|---|
| UNet++ | BCE DICE | ✗ | original | affine | 0.9797 | 0.8697 | 0.7526 | 0.7876 | 0.6937 |
| UNet++ | BCE DICE | ✓ | original | affine | 0.9828 | 0.8928 | 0.7760 | 0.8129 | 0.7277 |
| UNet++ | BCE DICE | ✗ | affine | affine | 0.9816 | 0.8860 | 0.7713 | 0.8061 | 0.7146 |
| UNet++ | BCE DICE | ✓ | affine | affine | **0.9843** | **0.9123** | **0.7898** | **0.8300** | **0.7460** |
| UNet++ | LOVÁSZ | ✗ | original | affine | 0.9811 | 0.8873 | 0.7635 | 0.8027 | 0.7129 |
| UNet++ | LOVÁSZ | ✓ | original | affine | 0.9826 | 0.8993 | 0.7746 | 0.8146 | 0.7293 |
| UNet++ | LOVÁSZ | ✗ | affine | affine | 0.9830 | 0.9088 | 0.7817 | 0.8216 | 0.7359 |
| UNet++ | LOVÁSZ | ✓ | affine | affine | 0.9836 | 0.9108 | 0.7803 | 0.8216 | 0.7363 |
| UNet | BCE DICE | ✗ | original | affine | 0.9794 | 0.8692 | 0.7427 | 0.7792 | 0.6847 |
| UNet | BCE DICE | ✓ | original | affine | 0.9818 | 0.8851 | 0.7626 | 0.8009 | 0.7120 |
| UNet | BCE DICE | ✗ | affine | affine | 0.9817 | 0.8888 | 0.7548 | 0.7946 | 0.7027 |
| UNet | BCE DICE | ✓ | affine | affine | 0.9828 | 0.8948 | 0.7726 | 0.8106 | 0.7212 |
| UNet | LOVÁSZ | ✗ | original | affine | 0.9796 | 0.8823 | 0.7364 | 0.7811 | 0.6873 |
| UNet | LOVÁSZ | ✓ | original | affine | 0.9816 | 0.8841 | 0.7645 | 0.8015 | 0.7129 |
| UNet | LOVÁSZ | ✗ | affine | affine | 0.9806 | 0.8926 | 0.7422 | 0.7870 | 0.6936 |
| UNet | LOVÁSZ | ✓ | affine | affine | 0.9821 | 0.8997 | 0.7619 | 0.8025 | 0.7123 |

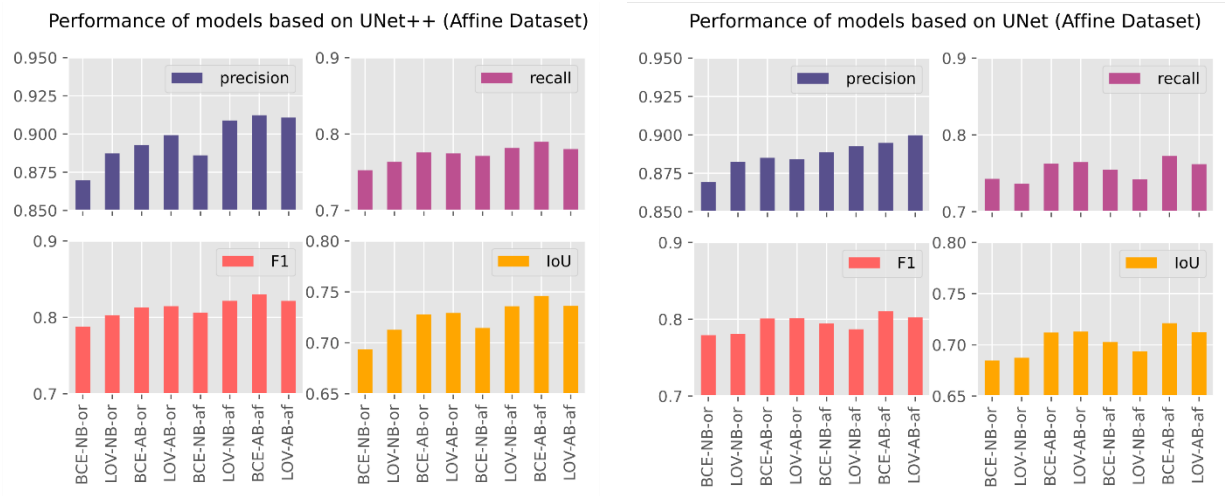


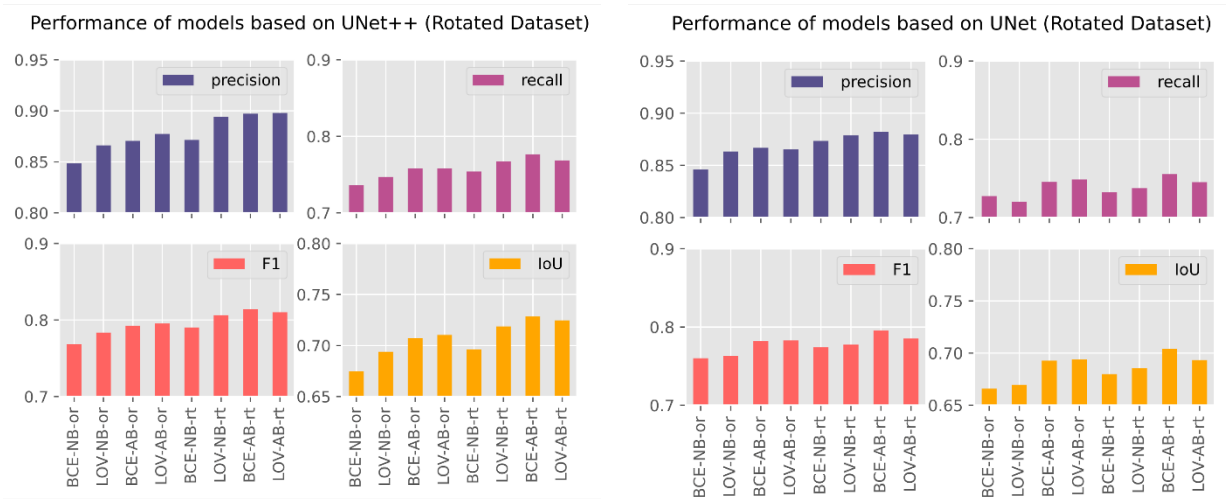

*Figure 4.17: Performance metrics for models trained using UNet++ (a) and UNet (b) as a backbone architecture and applied on the affine test set. The models were either trained on the original (or) or on the affine (af) training set. (Where BCE refers to BCE-Dice Loss; LOV to Lovász Hinge Loss; DS to Deep Supervision; AB to Added Boundary Information; NB to No Boundary Input).*

The same process was applied to the rotated dataset (Figure 4.18 (a) and (b) and Table 4.3), yielding similar results. Once again, UNet++ with additional boundary information injected into the first and last convolutional blocks of the network trained with BCE Dice Loss on the rotated

training set performed the best on almost every metric (accuracy≈98.3%, recall≈77.5%, F1≈81.4% and IoU≈72.8%). The same model trained with Lovász Hinge Loss on the rotated training set had the highest precision rate by a small margin (about 89.8% compared to the former model's 89.7%). The two main trends that were identified in the affine dataset analysis are also present in the rotated test set case, with all models tested on the rotated set showing improved performance metrics, averaging approximately 2% for the IoU, 1.7% for recall and F1 score and 1% for precision when boundary information is incorporated. All models performed better on the rotated test set when trained on the rotated training set. Thus, the effect of the proposed improvements is consistently beneficial across the examined performance metrics. This supports our initial hypothesis that the introduction of semantically informed edges can help the model perform more accurately on the semantic segmentation task.

*Table 4.3: Performance metrics on the rotated dataset test set for models with and without added boundary information, trained on either the original or the rotated training set.*

| Arch | Loss Function | Boundaries | Train Dataset | Test Dataset | accuracy | precision | recall | F1 | IoU |
|---|---|---|---|---|---|---|---|---|---|
| UNet++ | BCE DICE | ✗ | original | rotated | 0.9773 | 0.8484 | 0.7360 | 0.7682 | 0.6746 |
| UNet++ | BCE DICE | ✓ | original | rotated | 0.9800 | 0.8705 | 0.7581 | 0.7924 | 0.7072 |
| UNet++ | BCE DICE | ✗ | rotated | rotated | 0.9796 | 0.8715 | 0.7541 | 0.7901 | 0.6963 |
| UNet++ | BCE DICE | ✓ | rotated | rotated | **0.9826** | 0.8971 | **0.7764** | **0.8139** | **0.7284** |
| UNet++ | LOVÁSZ | ✗ | original | rotated | 0.9785 | 0.8661 | 0.7468 | 0.7836 | 0.6939 |
| UNet++ | LOVÁSZ | ✓ | original | rotated | 0.9801 | 0.8773 | 0.7579 | 0.7955 | 0.7104 |
| UNet++ | LOVÁSZ | ✗ | rotated | rotated | 0.9811 | 0.8942 | 0.7675 | 0.8062 | 0.7187 |
| UNet++ | LOVÁSZ | ✓ | rotated | rotated | 0.9819 | **0.8979** | 0.7685 | 0.8101 | 0.7246 |
| UNet | BCE DICE | ✗ | original | rotated | 0.9767 | 0.8462 | 0.7273 | 0.7600 | 0.6659 |
| UNet | BCE DICE | ✓ | original | rotated | 0.9793 | 0.8668 | 0.7455 | 0.7820 | 0.6927 |
| UNet | BCE DICE | ✗ | rotated | rotated | 0.9791 | 0.8734 | 0.7323 | 0.7739 | 0.6799 |
| UNet | BCE DICE | ✓ | rotated | rotated | 0.9809 | 0.8823 | 0.7554 | 0.7958 | 0.7041 |
| UNet | LOVÁSZ | ✗ | original | rotated | 0.9772 | 0.8632 | 0.7201 | 0.7628 | 0.6693 |
| UNet | LOVÁSZ | ✓ | original | rotated | 0.9790 | 0.8653 | 0.7486 | 0.7829 | 0.6940 |
| UNet | LOVÁSZ | ✗ | rotated | rotated | 0.9793 | 0.8789 | 0.7375 | 0.7776 | 0.6854 |
| UNet | LOVÁSZ | ✓ | rotated | rotated | 0.9799 | 0.8794 | 0.7452 | 0.7851 | 0.6932 |




(a) (b)

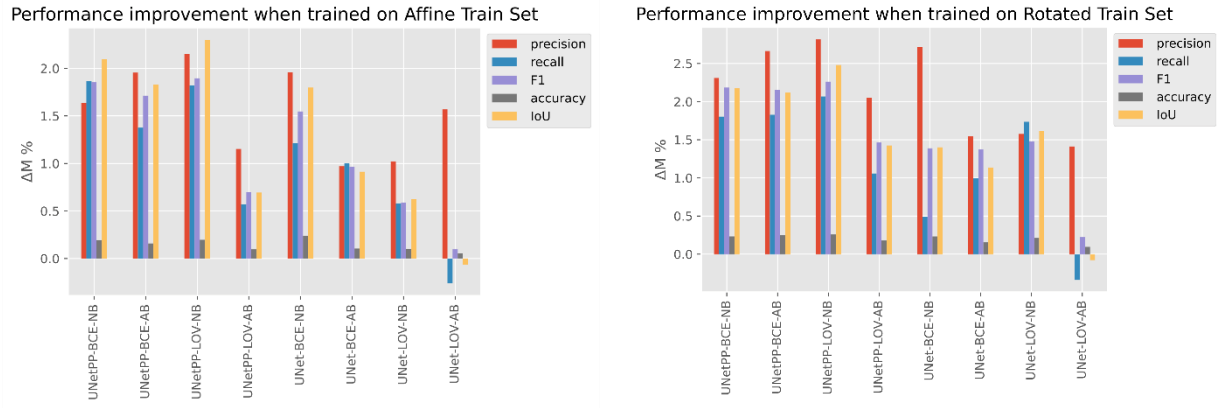
*Figure 4.18: Performance metrics for models trained using UNet++ (a) and UNet (b) as a backbone architecture and applied on the rotated test set. The models were either trained on the original (or) or on the rotated (rt) training set. (Where BCE refers to BCE-Dice Loss; LOV to Lovász Hinge Loss; DS to Deep Supervision; AB to Added Boundary Information; NB to No Boundary Input).*

### 4.3.2 Effects of training using added mis-registrations

In Figures 4.19 to 4.21, we present the effects on each change detection model's performance metrics when trained on the two training sets that include artificial mis-registrations between the two instances of an image pair. This includes the affine dataset (Figure 4.19(a), 4.20(a), 4.20(b)) that has random affine transformations applied to the second frame of the image pair, and the rotated dataset (Figure 4.19(b), 4.21(a), 4.21(b)), that rotates the second frame by a random angle of up to 10 degrees. The evaluation is performed on test sets produced similarly to the training sets: approximately 1/3 of the second period images were subjected to a random affine transformation/ rotation that simulates random misregistration errors between the instances of the affected image pairs. For each image pair, the evaluation is performed by considering the original first instance of the pair, the rotated or otherwise transformed second instance of the pair and the original change mask of the change detection dataset (since there are no extra changes introduced into the pair, only misregistration errors).

In Figures 4.19 (a) and (b), we present the results for all images from the transformed test dataset (either rotated or affine). The results suggest an overall improvement on every metric for all models, except for the one based on the UNet architecture with added boundary features trained using Lovász Hinge loss, where the precision was improved by about 1.5%. Nevertheless, the

recall was slightly reduced (less than 0.5%), and the rest of the metrics remained almost unchanged. The most significant improvements were recorded from the UNet++ architecture using Lovász Hinge loss with no boundaries added, which exceeded 2% for precision and IoU metrics for both test sets. It should be noted that the presented results reflect the magnitude of improvement on the average metrics for each model compared to its initial performance when trained using the original dataset and not its performance compared to other models on the rotated and affine test sets, which are presented in Figures 4.17 and 4.18.

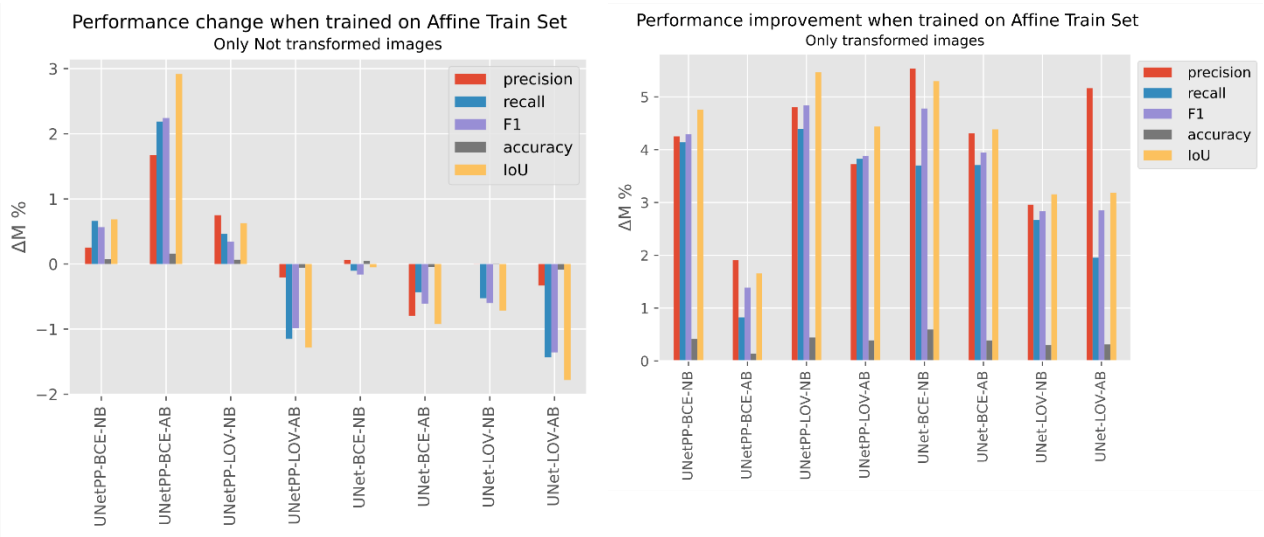



(a) (b)

*Figure 4.19: Improvements on different performance metrics retrieved on the affine (a) and rotated (b) test set when using models trained on the affine rotated training set, respectively. ΔM refers to the difference Mtr - Mor, where Mtr is any of the five metrics trained either on the affine or the rotated dataset, and Mor is the corresponding metric retrieved on the original dataset.*

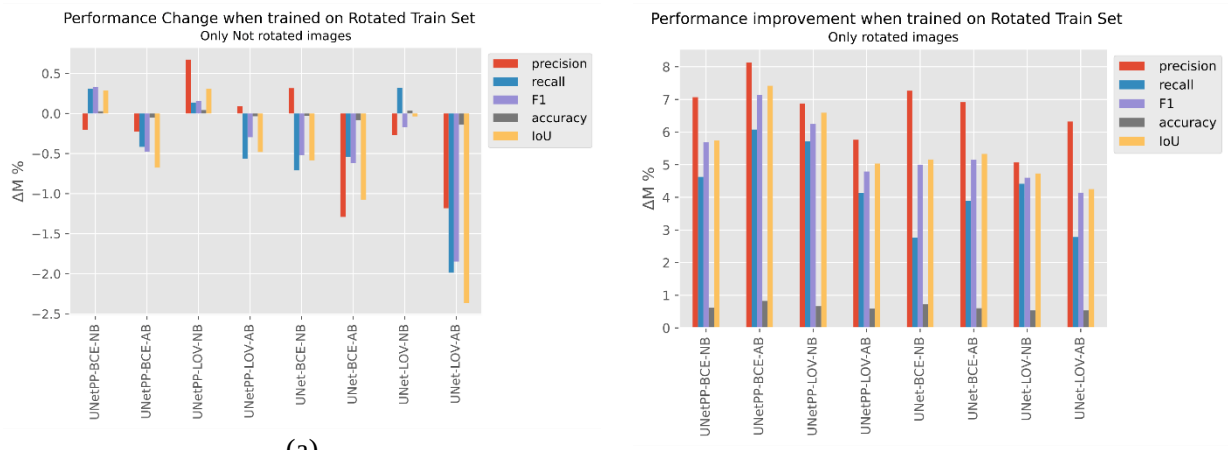
Since the affine and rotated test sets consist of both transformed (about 1/3 of the image pairs) and not transformed (about 2/3 of the image pairs) image pairs, we can divide the dataset into two distinct sets and separately evaluate the effects of training on noisy datasets for the image pairs that were subjected to a transformation (Figures 4.20 (b) and 4.21 (b)) and on the original image pairs (Figures 4.20 (a) and 4.21(a)) of the dataset. Figures 4.20 (a) and 4.21 (a) show that for the part of the test set that has not been subjected to an affine transformation (Figure 4.20(a)) or a rotation (Figure 4.21 (a)), the values of the average metrics are, in general, declining up to about 2% for the IoU metric and 1.3% for the rest, with some models (UNetPP-BCE-NB & UNetPP-LOV-NB on both datasets and UNetPP-BCE-AB tested on the affine dataset) still showing improved performance even when only considering the original image pairs. One possible explanation for the observed decline may be that, as the networks are trained on image pairs with misregistrations, they become more tolerant towards differences between corresponding pixels from the two images. Thus, they may lose a small portion of their

localization accuracy when the co-registration between the image pairs is more accurate. On the other hand, the change in the performance across all metrics shows a significant improvement in the transformed part of the dataset for all models. The increase in the precision metric reaches 8% (UNetPP-BCE-AB) for the rotated set and about 5.5% (UNet-BCE-NB) for the affine set.




(a) (b)

*Figure 4.20: Effect of using the affine training dataset on the average performance metrics considering only the images that were not (a) or that were (b) transformed in the test set.*




(a) (b)

*Figure 4.21: Effect of using **the rotated training dataset** on the average performance metrics considering only the images that were not (a) or that were (b) transformed in the test set.*

In Figure 4.22, we present multiple examples for four of the trained models in an attempt to give a general overview of the retrieved results. A qualitative assessment of the results suggests that all four models perform reasonably well on the Change Detection task, with the produced change

maps being very similar to the Ground Truth. All models are able to detect multiple small objects, such as cars (examples on rows 1, 3, 9), as well as thin and elongated elements like roads (examples on rows 2, 3, 1). They can also predict the boundaries of bigger objects such as buildings (examples on rows 5, 8, 10) with relatively high accuracy. In more demanding cases, such as a more complex system of very thin rural roads (example on row 7) or with both thin and small elements covered in snow, as in one of the instances (example on row 8), all networks misclassified parts of the elements that are more difficult to identify. However, the UNet++ models with the added edge feature maps (UNetPP-BCE-AB and UNetPP-BCE-AB-af) seem to perform better and exhibit relatively fewer misclassifications.

## 4.4. Summary

We have proposed two ways to improve the already high performance of CNN-based encoder-decoder architectures on satellite image-based Change Detection applications: the introduction of boundary information into the Network, in an attempt to improve the model's semantic segmentation accuracy, and the introduction of misregistration noise into the training image pairs, in order to make the networks more robust to the presence of misregistration errors between instances of an image pair. We have experimented using the proposed approaches on two different Network architectures (UNet and UNet++) and using two different loss functions. The experimental results suggest that, for all the tested models, our proposed approach improves the network's average prediction performance.

More specifically, the presented results suggest that the use of semantically informed edges into the models results in average improvements of about 2% on the IoU metric, 1.7% on the recall rate and F1 score and about 1% on the precision metrics, compared to the use of plain backbone architectures with no added boundaries (both for UNet and UNet++ architectures and BCE Dice and Lovasz Hinge loss functions). Therefore, the introduction of boundary information into the backbone architecture results in improved performance of the models, which agrees with our original hypothesis that this would implicitly help the network learn better representations through additional semantic information provided to the network by the boundary predictions.

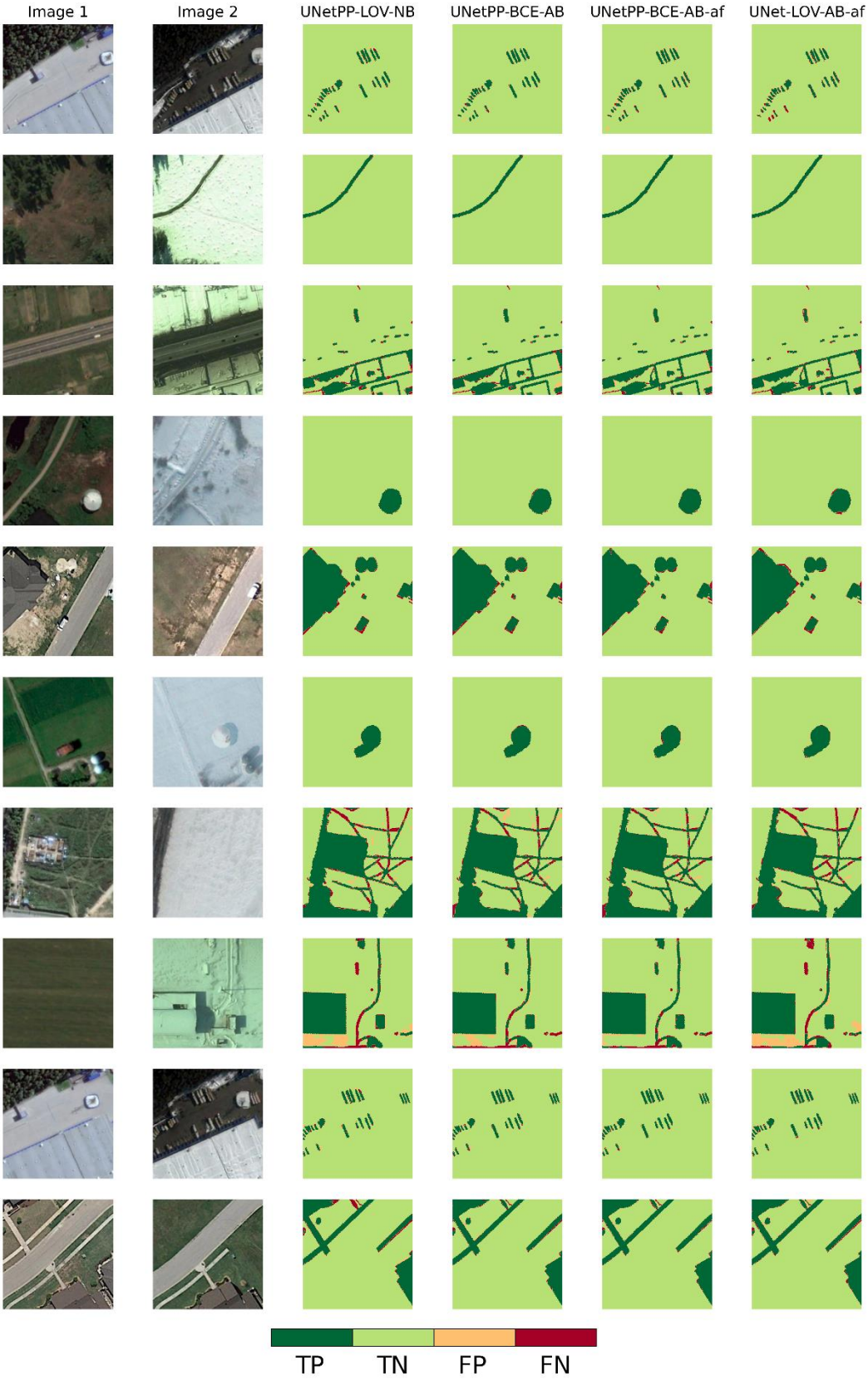



*Figure 4.22: Examples of different networks' change detection performance.*

With respect to the training process proposed to make the models more robust to misregistration errors between image pairs, our results indicate that training using the affine dataset results in average improvements of about 2% on the IoU metric, 1.7% on the recall rate and F1 score and around 1% on the precision metric. However, when solely considering image pairs with very low misregistration errors, some models (primarily those using UNet as a backbone network) trained using our approach perform worse than the models trained on the original dataset, with the average reduction in performance being around 1%. This reduction is counteracted by the significant improvements achieved when considering image pairs with higher misregistration errors, where the average improvement is about 4%. Thus, there is a trade-off to be made when training using the proposed augmentations compared to the conventional training process: improved performance in the presence of misregistration noise compared to slightly impaired performance when there are no significant co-registration errors between the instances of the image pair.

In the next chapter, we will introduce a semi-supervised training framework that addresses the lack of large, annotated change detection datasets necessary to train CNN-based change detection pipelines with high generalization capabilities reliably.

# 5

# Semi-Supervised Learning for CNN-Based Change Detection

## 5.1. Introduction

In many remote sensing applications, like land-cover and land-use classification or change detection applications from satellite imagery, there is an abundance of unlabelled training data available from sources such as Google Earth or Sentinel 2 imagery. The most challenging step for successfully training a CNN-based CD application is to create reliable annotations for a sufficiently large training dataset so that the algorithm may learn to generalize well to new images. In this chapter, we investigate a semi-supervised approach for CD from satellite imagery. The proposed approach utilizes all the additional unlabelled information by encouraging the predictions of images subjected to various transformations to remain consistent, expecting that it will lead to CD models that generalize well even when trained with a limited number of labelled examples. The objective is to reduce the need for labelled training samples by utilizing readily available unlabelled multi-temporal satellite imagery. The method is inspired by self-ensembling and the Mean Teacher training framework (Tarvainen & Valpola, 2017; X. Li et al., 2021).

The Mean Teacher training method builds on the idea of temporal ensembling (Laine & Aila, 2017), which suggests that averaging over the predictions of multiple instances of a CNN across the training process leads to more stable and more accurate results compared to solely using the output of the last model. Tarvainen and Valpola (2017) proposed averaging over consecutive models' weights instead of their predictions and utilizing the mean model as a teacher by comparing its predictions to the ones of the current model (student), thus introducing an additional loss function component. When combined with perturbations to the input images or

the models' features/weights, this extra loss component helps the model learn useful information about the data, even without training labels.

The proposed method is based on two simple ideas:

- The first is the consistency assumption, which suggests that the model's outputs should be consistent even if the input images have been subjected to a certain number of transformations. This assumption pertains to a property called transformation equivariance (X. Li et al., 2021, 2018; Cohen & Welling, 2016), and is applied by incorporating a consistency regularization term (i.e., a term that enables the predictions of the same input to remain consistent/unaffected even when subjected to various perturbations) into the loss function.

- The second is the Mean Teacher training framework, which is a form of self-ensembling, where multiple training instances of the same model are fused to produce better predictions[12]. Instead of only using our training model to compute the consistency regularization term, we compare the results of our training model (student) to the results of a mean model (teacher), whose weights are computed based on an Exponential Moving Average (EMA) of the weights of the student model throughout previous training epochs.

In Section 5.2, we present the proposed methodology for the development of the semi-supervised training process. Section 5.3 describes the two sets of experiments we designed to assess the proposed method and explore its potential benefits, followed by the results and discussion section (Section 5.4) and a summary of the chapter (Section 5.5).

## 5.2. Methodology

For the sake of simplicity, we are going to describe the proposed semi-supervised training process considering a single image pair, (Figure 5.1). The two RGB images are concatenated for each sample image pair into a new six-channel image, $x_i$. A set of augmentation transformations

[12] According to Bachman et al. (2014), Dropout can also be considered a form of pseudo-ensembling, as it generates a family of child subnetworks from the original parent model by randomly masking a portion of its nodes.

is then applied to the image pair that is used as input to the student model $f_\theta$ producing a change prediction map $p_i$ as output. This prediction map is then compared to the ground truth change mask, $y_i$, in order to compute the supervised component of the loss function, $L_1$. For the supervised loss component, we are using the Binary Cross Entropy (BCE) loss (Equation 5.1).

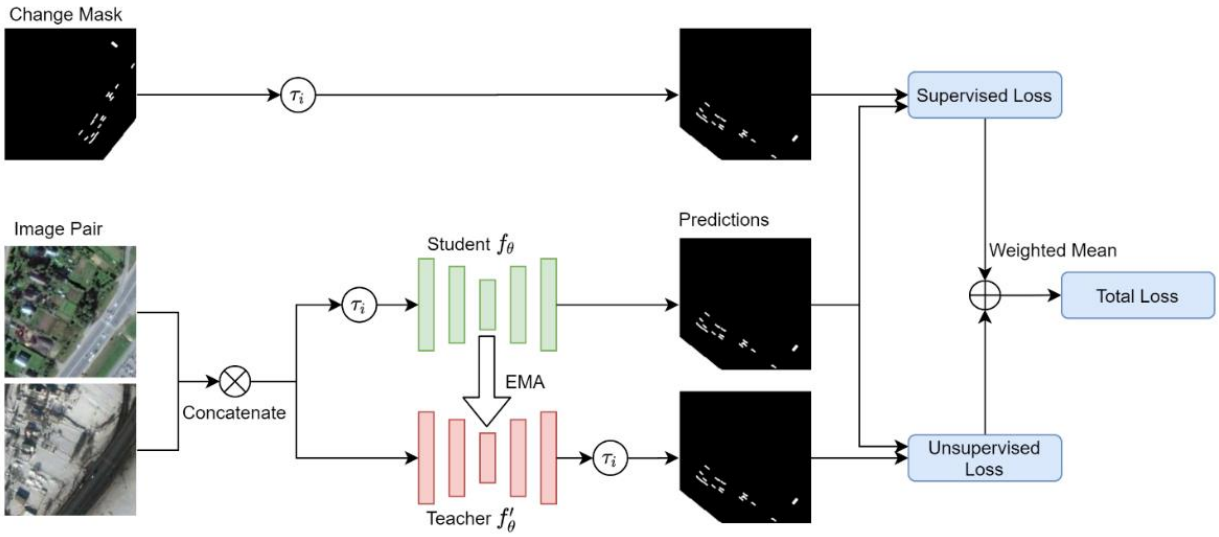



*Figure 5.1 : Overview of the proposed training approach.*

The original image pair is then fed to the teacher network, $f'_\theta$, to produce a second prediction, $f'_\theta(x_i)$, that is then subjected to the same set of transformations, $\tau_i$, so that the transformed prediction, $p'_i$, can be comparable to the one produced by the student model. Using predictions $p_i$ and $p'_i$, we can now compute the consistency regularization term of the loss function, $R_1$, that measures how consistent the model's predictions are when the images are subjected to random augmentations. For this unsupervised component of the loss function, we have used a Mean Squared Error (MSE) loss function (Equation 5.2).

The total loss function is a weighted average of the supervised and unsupervised components (Equation 5.3), where $\lambda(t)$ is an exponential weighting function that increases the weight of the consistency regularization term as the network's training progresses and its predictions become more accurate (Equation 5.4). The terms $t_{max}$ and $\kappa$ are hyperparameters of the function: $t_{max}$ is used to set the number of epochs (threshold) after which the ramp-up function takes its maximum value, and $\kappa$ is a scaling factor indicating the general weighting factor of the regularization term on the aggregated loss function.

The student's weights are updated through backpropagation while the teacher's weights are updated by an EMA (Equation 5.5).

$$L_1 = -(y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) \tag{5.1}$$

$$R_1 = \|p_i - p_i'\|_2 \tag{5.2}$$

$$Loss = L_1 + \lambda(t)(R_1 + R_2) \tag{5.3}$$

$$\lambda(t) = \begin{cases} \kappa \exp\left(-5\left(1 - \frac{t}{t_{max}}\right)^2\right), if\ t < t_{max} \\ \kappa, \qquad\qquad\qquad\qquad if\ t > t_{max} \end{cases} \tag{5.4}$$

$$f'_{\theta,t} = a\, f'_{\theta,t-1} + (a - 1) f'_{\theta,t} \tag{5.5}$$

The training process was described for a single image pair, while we used a minibatch size of 8 image pairs for the actual training. In the process so far, we have only considered the labelled examples. We can compute solely the consistency regularization term for the unlabelled part of the dataset. Still, since in practical applications the number of unlabelled samples greatly exceeds the number of labelled ones, we expect that the additional unlabelled information will improve the network's performance on the semantic segmentation task and lead to more robust predictions. Thus, the complete formula of the loss function will include one additional term, $R_2$, of the same form as $R_1$, referring to the consistency regularization term for the unsupervised image pairs of each minibatch. The complete training process is summarized in Algorithm 5.1.

For our student and teacher models we used the implementation of the UNet architecture we described in Section 3.1.1.

**Data:** $X_l : labeled\ image\ pairs$, $Y : labels$,
$X_u : unlabeledimagepairs$
**Input:** $Pairs(x_{i,1}, x_{i,2}) \in X_l \rightarrow y_i \in Y, \quad (x'_{i,1}, x'_{i,2}) \in X_u$

concatenate $x_{i,1}, x_{i,2}$ into $x_i \forall (x_{i,1}, x_{i,2})$ ;
concatenate $x'_{i,1}, x'_{i,2}$ into $x_i \forall (x'_{i,1}, x'_{i,2})$ ;
initialize student model, $f_\theta$ ;
initialize teacher model, $f'_\theta = f_\theta$ ;
create random minibatches $M$
with $m$ labeled and $m'$ unlabeled samples ;
**for** *t in number of epochs* **do**
  **for** *each minibatch M* **do**
    create random augmentations $\tau_i$ ;
    $p_{i \in m} \leftarrow f_\theta(\tau_i(x_m))$ ;
    $p'_{i \in m} \leftarrow \tau_i(f'_\theta(x_m))$ ;
    $L_1 = \frac{1}{|m|} \sum_{i \in m} -(y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$ ;
    $R_1 = \frac{1}{|m|} \sum_{i \in m} \|p_i - p'_i\|_2$ ;
    $R_2 = \frac{1}{|m'|} \sum_{i \in m'} \|p_i - p'_i\|_2$ ;
    $Loss = L_1 + \lambda(t)(R_1 + R_2)$ ;
    update $f_\theta$ weights using Adam optimizer ;
    update $f'_\theta$ weights using EMA ;
  **end**
  create new random minibatches ;
**end**

*Algorithm 5.1* **:** *Proposed semi-supervised training approach.*

## 5.3. Experiments

The purpose of the experiments was to assess the proposed semi-supervised change detection approach and to compare it to a fully supervised training framework in different scenarios varying the number of available labelled training samples. We conducted the initial set of experiments on the CD dataset proposed by Lebedev et al. (2018), which was also used in Chapters 4 and 5 and was described in more detail in Section 3.2.1. We will refer to this dataset from now on as Change Detection Dataset (CDD). The methodology, as well as this first set of experiments, were published as a conference paper in the ISPRS2021 Congress (Bousias Alexakis & Armenakis, 2021a). However, as seen in Section 5.4.1 and the published paper, the initial application of the proposed Mean Teacher semi-supervised method on CDD did not improve the models' performance in any notable way compared to a typical fully supervised training framework.

During the development of the post-classification change detection approach, which is described in the following chapters (Chapters 6 and 7), we developed a method for domain adaptation of building semantic segmentation, which was based on a similar Mean Teacher training strategy – although it also introduced several further enhancements which will be discussed in the respective chapters – and significantly improved the model's performance on the target domain. Thus, we conducted a second set of experiments in an attempt to enhance the results of the semi-supervised approach and extract more solid conclusions by applying the newly acquired knowledge on training using the Mean Teacher framework. For this second set of experiments, we used the LEVIR-CD building change detection dataset (H. Chen & Shi, 2020), which is a more recent and significantly more extensive change detection dataset that should also lead to more reliable conclusions.

Therefore, each of this chapter's Experiments and Results and Discussion sections is organized into two subsections relating to the initial set of experiments on the CDD dataset and to the second – improved – set of experiments conducted on the LEVIR-CD dataset, respectively.

### 5.3.1 Experiment 1: Training Details (CDD)

This section describes the training details of the first set of experiments conducted on the CDD dataset.

We have randomly split the CDD training set into smaller sets $S_{500} \subset S_{1000} \subset S_{2500} \subset S_{5000} \subset S_{7500}$, each containing 500, 1000, 2500, 5000, and 7500 labelled image pairs. We have used the rest of the image pairs for each smaller set as unlabelled training samples. So, for example, $S_{500}$ will be trained using 500 labelled and 9500 unlabelled samples, $S_{1000}$ will be trained using 1000 labelled and 9000 unlabelled samples, and so forth. Finally, we have trained the network using all 10000 labelled training samples in a fully supervised way to use as a benchmark for the results retrieved using the smaller labelled training sets.

The training was conducted on an NVIDIA Quadro RTX 5000 GPU using PyTorch (Paszke et al., 2019). We used the Albumentations (Buslaev et al., 2020) library for applying the image data augmentations. For the transformations $\tau_i$, we have used random 90-degree angle rotations, horizontal and vertical flipping, and random crop and rescale transforms. Besides the geometric transformations, we introduced additional radiometric augmentations: an RGB shift

augmentation, where the RGB values of an image are shifted by a randomly chosen value for each channel in the interval (-20, +20), as well as a random brightness and contrast augmentation.

In order to reduce the training time, we first trained a UNet model from scratch on the 2500 sample set without any data augmentation for 150 epochs – as we noticed that at that point, the network started to overfit the training set. For the training sets containing 2500 samples or more, we used the pre-trained network's weights as starting weights and trained for another 46875 iterations[13]. We used a learning rate of 0.0003 for the first 2/3 of the training and of 0.0001 for the last 1/3. The training sets containing less than 2500 labelled examples ($S_{500}$ and $S_{1000}$) were trained from scratch following the same principles.

### 5.3.2 Experiment 2: Training Details (LEVIR-CD)

This section describes the training details of the second set of experiments conducted on the LEVIR-CD dataset.

Using the LEVIR-CD training set, we created four different training datasets corresponding to four scenarios with varying portions of the training samples' labels available for training: 5%, 10%, 20%, and 100%. The labelled examples for each new dataset were selected randomly, and from the remaining samples, we only used the two RBG instances of each image pair for the unsupervised component of our semi-supervised training framework.

All models were trained for 25,000 iterations. To speed up the training process, we used the weights of the model trained on the smallest training set (i.e., using 5% of the labelled samples) for 10,000 iterations in a supervised manner as the initial weights for all other models. The models were then trained for another 15,000 iterations on their corresponding dataset using the Adam optimizer and a learning rate that starts at 0.0003 and decreases exponentially following the equation $lr_{t+1} = 0.9 \cdot lr_t$ every 1,000 iterations ($t$ represents thousands of iterations).

[13] The number of iterations is not rounded because, in our original implementation, we used the number of epochs and not the number of iterations to iterate over the datasets. Thus 150 epochs on the 2500-sample set with a minibatch size of 8 translate to 46875 iterations. Similarly, 46875 iterations correspond to 75 epochs on the 5000-sample set with the same minibatch size and so on for the rest of the training sets.

The training was conducted on an NVIDIA Quadro RTX 5000 GPU using PyTorch (Paszke et al., 2019). We used the same data augmentation process as the one described in the first experiment.

In addition, we replaced the batch normalization layers of the UNet architecture with group normalization layers following the discussion and findings of Section 7.3.2.2 regarding the effects of Batch Normalization on domain adaptation and the suggestions of (Y. Sun et al., 2020) on the benefits of Group Normalization compared to Batch Normalization for Test Time training – an approach similar to domain adaptation – using small minibatch sizes. Finally, we changed the regularization loss functions from Mean Square Error to Binary Cross Entropy Loss in accordance with the findings of Appendix C.

#### 5.3.2.1 Building Change Detection Dataset – LEVIR-CD

The dataset consists of 637 co-registered bitemporal RGB images and their corresponding building-related change masks. The images, which have a size of 1024 × 1024 pixels and a spatial resolution of 0.5m, were collected from Google Earth and cover 20 different regions of various cities in Texas, USA. The capture time ranges from 2002 to 2018 and may vary for each region. The existence of significant land-use changes, as well as the variations in illumination and seasonal changes between the bi-temporal images, were some of the main factors that were considered for the time-capture and region selection (H. Chen & Shi, 2020). Figure 5.2 shows four randomly selected image pair samples and the corresponding ground truth building change masks.

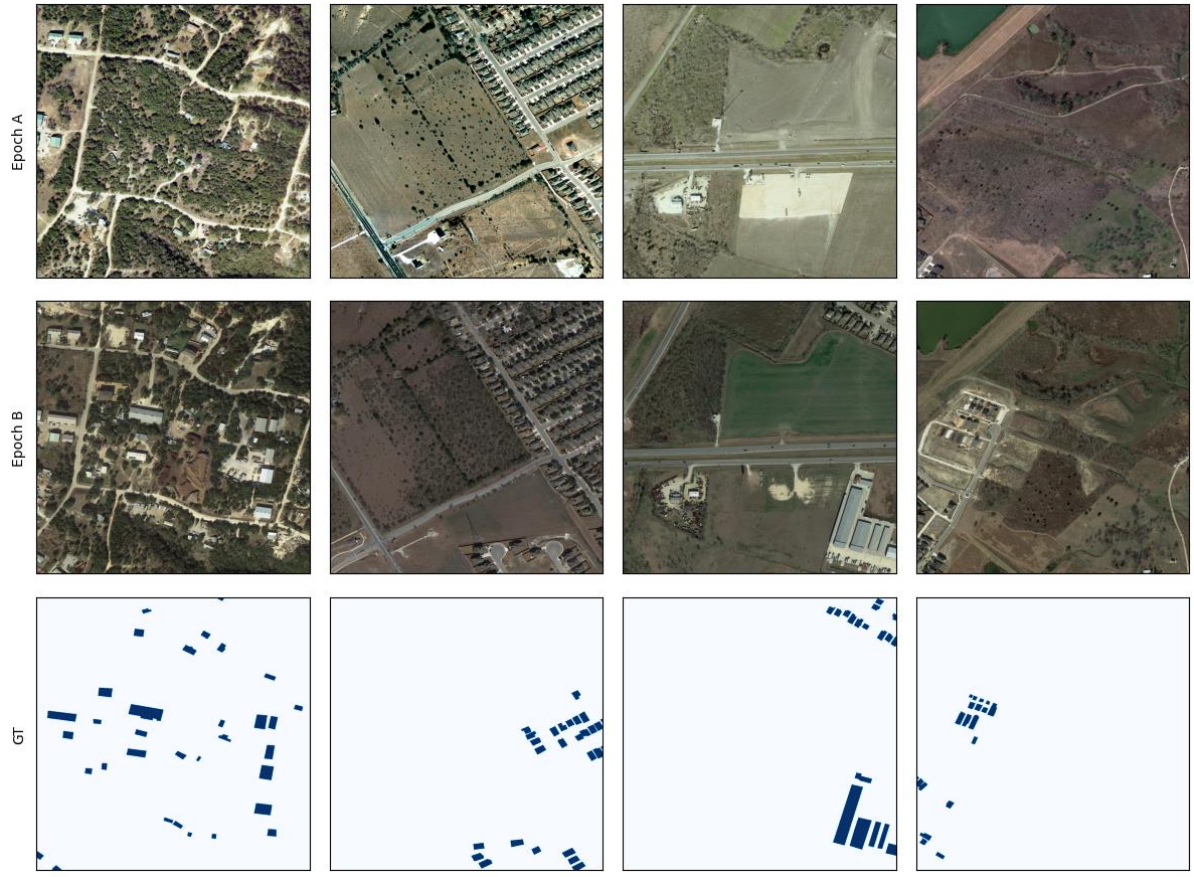



*Figure 5.2: Sample image pairs with their corresponding building change masks for LEVIR-CD.*

## 5.4. Results and Discussion

### 5.4.1 Results on CDD dataset – Experiment 1

This Section presents the results on the first set of experiments conducted on the CDD dataset. Figure 5.3 and Table 5.1 present the Intersection over Union (IoU) results we retrieved on the training and validation dataset for different labelled sample sizes. The only case where the semi-supervised training achieves better performance on the validation set is for the 2500 labelled sample size. In all other cases, the supervised training with augmentations performs better than the proposed semi-supervised approach.

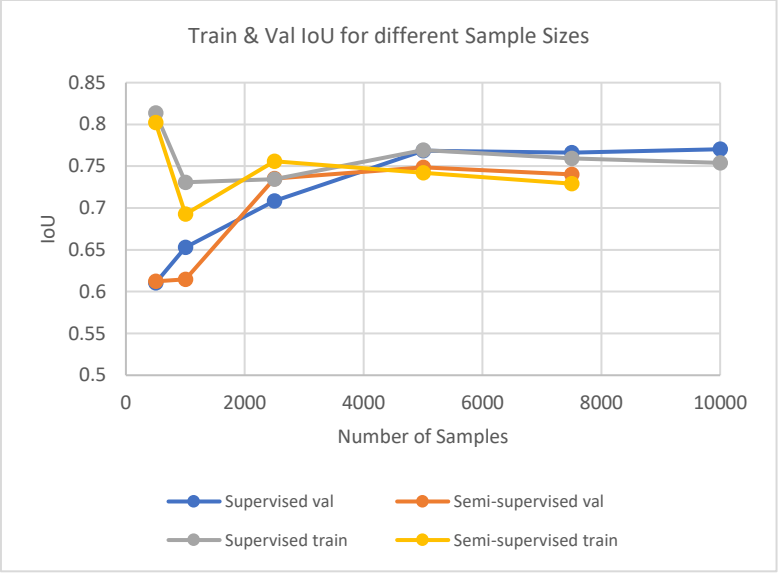



*Figure 5.3: Training and validation IoU metrics for semi-supervised and supervised models for varying number of labelled training samples.*

The results are contrary to our initial expectations. We expected that when the number of samples is small, the semi-supervised approach would perform better than the supervised one thanks to the extra information provided by the unlabelled data and that as the number of training samples increased, the benefits of the semi-supervised training scheme would wear out with the two methods producing similar results for higher sample sizes. Instead, there is no distinct pattern connecting the relative performance of the two methods with the number of samples. On the smallest labelled sample size, both approaches perform similarly (around 61%) and greatly overfit the training data. When the labelled sample size is set to 1000, the semi-supervised model has a validation IoU about 4% lower than the fully supervised one. For more extensive training sizes, the IoU of the fully supervised approach exceeds the semi-supervised IoU (by about 2% on $S_{5000}$and 2.5% on $S_{7500}$). The only case where the semi-supervised approach outperforms the fully supervised results is on the $S_{2500}$ (by 2.6%).

In our initial experiments, we did not use any dropout layers (Srivastava et al., 2014) in order to examine whether the geometric and radiometric augmentations would be sufficient perturbations for the semi-supervised training to succeed. Since (X. Li et al., 2021) used dropout in their solution that outperformed the fully supervised training, we also ran extra experiments applying an additional dropout regularization with a probability of 0.3 (or 30%) on the output of the last convolutional block and before applying the final convolutional layer of the model. Dropout was

applied on three of the training schemes ($S_{500}$,$S_{1000}$, $S_{5000}$) and resulted in an improved IoU for $S_{1000}$ (about 2.5%), but still lower than the respective fully supervised result and in slightly worse IoUs (less than 0.5%) in the case of $S_{500}$ and $S_{5000}$.

When considering each method individually, the results seem reasonable. For very small sample sizes, there is a large gap between training and validation IoU, suggesting overfitting to the small training set for both models (the gap is around 20% for both models), which is gradually closing as the number of samples increases, with the validation IoU being even higher than the training IoU for bigger labelled sample sizes (this is the case for $S_{7500}$ for both supervised and semi-supervised methods and $S_{10000}$ for the supervised one) indicating that more labelled training samples help the models generalize better, which is the expected behaviour. The fact that the validation IoU is higher than the training IoU may relate to a condition included in the training when saving the best model. The condition was based solely on the IoU performance on the validation set as a safety net against overfitting.

| Number of labelled samples | Dropout | Training method | Initial Weights | Train loss | Train IoU | Val loss | Val IoU |
|---|---|---|---|---|---|---|---|
| 500 | ✗ | Sup | Random | 0.0483 | 0.8136 | 0.1617 | 0.6105 |
| 500 | ✗ | Semi-Sup | Random | 0.0649 | 0.8021 | 0.1554 | 0.6123 |
| 500 | ✓ | Semi-Sup | Random | 0.0552 | 0.8316 | 0.1664 | 0.6101 |
| 1000 | ✗ | Sup | Random | 0.0759 | 0.7308 | 0.1081 | 0.6529 |
| 1000 | ✗ | Semi-Sup | Random | 0.0952 | 0.6926 | 0.1178 | 0.6146 |
| 1000 | ✓ | Semi-Sup | Random | 0.0851 | 0.7415 | 0.1185 | 0.6403 |
| 2500 | ✗ | Sup | Pretrained | 0.0751 | 0.7347 | 0.0823 | 0.7087 |
| 2500 | ✗ | Semi-Sup | Pretrained | 0.0780 | 0.7561 | 0.0727 | 0.7351 |
| 5000 | ✗ | Sup | Pretrained | 0.0649 | 0.7695 | 0.0621 | 0.7686 |
| 5000 | ✗ | Semi-Sup | Pretrained | 0.0836 | 0.7421 | 0.0686 | 0.7487 |
| 5000 | ✓ | Semi-Sup | Pretrained | 0.0796 | 0.7426 | 0.0709 | 0.7446 |
| 7500 | ✗ | Sup | Pretrained | 0.0673 | 0.7592 | 0.0613 | 0.7663 |
| 7500 | ✗ | Semi-Sup | Pretrained | 0.0892 | 0.7290 | 0.0710 | 0.7402 |
| 10000 | ✗ | Sup | Pretrained | 0.0694 | 0.7539 | 0.0607 | 0.7703 |

*Table 5.1 Training and validation IoU metrics for semi-supervised and supervised models for varying number of labelled training samples*

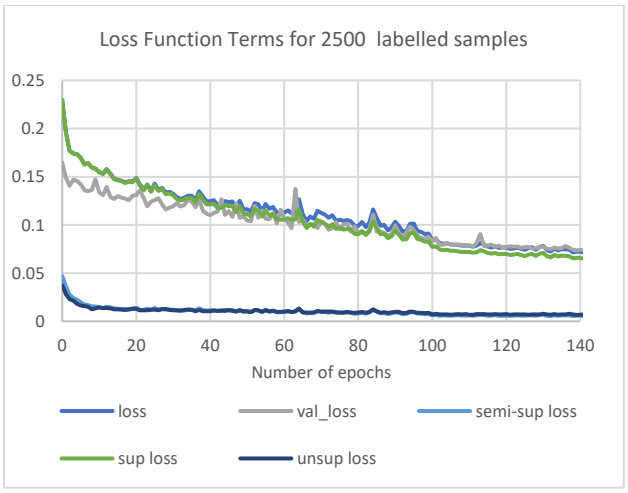



*Figure 5.4: Loss function terms of the semi-supervised method for $S_{2500}$. Loss is the aggregate loss, sup loss is the supervised component of the loss function ($L_1$), semi-sup loss is the consistency regularization term for the labelled samples ($R_1$), and unsup loss is the consistency regularization term for the unlabelled samples ($R_2$) of the dataset.*

The training and validation loss curves (average loss function values per epoch) presented in Figure 5.4 and Figure 5.5 can help us examine the learning behaviour of the semi-supervised approach. We can see that the consistency regularization terms have a significantly lower range of values. All terms decrease over time and seem to converge by the end of training. Also, the change of the learning rate from 0.0003 to 0.0001 at epochs 100 (for $S_{2500}$) and 75 (for $S_{5000}$) is visible for both learning curves.

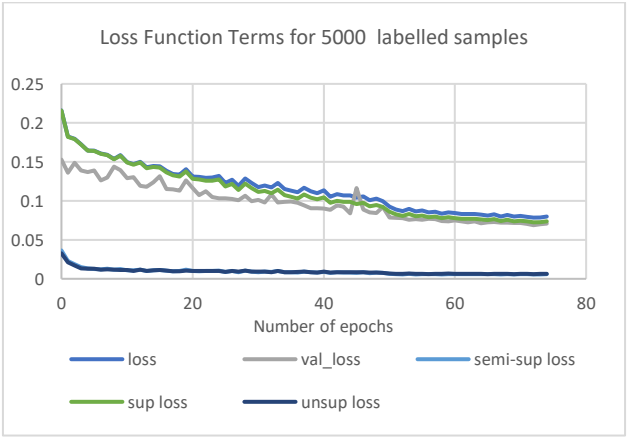



*Figure 5.5: Loss function terms of the semi-supervised method for $S_{5000}$. Loss is the aggregate loss, sup loss is the supervised component of the loss function ($L_1$), semi-supervised loss is the consistency regularization term for the labelled samples ($R_1$), and unsup loss is the consistency regularization term for the unlabelled samples ($R_2$) of the dataset.*

In Figure 5.6, we present predictions from different models for eight image pairs selected randomly from the validation set. Overall, a qualitative analysis of the predictions suggests that the models trained on 2500 images (both supervised and unsupervised) seem to perform similarly to the fully supervised model trained on 10000 images, while the models trained on the 500 images do not seem to produce accurate predictions, especially when it comes to small or thin elongated objects and regions with complex shapes/boundaries.

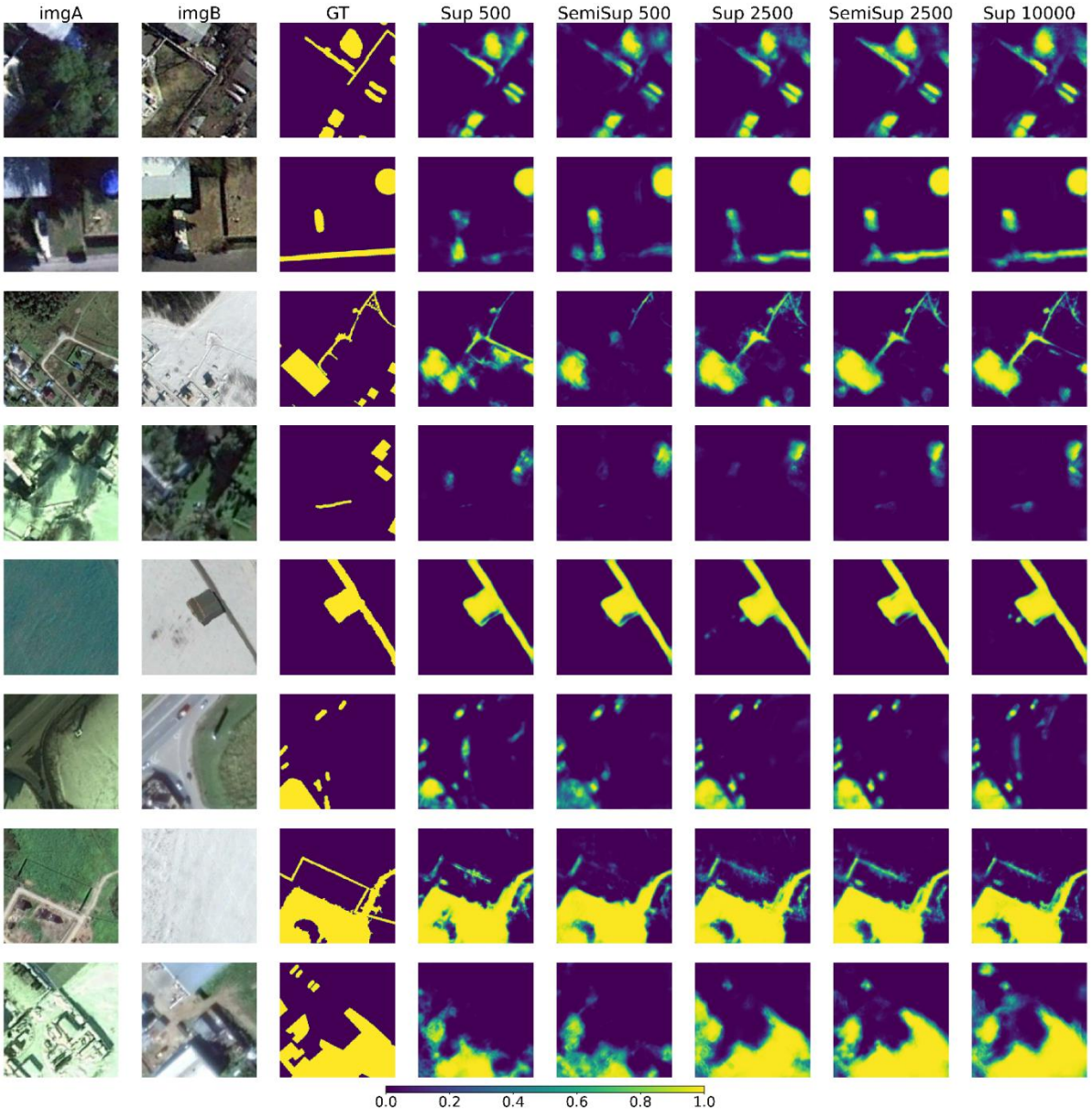
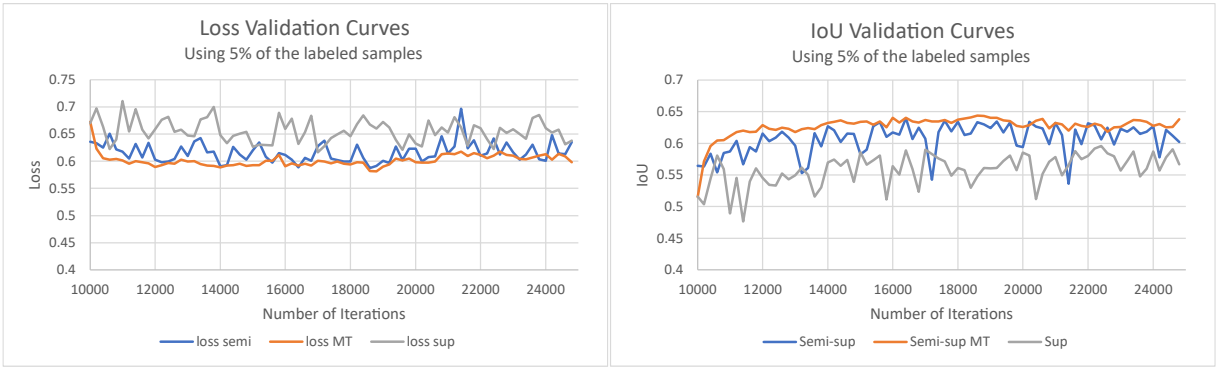



*Figure 5.6: Prediction examples for different sample sizes for both supervised and semi-supervised training.*

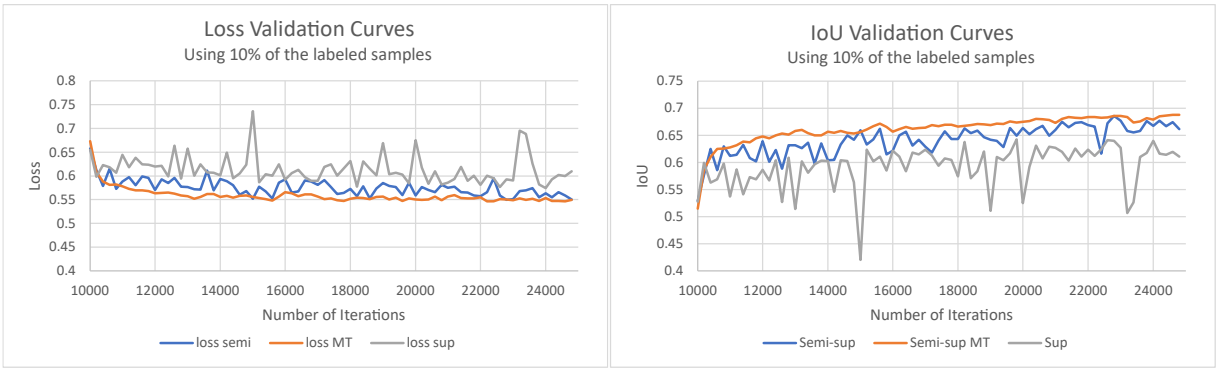
### 5.4.2 Results on LEVIR-CD dataset – Experiment 2

This Section presents the results, analysis and discussion of the second set of experiments conducted on the LEVIR-CD dataset. Figures 5.7 to 5.10 illustrate the validation curves of (a) the loss function and (b) the IoU metric for increasing percentages of labelled samples used for training the models either in a supervised or semi-supervised manner. The figures include the performance of both the Mean Teacher (MT) and the student models in the case of semi-supervised training. In this case, the semi-supervised approach consistently outperforms the traditional supervised method, especially in the cases where we used 5%, 10%, and 20% of the labelled data, with the semi-supervised MT models exhibiting the best and most stable overall performance in every experiment.



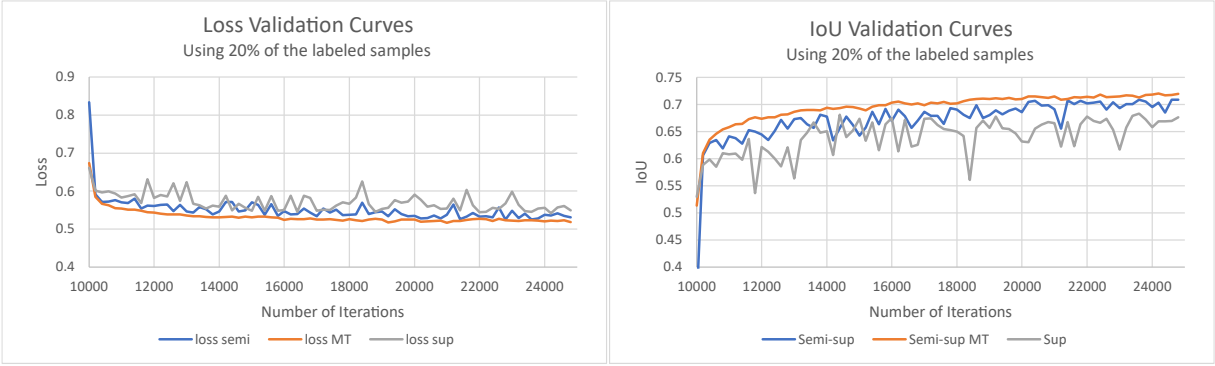

(a) (b)

*Figure 5.7:Validation curves of the loss function (a) and the IoU metric (b) for the supervised and semi-supervised training frameworks when using only 5% of the training labels while training.*



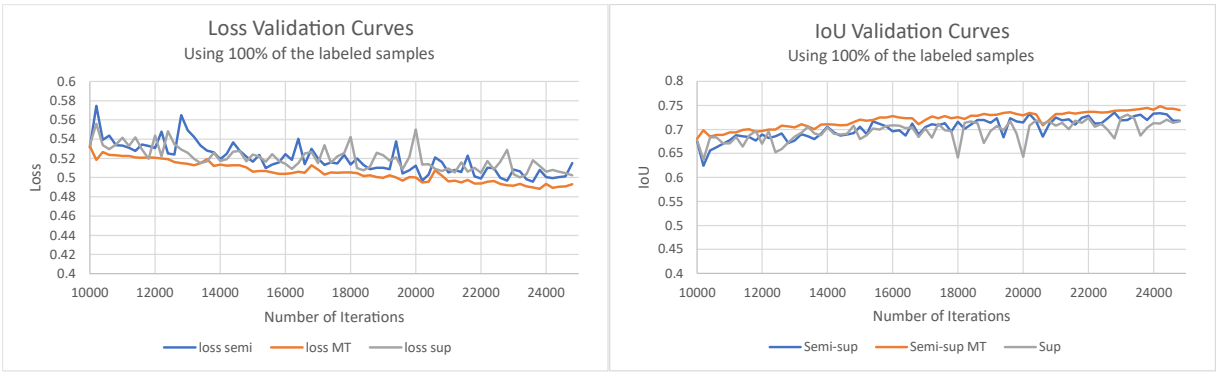

(a) (b)

*Figure 5.8:Validation curves of the loss function (a) and the IoU metric (b) for the supervised and semi-supervised training frameworks when using only 10% of the training labels while training.*




(a) (b)

*Figure 5.9:Validation curves of the loss function (a) and the IoU metric (b) for the supervised and semi-supervised training frameworks when using only 20% of the training labels while training.*




(a) (b)

*Figure 5.10:Validation curves of the loss function (a) and the IoU metric (b) for the supervised and semi-supervised training frameworks when using 100% of the training labels while training.*

Figure 5.11 compares the validation IoU curves retrieved using the supervised training method and 5, 10, 20, and 100% of the available labelled training data, respectively. As expected, the results indicate that as the number of labelled training samples increases, so does the model's performance on the IoU metric. Similarly, Figure 5.12 presents the validation IoU curves of the Mean Teacher model for the semi-supervised training scheme when utilizing 5, 10, 20, and 100% of the available training labelled data. It also compares the curves to the supervised solution that was trained on the entire training dataset. Once again, increasing the percentage of labelled samples used for training improves the model's performance as measured by the IoU metric. It should be noted that the MT model trained according to the semi-supervised framework using only 20% of the training samples matches the performance of the model trained according to the ordinary supervised training framework using the entire labelled training set, thus practically reducing the need for labelled training data by 80%.

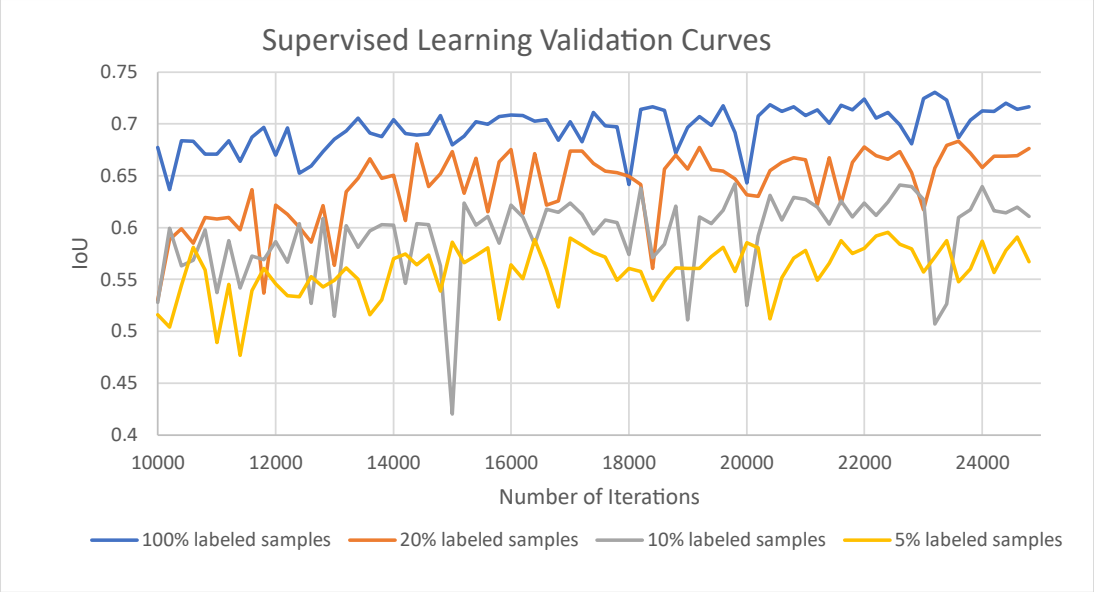



*Figure 5.11: Validation IoU curves retrieved when implementing supervised training and 5, 10, 20 and 100% of the available labelled training data.*

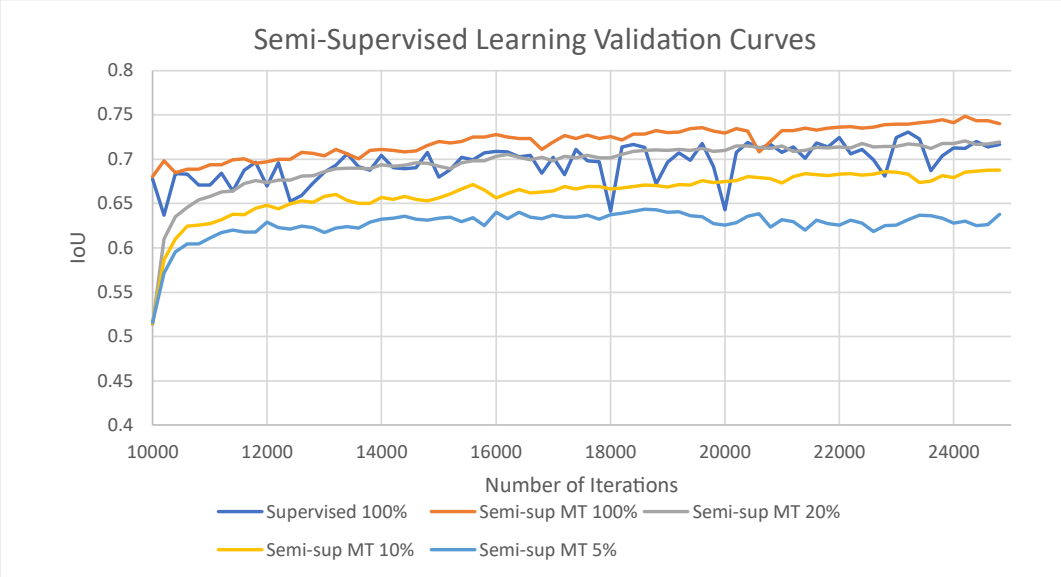



*Figure 5.12: Validation IoU curves retrieved when applying semi-supervised training and 5, 10, 20, and 100% of the available labelled training data. The supervised validation curve when using 100% of the available labelled samples for training is also included for comparison purposes.*

Table 5.2 summarizes the evaluation metrics – accuracy, precision, recall, F1 score, and IoU – achieved by each training method when using 5, 10, 20, or 100% of the training set's labelled samples. The table presents multiple models corresponding to the same training scenario (dataset) and training method, which are selected based on a different end criterion. A simple way to choose the final student model is to pick the model produced after the last training

iteration (named "# of Iterations" in the table below). Alternatively, we could select the model with the highest IoU value in the validation dataset. Although this model will most likely perform better in cases of extensive validation sets, if very few labelled training samples are available – which is the scenario we are interested in – it will not be possible to have a big enough validation dataset to ensure that the model selected based on the validation IoU does not simply overfit the data[14].

In all four training datasets, the Mean Teacher model trained according to the semi-supervised approach outperformed the corresponding supervised models in all metrics. In addition, to better visualise the benefits of the semi-supervised Mean Teacher model compared to the model produced through the original supervised training framework, we plotted the evaluation metrics' differences in Figure 5.13. According to the graph, the greatest boost was reported for the recall rate in all four scenarios. For the 5,10, and 20% cases, the improvements in the IoU are higher than 4%, in the F1 score higher than 3%, and in the Recall rate as high as 6%. At the same time, there are also benefits, albeit smaller ones (between 1 and 2% on recall, F1 score, and IoU), even when using all the labelled samples of the training set.

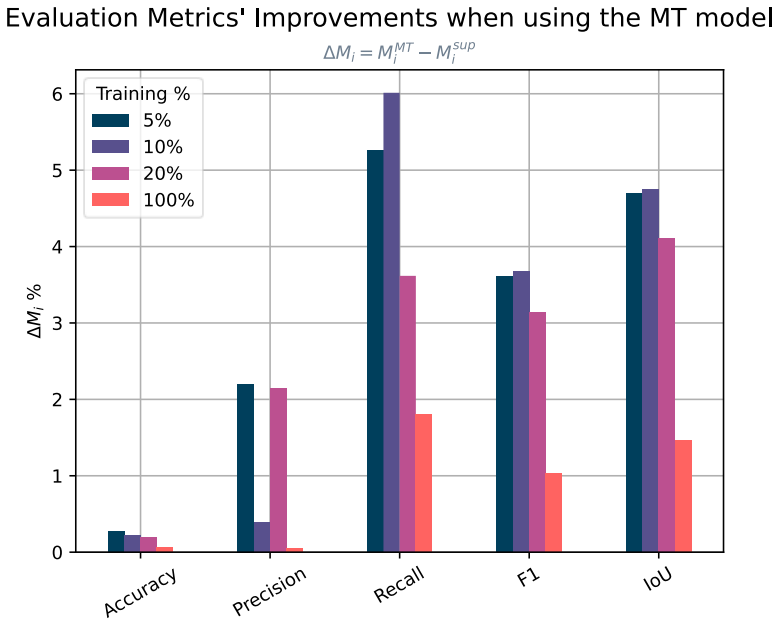



*Figure 5.13: Difference between the evaluation metrics retrieved using the semi-supervised Mean Teacher model and the original supervised solution, where $M_i$ corresponds to any of the metrics.*

[14] Section 7.3.1.1 further discuss the best model selection issue.

*Table 5.2: Evaluation metrics on the validation dataset using 5, 10, 20 and 100% of the training set's labelled samples for the supervised and semi-supervised training methods.*

| Training Percentage | Training Method | Best Model Criterion | Accuracy | Precision | Recall | F1 | IoU |
|---|---|---|---|---|---|---|---|
| 5.00% | Supervised | # Iterations | 0.9770 | 0.7759 | 0.7257 | 0.7379 | 0.5978 |
| 5.00% | Supervised | Best IoU | 0.9781 | 0.7935 | 0.7249 | 0.7507 | 0.6101 |
| 5.00% | Semi-sup | # Iterations | 0.9787 | **0.8284** | 0.6975 | 0.7430 | 0.6048 |
| 5.00% | Semi-sup | Best IoU | 0.9804 | 0.8077 | **0.7811** | 0.7863 | 0.6554 |
| 5.00% | Semi-sup | MT | **0.9808** | 0.8155 | 0.7775 | **0.7868** | **0.6570** |
| 10.00% | Supervised | # Iterations | 0.9760 | 0.8496 | 0.6067 | 0.6870 | 0.5447 |
| 10.00% | Supervised | Best IoU | 0.9817 | 0.8489 | 0.7457 | 0.7858 | 0.6581 |
| 10.00% | Semi-sup | # Iterations | 0.9817 | 0.8080 | 0.8178 | 0.8062 | 0.6821 |
| 10.00% | Semi-sup | Best IoU | 0.9836 | 0.8422 | **0.8102** | 0.8198 | 0.7019 |
| 10.00% | Semi-sup | MT | **0.9839** | **0.8528** | 0.8058 | **0.8225** | **0.7056** |
| 20.00% | Supervised | # Iterations | 0.9818 | 0.8784 | 0.7060 | 0.7722 | 0.6456 |
| 20.00% | Supervised | Best IoU | 0.9843 | 0.8522 | 0.7943 | 0.8166 | 0.7001 |
| 20.00% | Semi-sup | # Iterations | 0.9838 | 0.8405 | 0.8143 | 0.8238 | 0.7057 |
| 20.00% | Semi-sup | Best IoU | 0.9854 | 0.8567 | **0.8305** | 0.8407 | 0.7300 |
| 20.00% | Semi-sup | MT | **0.9862** | **0.8737** | 0.8304 | **0.8480** | **0.7412** |
| 100.00% | Supervised | # Iterations | 0.9806 | 0.7638 | 0.8337 | 0.7930 | 0.6667 |
| 100.00% | Supervised | Best IoU | 0.9868 | 0.8691 | 0.8461 | 0.8547 | 0.7505 |
| 100.00% | Semi-sup | # Iterations | 0.9853 | 0.8363 | 0.8580 | 0.8447 | 0.7346 |
| 100.00% | Semi-sup | Best IoU | 0.9867 | **0.8748** | 0.8418 | 0.8560 | 0.7516 |
| 100.00% | Semi-sup | MT | **0.9874** | 0.8696 | **0.8641** | **0.8650** | **0.7652** |

## 5.5. Summary

In this work we developed a Mean Teacher semi-supervised training setup and applied it to a Change Detection semantic segmentation setting to explore the potential benefits of the method when only a few labelled training examples are available and compare to a fully supervised training process. We expected that the consistency regularization constraint would allow the model to learn useful information from unlabelled data, improving the model's performance when limited labelled samples are available, which is a common issue in CD applications.

We conducted two sets of experiments designed to assess the potential benefits of the proposed training algorithm. The preliminary results of the first experiment, on the CDD dataset (Lebedev et al., 2018), suggested that the proposed approach did not outperform the typical fully supervised training process. Contrary to our initial expectations, there is no clear relation between the size of the labelled training set and any performance benefits of applying the semi-supervised training scheme instead of a fully supervised solution. Thus, the results of the first experiment suggest that the fully supervised approach slightly outperforms the semi-supervised method for almost all labelled training set sizes, with the exception of $S_{2500}$.

On the contrary, the results of the second set of experiments, conducted on the much larger and thus more reliable LEVIR-CD building change detection dataset (H. Chen & Shi, 2020), clearly indicated improved semantic segmentation performance when training the models according to the proposed semi-supervised Mean Teacher framework. The models achieved higher evaluation metrics on every training scenario – where only a small portion (5, 10, and 20%) of the training samples' labels was provided during the model's training – and for every metric, with the strongest benefits reported on the recall rate (6%). Notably, by comparing the evaluation curves of the MT semi-supervised models and the fully supervised models we found that a MT model trained using only 20% of the labelled samples can approximately match the performance of a fully supervised model trained on the entire training dataset.

# 6

# Decoupling Body and Edge Information for Enhanced Building Semantic Segmentation

## 6.1. Introduction

In the present and the following chapters, we present a post-classification change detection approach that aims to eliminate the need for labelled change detection training samples. Our method classifies each epoch of the image pair separately and can take advantage of existing extensive semantic segmentation remote sensing datasets thanks to a proposed unsupervised domain adaptation approach. As a proof of concept, we focused our implementation on determining the land cover spatial changes of building rooftops, but the approach could be generalized to any other class. The proposed post-classification change detection pipeline consists of three main components:

- A ResUNet-based encoder-decoder CNN for building semantic segmentation, which leverages the buildings' boundary information to enhance the building segmentation performance (and is presented in the current chapter),
- A domain adaptation training framework based on the Mean Teacher (MT) (Tarvainen & Valpola, 2017) training paradigm, which aims to bridge the performance gap between the source and the target domains (Chapter 7) and

- A filtering process to remove prediction errors due to small misclassification and misregistration errors in the prediction building masks (Chapter 7).

The remainder of the chapter is dedicated to the first component of the process, which is the development and evaluation of an encoder-decoder architecture for the semantic segmentation of building rooftops from high-resolution imagery. The complete post-classification pipeline and focus on the domain adaptation approach and the pruning of the produced change maps will be presented in detail in Chapter 7.

For the building semantic segmentation component of the pipeline, we introduce a new end-to-end approach that aims to leverage the semantic information provided by the objects' boundaries to improve the quality and detail of the model's semantic segmentation output.

As mentioned in Section 2.2, various recent research works focus on leveraging the semantic information of object boundaries in order to improve the localization accuracy of the model's semantic segmentation predictions (Girard et al., 2021; X. Li et al., 2020; Marmanis et al., 2018; K. Zhao et al., 2018). X. Li et al. (2020) propose decoupling the body from the boundary of each segmented object within the model in order to learn separate representations for the body and edge parts, thus improving the final segmentation performance. In Chapter 4, under direct change detection determination, we introduced an edge-enhanced architecture that combines a U-shaped fully convolutional architecture and an edge detection CNN architecture to improve the model's prediction accuracy on the direct change detection approach.

In this chapter, we follow a different approach to leverage the objects' edge information, drawing inspiration from the work of X. Li et al. (2020). We develop an encoder-decoder architecture that learns explicit representations of the main body and the boundaries of buildings by incorporating a decoupling module into a UNet-based CNN architecture. We explore the potential benefits of explicitly using the objects' boundary information in a CNN architecture to help the model learn better feature representations and improve the accuracy of the predicted outputs. When enhancing the direct change detection approach with semantically informed edges (method proposed in Chapter 4), we trained a separate edge detection network using the boundary information from the Vaihingen land cover semantic segmentation dataset. Thus, the edges reflect the semantic information of the six categories/classes defined in the dataset. In this case,

the boundary information is retrieved from the same dataset used to train the model on the building semantic segmentation task (i.e., the Inria aerial image labelling dataset). Hence, the used edges correspond to semantic boundary information in both cases.

We also compare the proposed model to an equivalent 'plain' UNet-based architecture and to a UNet-based network, which, besides the building segmented area, also learns to predict the building edges as a separate object class. Furthermore, we investigate whether the more straightforward approach – compared to the decoupled body and edge segmentation – of introducing a separate edge class helps the model learn better representations and improve the predicted footprints compared to the plain UNet-based architecture.

In Section 6.2, we present the proposed network architecture, discuss the training details, and briefly introduce the Inria Aerial Image Labelling Dataset (Maggiori et al., 2017) that was used for the training and evaluation of all our models. In Section 6.3, we present the results analysis and discussion, and finally, in Section 6.4, we outline our conclusions regarding the use and utility of the proposed decoupled body and edge architecture.

## 6.2. Methodology

The proposed architecture was designed to combine the benefits of a UNet-based encoder-decoder architecture and the body and edge decoupling module introduced by X. Li et al. (2020).

Initially, we describe our backbone UNet-based architecture, which will also be used as our baseline in our experiments. Then, we introduce the body and edge decoupling module and explain how and why it is integrated into the ResUNet backbone model to create the proposed architecture. In the end, we present the loss functions and the dataset we used to train and evaluate the models, as well as some details pertaining to the supervised training process.

### 6.2.1 Baseline Model – Backbone architecture

Once again, we use a UNet model variation as our baseline architecture for building semantic segmentation (Figure 6.1). Our model has certain improvements compared to the traditional UNet architecture we used in previous chapters. First, we have replaced the original encoder with the much deeper and more expressive ResNet50 model (K. He et al., 2016). Residual Networks (ResNets) introduce shortcut connections between layers to address the degradation of training

accuracy that is observed in very deep networks. Second, we have introduced batch normalization layers (Ioffe & Szegedy, 2015) in all up-sampling blocks as they have been shown to facilitate deep models' training and help address the vanishing/ exploding gradients problem. We have also experimented with introducing the building edges ($L_e$) as an extra, optional network output besides the building footprints to investigate whether adding an extra supervised task would lead to more descriptive features and slight improvements in the accuracy of the building footprint predictions.

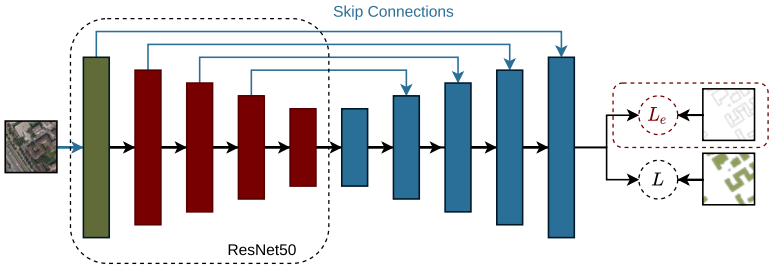



*Figure 6.1. Baseline model to predict the building footprints (L) and optionally the building boundaries ($L_e$) as an auxiliary task.*

### 6.2.2 Body and Edge Segmentation

The development of separate feature representations for the body and boundaries of objects is achieved via the decoupled edge and body attention module. The module is inspired by the work of (X. Li et al., 2020) and comprises the body and edge decoupling submodule (Figure 6.2 – bottom right) and the Decoupled Attention and Feature Fusion sub-module (Figure 6.2 – top right). An overview of the proposed architecture is displayed in the bottom left corner of Figure 6.2.

The body and edge decoupling module is designed to decompose the feature maps' objects into their body (low frequency) and edge (high frequency) components. The module's input is a set of intermediate (level 2) features which are transformed into a lower frequency / simplified version of themselves by first passing them through a down-sampling block that reduces the features' dimension by a factor of four and consecutively re-up-sampling the features back to their original resolution by linear interpolation. Even though it is not guaranteed that the lower resolution representations will not also contain high-frequency information, X. Li et al. (2020) suggest that the down-sampled feature maps summarize most of the salient parts of objects and can be used

as an approximation of smoothed representations of the initial feature maps, where the high-frequency parts are ignored. Next, a convolutional layer, which takes both the original input features and the simplified ones as input, is used to create a flow field that points toward each object's centre. The estimated field is used to warp the original feature maps into the body feature maps using the warping process described by (Jaderberg et al., 2015). The edge features can then be estimated by subtracting the body from the original input features.

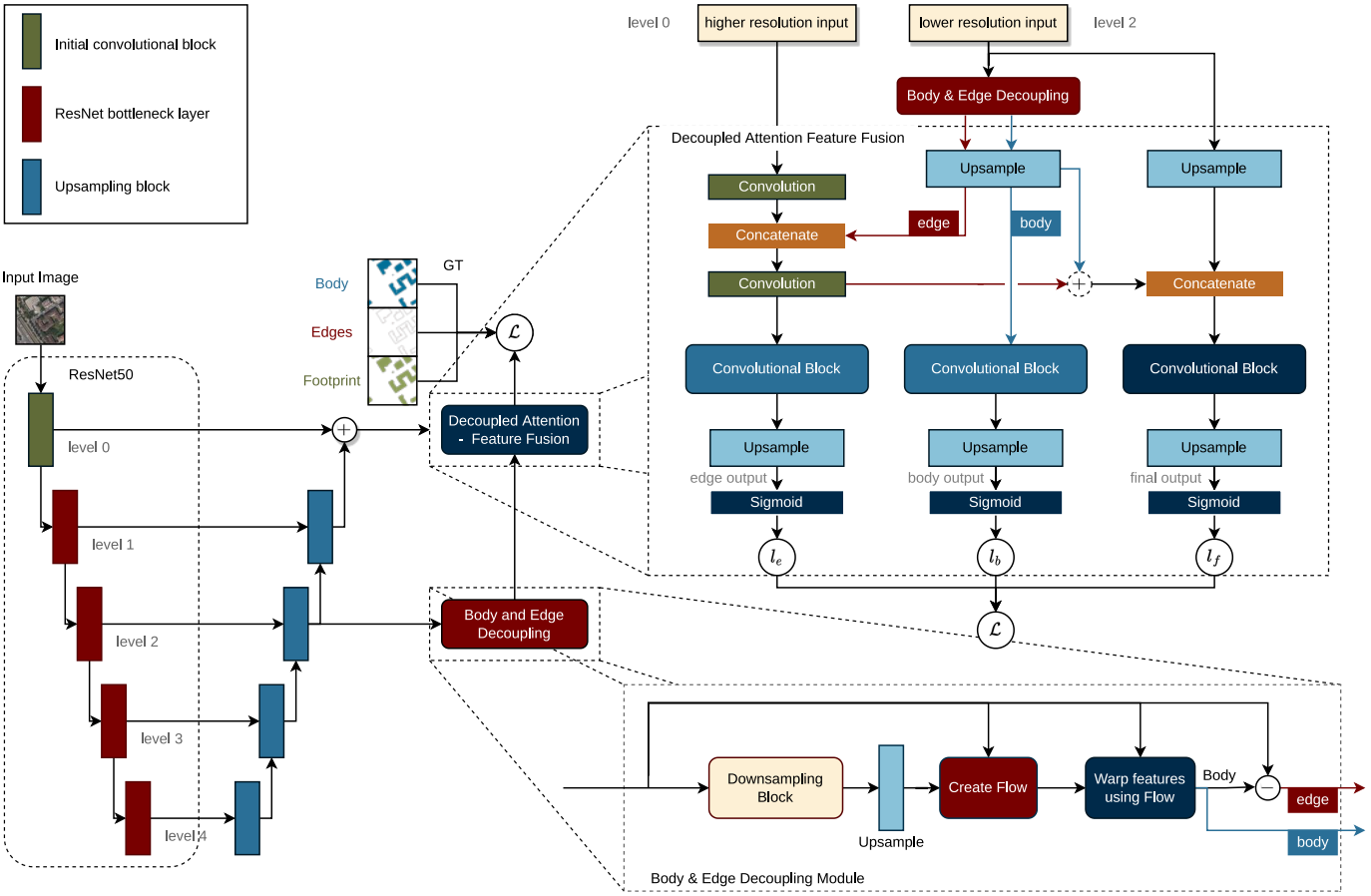



*Figure 6.2: The proposed Decoupled Edge and Body (DEB) ResUNet architecture for building semantic segmentation.*

The Decoupled Attention and Feature Fusion sub-module (Figure 6.2 top right) consists of three main upsampling flows (streams): one for the edge representations, one for the low frequency/body representations and one that combines the two and produces the final/fused building predictions. The body, edge, and fused output streams can all be trained in a supervised manner. Given the original 'complete' building masks of the dataset one can easily extract the edge and body masks that can then be used to train the network (see Section 6.2.4).

### 6.2.3 Loss Functions

Each of the three outputs has its own loss function component: the first component, $l_f$, relates to the final/fused result compared to the building footprint mask, a second term, $l_e$, measures the agreement between the predicted and the ground truth edges and finally the $l_b$ term relates to the body part of the predicted objects. We have used a combination of the Binary Cross Entropy (BCE) loss and the dice coefficient (Eq. 6.1) to model the $l_f$ and $l_e$ components and the binary cross entropy loss (Eq. 6.2) to model the body component, $l_b$, of the loss function. The total loss is an aggregate of the three losses, with the building loss given a smaller weight than the other two components (Eq. 6.3).

$$l_{\{f,e\}} = -\frac{1}{n} \cdot \sum_{i=1}^{n} (\lambda_1 \cdot (y_i \cdot \log(\hat{y}_i) + (1 - \hat{y}_i) \cdot \log(1 - y_i)) + \lambda_2 \cdot \frac{2 \cdot y_i \cdot \hat{y}_i}{y_i + \hat{y}_i}) \tag{6.1}$$

$$l_{\{f,b\}} = -\frac{1}{n} \cdot \sum_{i=1}^{n} (y_i \cdot \log(\hat{y}_i) + (1 - \hat{y}_i) \cdot \log(1 - y_i)) \tag{6.2}$$

$$\mathcal{L} = l_f + w_1 \cdot l_e + w_2 \cdot l_b \tag{6.3}$$

where $y_i$ is the flattened model's predictions for image i (could be referring to either body or edge of final prediction depending on the context), $\hat{y}_i$ is the flattened ground truth values for image i, $n$ is the number of images per batch and $\lambda_1$ and $\lambda_2$ are weighting parameters that were set to 0.5 and 1 respectively for all training experiments for both the fused and the edge components of the loss function. The $w_1$ and $w_2$ weighting terms were experimentally set to 1 and 0.5, respectively.

### 6.2.4 Dataset

For our experiments we used the Inria Aerial Image Labelling dataset (Maggiori et al., 2017), which consists of aerial orthorectified RGB imagery of 0.3m ground resolution and the corresponding binary masks classifying each pixel as either building or not building. The dataset covers an area of 810 km$^2$ of dissimilar urban settlements from different regions around the globe captured with varying illumination conditions and at different seasons, aiming to assess the generalization capacity of the applied models.

As mentioned in Section 6.2.2 we extracted the boundaries of each building from the dataset's binary building masks to create the edge ground truth masks. We then subtracted the edges from the edge areas from the original building footprints to create the body ground truth masks (Figure 6.3). For our experiments we have randomly split the 180 images of the training dataset into a training and a validation set consisting of 145 and 35 images respectively.

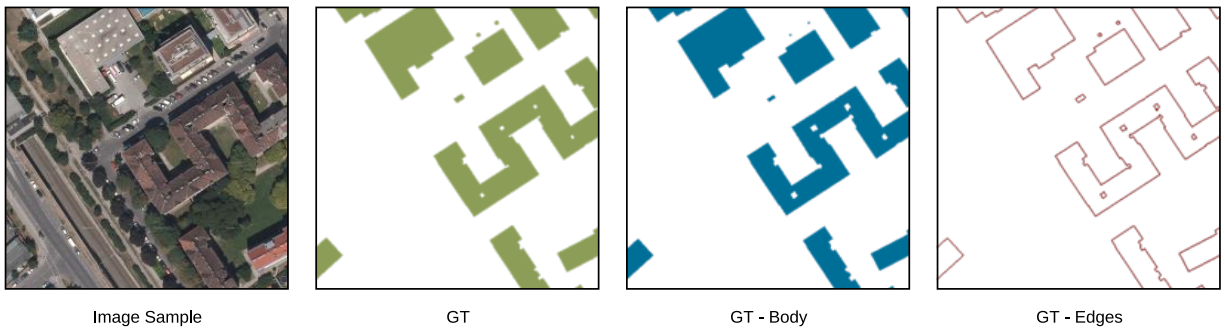



*Figure 6.3. Image and ground truth masks sample.*

### 6.2.5 Model training

For our encoder network we have used the ResNet50 architecture pretrained on ImageNet. Using an encoder with pretrained weights on a very big dataset like ImageNet is a simple and common transfer learning technique that significantly reduces training time by taking advantage of the expressive features learned on the big dataset, especially the features of the first few layers which correspond to simple visual building blocks like edges, corners and simple blob-like structures that are generally task-independent.

For the training of all models, we used the Adam optimizer with the default parameters for the coefficients of the running average (0.9 and 0.999) and without weight decay. We used a batch size of eight and a learning rate of 0.0003 for the first 54000 iterations which was then reduced to 0.0001 for the final 18000 iterations. The training was performed using the PyTorch framework (Paszke et al., 2019) on an NVIDIA Quadro RTX 5000 GPU.

### 6.2.6 Data Augmentation

Both for training and validation we used 512 × 512 pixels image samples. Since the original image size is 5000 × 5000 pixels for each training image we randomly crop 512 × 512 sample

windows that we use as input to our models. We also randomly apply horizontal and vertical flips and 90º rotations to the sampled images as well as random shifts in the interval [-20,20] to the RGB values of the image samples to help the model generalize better and reduce overfitting to the training data. For the validation set, we simply divide each image into 100 smaller images (with a very small overlap between image samples) in order to be able to monitor the training progress of a model in a consistent way.

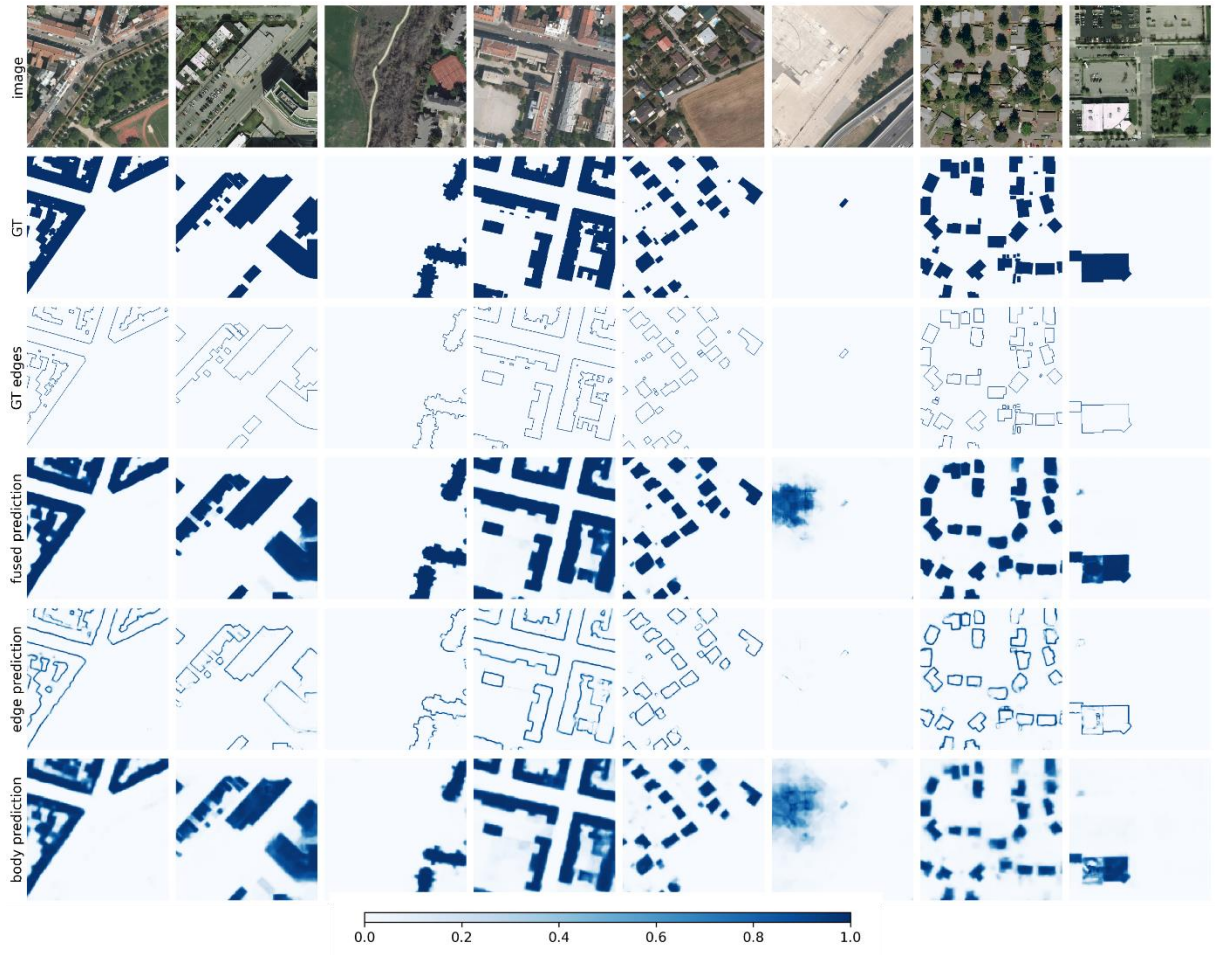



*Figure 6.4. Random image samples together with the ground truth building footprints (row 2) and building boundaries (row 3) as well as the proposed model predictions.*

## 6.3. Results

In this section we present the results for the proposed model that incorporates the decoupled body and edge module, referred as ResUNet[15] Decoupled, and compare them to the results of the two baseline approaches, referred as Plain ResUNet and Plain ResUNet with edges respectively.

Figure 6.4 presents 8 random examples of ResUNet Decoupled predictions together with the ground truth building footprints (row 2) and boundaries (row 3). The fused building footprint predictions are presented in row 4 and the edge and body predictions in rows 5 and 6 respectively. The values of the color scale correspond to the confidence of the model that a pixel belongs to the building class with 1 meaning that the pixel is most certainly a part of a building (footprint or edge) and 0 meaning that the pixel is part of the background. A visual inspection of the examples suggests that the model predicts most of the building footprints and the building boundaries successfully. In certain more challenging cases where the buildings might:

- not be fully visible,
- be located near the image boundaries (columns 2 and 4),
- have intricate structures (column 4)
- have strong shadows (column 2)

then the predictions of the fused building footprint can be blurry, and the boundaries not so well defined for certain parts of the building.

For the quantitative evaluation of the models' performance, we used the average per image values of precision, recall, F1 score and Intersection over Union (IoU) for the validation set (Equations 3.6 to 3.8 and Equation 4.3). TP (True Positive) refers to pixels that were correctly identified as buildings, TN (True Negative) refers to pixels that were correctly classified as background, FP (False Positive) are the pixels that were erroneously classified as buildings and FN (False Negative) are pixels that should have been classified as buildings but were instead classified as background.

Table 6.1 summarizes the mean evaluation metrics per image for the validation set for all three models. We can see that for most metrics (precision, F1 score and IoU) the proposed approach

[15] Network architecture described in Section 6.2. Not to be confused with ResUNet by Zhang et al. (2018).

(ResUNet – Decoupled) outperforms the baseline models by a small margin (from 1.1 to 2.7%). However, the plain ResUNet has a slightly higher recall rate (less than 0.3%) compared to the proposed approach. A visual comparison of the Decoupled semantic segmentation model's performance compared to the two baseline models is illustrated in Figure 6.5.

*Table 6.1. Mean evaluation metrics per image for the validation dataset.*

| Model | Precision | Recall | F1 | IoU |
|---|---|---|---|---|
| Plain ResUNet | 0.8457 | **0.8862** | 0.8453 | 0.7632 |
| Plain ResUNet & edges | 0.8344 | 0.8828 | 0.8314 | 0.7493 |
| ResUNet – Decoupled | **0.8617** | 0.8837 | **0.8565** | **0.7754** |

Contrary to our expectations the introduction of edges as an extra supervised task in the Plain ResUNet & edges model did not lead to improvements compared to the Plain ResUNet model but instead led to a small deterioration of the values for all four metrics. One possible explanation of this deterioration might be that the additional edge supervision makes the model less robust to misclassification errors that are present in the dataset and leads to this small degradation of the average metrics' values.

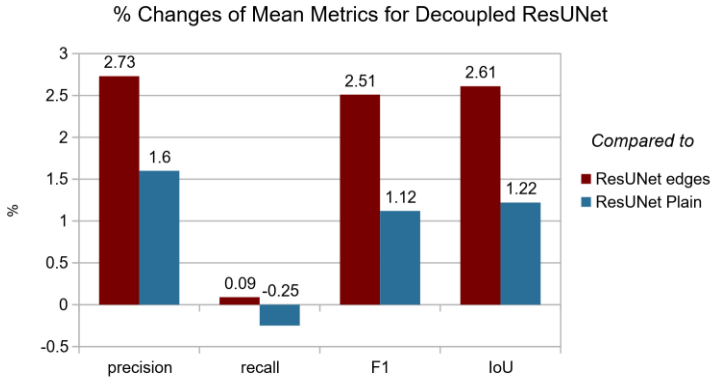



*Figure 6.5. Comparison of average evaluation metrics between the proposed architecture and the baseline models.*

Figure 6.6 presents the final predictions of the ResUNet Decoupled model and the Plain ResUNet with edges model and a comparison between the predicted and the ground truth masks. The second row presents the ground truth mask for each image. The third and fourth rows present the predicted (fused) building footprints for the proposed model (ResUNet Decoupled) and the Plain ResUNet with edges model respectively. The last two rows present a per pixel

comparison between the predicted and the ground truth binary masks. The color scale for row 3 and 4 is the same as the one in Figure 6.4.

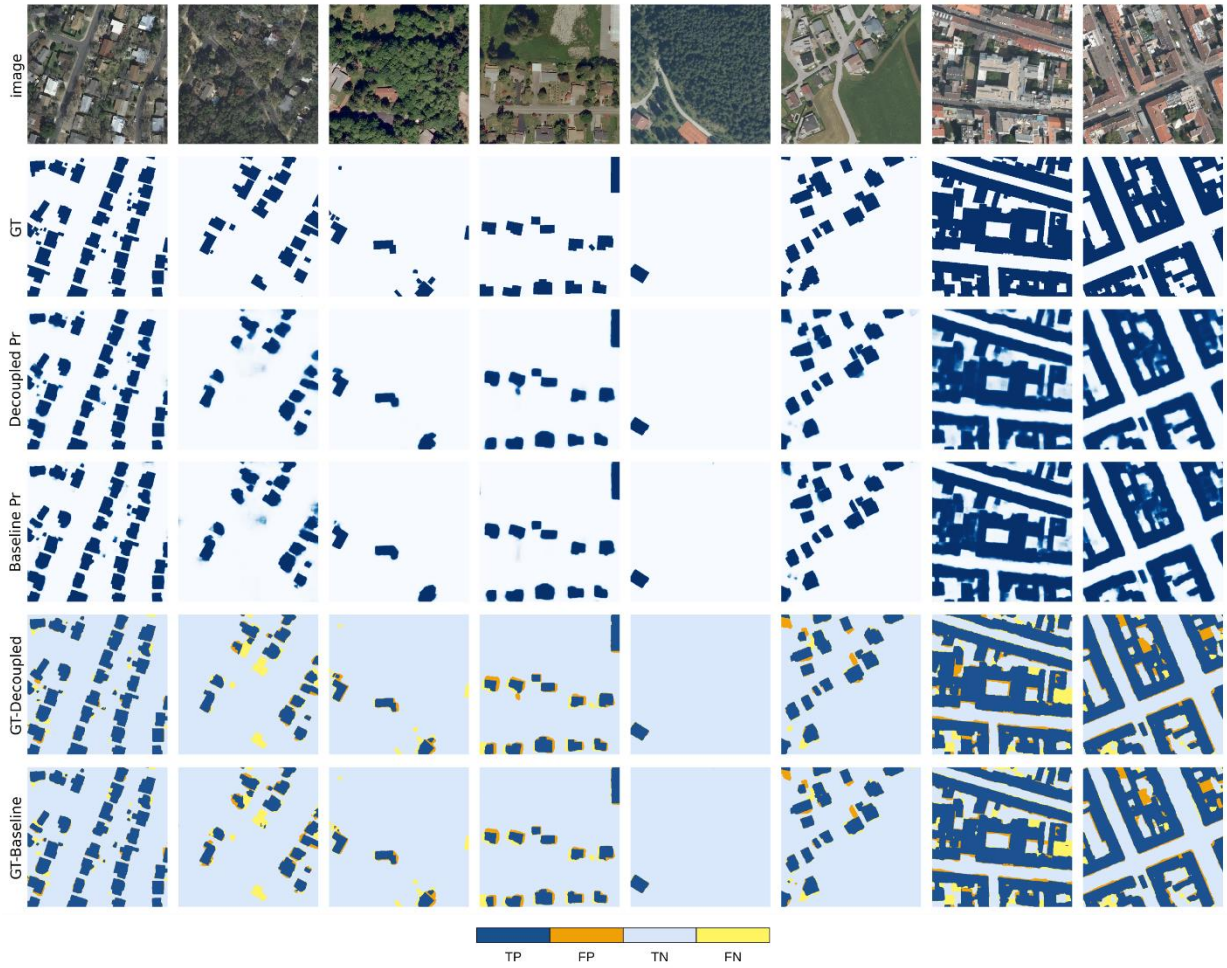



*Figure 6.6. Comparison between the predicted results and the ground truth data on random image samples.*

Based on a visual comparison of the results of the two models (Figure 6.6) there are no distinguishable benefits when using the ResUNet Decoupled architecture. It seems that the small quantitative improvements on the average evaluation metrics are not strongly reflected in the visual inspection. Both models seem to perform relatively well on predicting regularly shaped buildings, and both have small discrepancies from the ground truth when predicting the footprints of certain complicated building structures such as the examples of the last two columns. Regarding the example on the second column, we can see that the two building masks that have been classified as FN in the comparison rows for both models do not actually correspond to a building in the image but are a result of misclassification errors in the ground

truth masks. Another building on the top middle of the same image is partly misclassified as a result of tree canopy occlusions. The last two observations introduce another important aspect of the validation process which relates to the validity of the ground truth data predictions.

In Figure 6.7 we present examples that highlight cases with inconsistencies in the ground truth masks. The organization of the results and the color bar used for the comparisons are the same as Figure 6.6. Examples in columns 1, 5 and 8 illustrate cases where the ground truth mask includes building footprints for buildings that are not present in the images. In all 3 cases both models correctly classified these regions as background. In examples 3,4 and 6 we can see cases of missing footprints from the ground truth masks. Finally, in examples 2, 7 and 9 we can see cases of certain small buildings being partly or entirely occluded by the tree canopy. Although most ground truth building masks are valid there are multiple cases of inconsistencies between ground truth building footprints and the corresponding images, which cannot be easily quantified. Since most of the GT misclassification cases have been (at least partly) correctly classified by the models, an interesting future application might be the creation of a semi-automatic process for the refinement of the dataset labels.

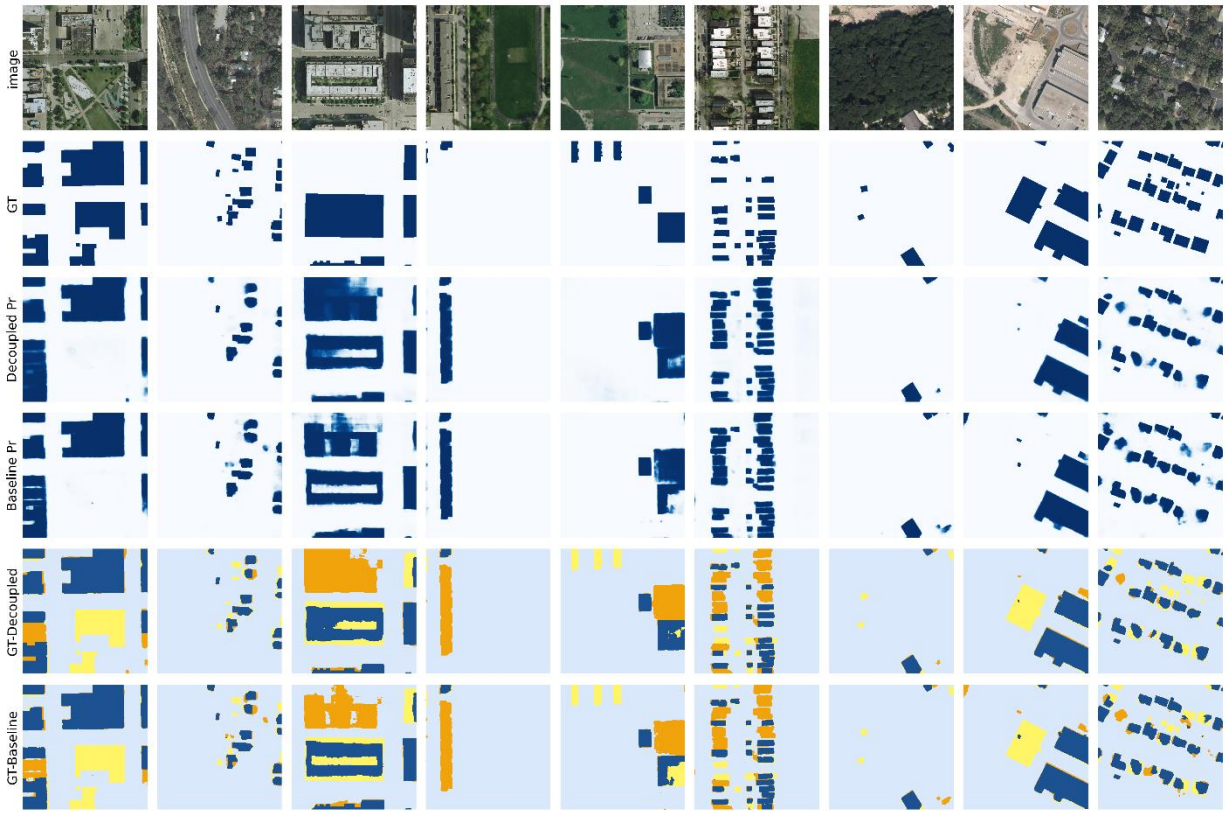



*Figure 6.7. Examples of poor ground truth building masks.*

## 6.4. Summary

In this chapter we focused on the semantic segmentation of building footprints from aerial images, which is the first component of our proposed Post-Classification Change Detection Approach. We investigated the benefits of leveraging semantic information of objects' boundaries in CNN architectures to improve the localization accuracy of the building footprints. We proposed a new end-to-end UNet/ResNet based approach that incorporates an edge and body decoupling module and showed that for most evaluation metrics it outperforms by a small margin equivalent architectures that do not explicitly model the objects' body and edges.

We will begin the next chapter by introducing our domain adaptation approach that aims to help the models trained on the Inria building semantic segmentation dataset better generalize to the imagery of each of the epochs of the change detection dataset.

# 7

# Post-Classification Change Detection: Domain Adaptation & Filtering

## 7.1. Introduction

The main drawback of the direct change detection deep learning approaches presented in chapters 3 and 4 is that they require substantial amounts of labelled training data. In many cases, even when such datasets are available, the models cannot generalize effectively to new data (target domain) that may have significantly different characteristics from the training dataset (source domain). This performance degradation that arises when applying a trained model to a new target dataset, different from the model's source domain, is due to the different characteristics between the distributions of the source and target domains, respectively. In the domain adaptation research literature, this phenomenon is referred to as domain shift.

This chapter focuses on the remaining two components of the post-classification change detection approach: a) a domain adaptation training framework inspired by Mean Teacher self-ensembling (Tarvainen & Valpola, 2017; X. Li et al., 2018, 2021; French et al., 2019), and b) a filtering process to remove prediction errors. Our main goal is to take advantage of readily available, extensive labelled training datasets, $\mathcal{D}_S$ (Source Domain), for semantic or instance segmentation to be used for training deep convolutional models that are then generalized to new unlabelled data, $\mathcal{D}_T$ (Target Domain), that may come from a distribution different from the one of the labelled dataset used for training. In other words, we are trying to minimize the domain shift between the labelled training dataset (Source Domain) and the unlabelled target dataset (Target

Domain) in order to improve the performance of the trained model on the target domain. We focus on the task of building semantic segmentation, but the presented approach could be easily extended/adjusted to any object class and various other tasks (such as classification, semantic segmentation, object detection, instance segmentation, panoptic segmentation) provided that a relevant source domain dataset is available for the supervised component of the training process.

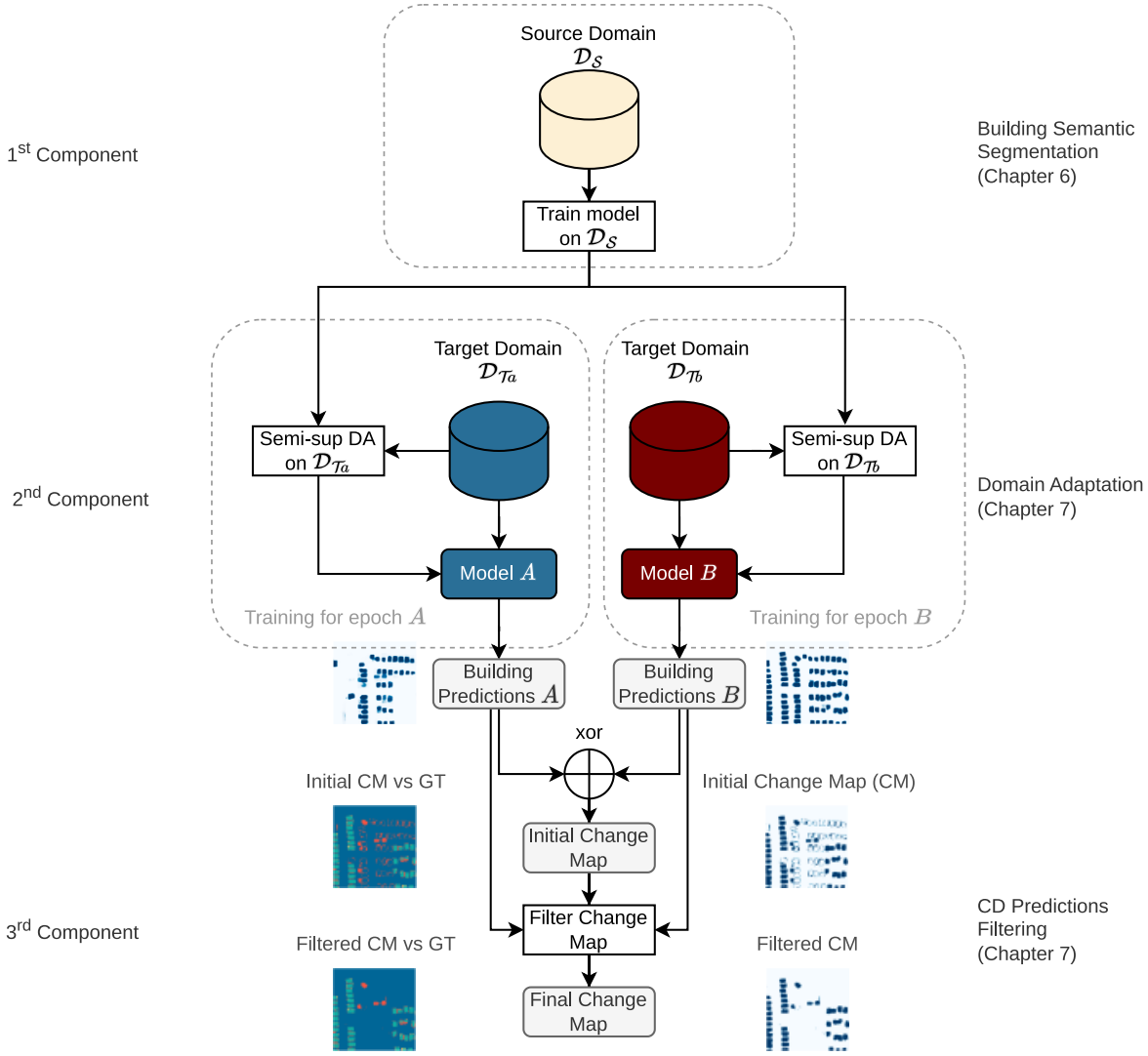



*Figure 7.1: Overview of the proposed post-classification change detection pipeline.*

An overview of our approach is illustrated in Figure 7.1. Given a change detection dataset consisting of two target domains $\mathcal{D}_{\mathcal{T}a}$ and $\mathcal{D}_{\mathcal{T}b}$, which comprise the images collected at epochs $A$ and $B$, respectively, the proposed change detection pipeline starts by training a model on the pixel-wise classification task for the classes of interest (in our case building footprints) on a source domain dataset $\mathcal{D}_S$. $\mathcal{D}_S$ should provide a plethora of labelled samples for the task at hand,

albeit not necessarily sharing the same characteristics as the elements of the target domain. For the building semantic segmentation task, we are using the decoupled body and edge ResUNet model we developed in Chapter 6 to improve the accuracy of the predicted building masks.

Next, the domain adaptation training framework is applied to each target domain $\mathcal{D}_{\mathcal{T}a}$ and $\mathcal{D}_{\mathcal{T}b}$ separately to produce the updated models $A$ and $B$, i.e., the domain-specific models for the images of epochs A and B, respectively[16]. Then, each trained model is used to compute building predictions for its corresponding epoch images. The building predictions are turned into building binary masks via a selected threshold and the produced binary outputs of epochs $A$ and $B$ are then used to create an initial change map by applying an $XOR$ operation. Finally, to improve the algorithm's performance, a filtering step is applied to the initial change maps for the removal of false positive changes. The filter is based on the assumption that similar building footprint predictions along the two instances imply unchanged regions on the ground.

First, we begin with a detailed description of the newly introduced domain adaptation approach and the pruning process we developed to filter out misclassifications in the produced building change masks. Next, we introduce the datasets we used throughout our experiments and present the training details for the different steps of the proposed pipeline. In the results section, we examine the benefits of both the proposed domain adaptation approach and the complete post-classification pipeline and conduct multiple ablation studies. We show that our method achieves significantly higher values on all the evaluation metrics compared to fully supervised and typical Mean Teacher training frameworks.

# 7.2. Methodology

## 7.2.1 Domain Adaptation

Figure 7.2 presents an overview of the proposed domain adaptation approach. Following the self-ensembling Mean Teacher concept, we introduce two different models during training: a student model, $f_\theta$, which is updated via backpropagation of the loss function $\mathcal{L}_{total}$, and a

[16] In our experiments, we need to deal with two different target domains that correspond to the data collected at different epochs, $\mathcal{D}_{\mathcal{T}a}$ and $\mathcal{D}_{\mathcal{T}b}$. However, it should be noted that the approach could be easily generalized to more than two epochs/domains.

teacher model, $f_\theta'$, an ensemble of previous instances of the student model, which is updated as an exponential moving average of the weights of the student model. At each training iteration, the model takes as input a batch of labelled samples from the source domain $\mathcal{D}_S$ and a batch of unlabelled samples from the target domain $\mathcal{D}_\mathcal{T}$. The training consists of a supervised pixel-wise classification loss, $\mathcal{L}_{sup}$ applied on the labelled samples and a consistency regularization loss term applied to both the labelled and unlabelled batches (named $\mathcal{R}_\ell$ and $\mathcal{R}_u$, respectively). The use of the consistency regularization term helps enforce consistent predictions among the teacher and student networks under various randomly selected perturbations, $\tau$, that are different for the labelled ($\tau_i$) and the unlabelled ($\tau_j$) samples. In this way it also helps the model retain useful information relating to the unlabelled samples from $\mathcal{D}_\mathcal{T}$. We have experimented with random perturbations both at the image level (image augmentations $\tau_i$ and $\tau_j$) and at the feature level (spatial dropout). Further discussion and details on the applied perturbations are provided in sections 7.2.3 and 7.3.2.

For the *Consistency Regularization Losses (CRL)* $\mathcal{R}_\ell$ and $\mathcal{R}_u$, we used a Binary Cross Entropy Loss (Equations 7.1, 7.2) instead of the Mean Square Error (MSE) loss that is commonly used with the Mean Teacher models, as it was experimentally shown to perform better (Appendix C). We hypothesize that the better performance might be due to the greater magnitude of the regularization loss of BCE compared to MSE.

$$\mathcal{R}_l = -\frac{1}{|l|}\sum_{i=1}^{|l|}(\,p_i' \cdot \log(p_i) + (1 - p_i') \cdot \log(1 - p_i)) \tag{7.1}$$

$$\mathcal{R}_u = -\frac{1}{|u|}\sum_{i=1}^{|u|}(\,p_i' \cdot \log(p_i) + (1 - p_i') \cdot \log(1 - p_i)) \tag{7.2}$$

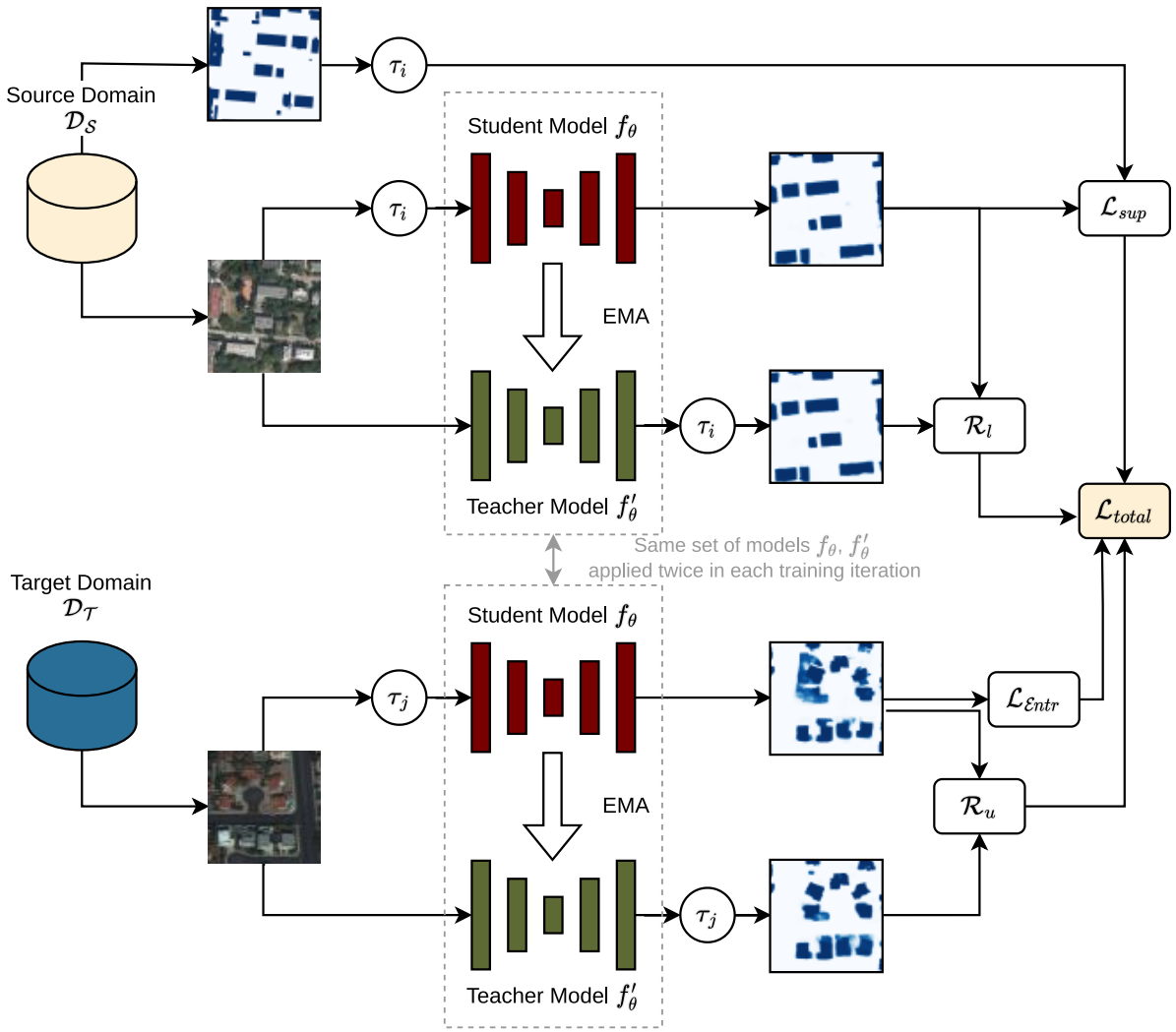



*Figure 7.2: Proposed semi-supervised self-ensembling approach using the Mean Teacher framework for building semantic segmentation domain adaptation.*

In addition to the consistency regularization term, we also introduce an entropy term inspired by (Vu et al., 2019), $\mathcal{L}_{\mathcal{E}ntr}$, which can be interpreted as a measure of how confident the student model is of its predictions on the target domain unlabelled samples. Vu et al. (2019) also suggest that the entropy loss has a similar effect to a soft-assignment version of the pseudo-label cross-entropy loss commonly used in self-training frameworks. The entropy regularization term is presented in Equation 7.3, where $p_{(i,h,w)}$ is the model prediction for a pixel in position $(h, w)$ of batch sample $i$. Given a model prediction for a pixel in position $\mathbf{x}$, $p_{\mathbf{x}} = p_{(i,h,w)}$, the binary cross entropy loss is going to be low for high and low prediction values and high for the middle prediction values, with the maximum value occurring at $p_{\mathbf{x}} = 0.5$ (Figure 7.3). Hence $L_{entr}$ will encourage the model to be more confident in its predictions. In addition, $\mathcal{L}_{\mathcal{E}ntr}$ can be used as a

measure of the quality of the model predictions, as will be discussed in more detail in Section 7.3.1.1.

$$\mathcal{L}_{entr} = -\frac{1}{|u||w||h|}\sum_{i=1}^{|u|}\sum_{h=1}^{|h|}\sum_{w=1}^{|w|}\left(p_{(i,h,w)} \cdot \log(p_{(i,h,w)}) + (1 - p_{(i,h,w)}) \cdot \log(1 - p_{(i,h,w)})\right) \quad (7.3)$$

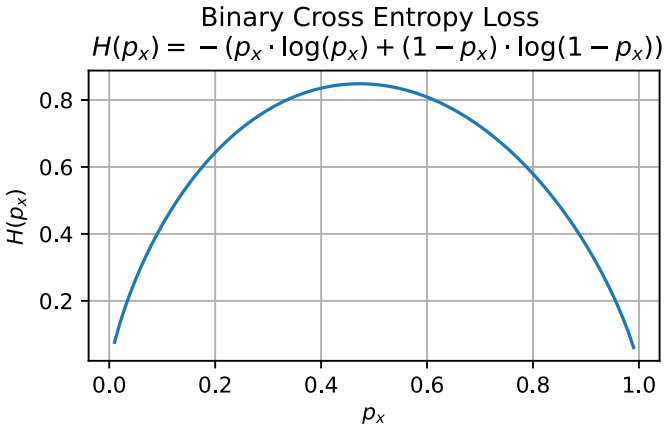



*Figure 7.3: Entropy loss curve using the natural logarithm.*

The final loss function (Equation 7.4) is a weighted aggregate of the supervised loss component, $\mathcal{L}_{sup}$, the two regularization losses, $\mathcal{R}_{\ell}$ and $\mathcal{R}_{u}$, and the unsupervised Entropy loss $\mathcal{L}_{entr}$, where $\mu(t)$ is a weighting coefficient that may be a function of the number of epochs that increases as the training proceeds and the model's predictions become more accurate. Since in our implementation we used a model pre-trained on the source domain, we found that setting $\mu(t)$ to a constant value of 0.5 worked well for the domain adaptation experiments. The domain adaptation pipeline is summarized in Algorithm 7.1.

$$\mathcal{L}_{total} = \mathcal{L}_{sup} + \mu(t) \cdot (\mathcal{R}_{l} + \mathcal{R}_{u} + \mathcal{L}_{entr}) \quad (7.4)$$

**Algorithm 1:** Domain Adaptation Training Framework

**Input:**

$\{x_i, y_i\} \in \mathcal{D}_\mathcal{S}$ /* Labelled samples */

$\{x'_i\} \in \mathcal{D}_\mathcal{T}$ /* Unlabelled samples */

$f_\theta$ /* Model pretrained on $\mathcal{D}_\mathcal{S}$ */

iter_n, $|l|, |u|, \lambda, \mu(i)$ /* training parameters */

**Output:** $f_\theta, f'_\theta$ /* Trained Student and Teacher models */

Load student model $f_\theta$ ;

Set teacher model $f'_\theta = f_\theta$ ;

$DL_S \leftarrow$ `Dataloader`$(\mathcal{D}_\mathcal{S})$ ; /* Create Dataloader for $\mathcal{D}_\mathcal{S}$ */

$DL_T \leftarrow$ `Dataloader`$(\mathcal{D}_\mathcal{T})$ ; /* Create Dataloader for $\mathcal{D}_\mathcal{T}$ */

**for** $i \in \{1, 2, \ldots,$ iter_n$\}$ **do**

  /* Get labelled and Unlabelled minibatches */

  $x_l \leftarrow$ `next`$(DL_S)$ ;

  $x_u \leftarrow$ `next`$(DL_T)$ ;

  /* Create random augmentations for each sample in each minibatch */

  Random augmentations $\tau_l = \{\tau_1, \ldots, \tau_{|l|}\}$ ;

  Random augmentations $\tau_u = \{\tau'_1, \ldots, \tau'_{|u|}\}$ ;

  /* Predictions for $\mathcal{D}_\mathcal{S}$ */

  $p_l \leftarrow f_\theta(\tau_l(x_l))$ ;

  $p'_l \leftarrow \tau_l(f'_\theta(x_l))$ ;

  /* Predictions for $\mathcal{D}_\mathcal{T}$ */

  $p_u \leftarrow f_\theta(\tau_u(x_u))$ ;

  $p'_u \leftarrow \tau_u(f'_\theta(x_u))$ ;

  /* Supervised Loss */

  $\mathcal{L}_{sup} = -\frac{1}{|l|}\sum_{i=1}^{|l|}(\lambda \cdot (y_i \cdot \log(p_i) + (1 - y_i) \cdot \log(1 - p_i)) + \frac{2y_i p_i}{y_i + p_i})$ ;

  /* Consistency Regularization Terms */

  $\mathcal{R}_l = -\frac{1}{|l|}\sum_{i=1}^{|l|}(p'_i \cdot \log(p_i) + (1 - p'_i) \cdot \log(1 - p_i))$ ;

  $\mathcal{R}_u = -\frac{1}{|u|}\sum_{i=1}^{|u|}(p'_i \cdot \log(p_i) + (1 - p'_i) \cdot \log(1 - p_i))$ ;

  /* Entropy Regularization */

  $\mathcal{L}_{entr} = -\frac{1}{|u|}\sum_{i=1}^{|u|}(p_i \cdot \log(p_i) + (1 - p_i) \cdot \log(1 - p_i))$ ;

  /* Total Loss */

  $\mathcal{L}_{total} = \mathcal{L}_{sup} + \mu(t) \cdot (\mathcal{R}_l + \mathcal{R}_u + \mathcal{L}_{entr})$ ;

  /* Weights Update */

  update $f_\theta$ weights using the Adam optimizer ;

  update $f'_\theta$ weights using EMA ;

**end**

**def** `next` *(Dataloader)* **:**

  **if** *Dataloader has batches to return* **then**

    return batch;

  **else**

    reinitialize *Dataloader* ;

    return batch ;

  **end**

**end**

*Algorithm 7.1: Domain Adaptation Training Framework.*

In the case of the decoupled body and edge ResUNet model, the supervised loss function comprises the body, edge and fused loss components as described in the previous chapter and is computed according to Equation 6.3. However, the loss function selection has changed compared to Chapter 6. For the fused result and the body loss components we used the combination of BCE and Dice loss (Equation 6.1). For the edge predictions, we chose a class balanced Binary Cross Entropy loss (Equation 7.5), where $w$ is the inverse frequency of the pixel classified as boundaries in the training set, in order to counter the great imbalance in the distribution of the boundary/no boundary pixels[17].

$$\mathrm{l_e} = -\frac{1}{\mathrm{n}}\sum_{i=1}^{n}(w \cdot y_i \cdot log(\hat{y}_i) + (1-w) \cdot (1-y_i) \cdot log(1-\hat{y}_i)) \tag{7.5}$$

### 7.2.2 Filtering Model Predictions for Post-classification Change Detection

Following the application of the Mean Teacher training framework for the domain adaptation of the pre-trained model on the two target domains $\mathcal{D}_{\mathcal{T}a}$ and $\mathcal{D}_{\mathcal{T}b}$ corresponding to the two epochs of the change detection dataset, we use the adapted models $A$ and $B$ to estimate the building footprint predictions for each epoch. We then compare the building predictions to extract the changes in the building coverage. When examining the initial change maps produced by applying the *XOR* operator on the building predictions, we noticed, in many cases, artifacts which were mainly caused by either small errors in the co-registration of the two instances or partial misclassifications of individual building footprints (as can be seen in the example presented in Figure 7.4 – top of column 5). One straightforward way to address this issue is to try and consider each building separately and to check whether each building footprint that is present in the building map of one of the two epochs is also classified as a building in the other epoch's prediction.

Figure 1.4 shows different stages of the post-classification and change filtering process for a sample image pair. The first column shows the two image instances; the second column shows

[17] This is common practice in the training of CNN for edge detection as can be seen for example in (J. He et al., 2022; Xie & Tu, 2015)

the model predictions for instances A (top) and B (bottom); in the third column, the building instances that have an IoU higher than 50% are highlighted in red: these buildings are considered to be matched and thus the respective pixels cannot be classified as changed; column 4 shows the ground truth change map (GT CM); In column 5 we present the initial change maps and the final change maps and finally in column 6 we can see the TP, TN, FP and FN for the initial (top) and the final (bottom) change maps.

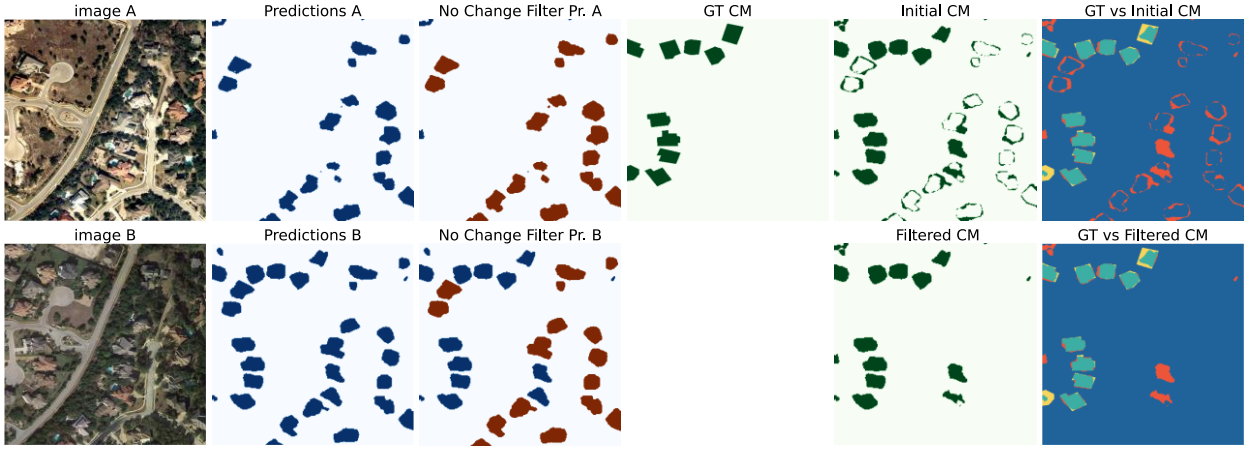



*Figure 7.4: Different stages of the post-classification change detection approach that highlight the need for a filtering step based on the overlap between corresponding buildings.*

Thus, in order to further improve the quality of the predicted changes, we propose a filtering algorithm that makes this check for each region classified as a building in either of the two epochs (Algorithm 7.2). The process takes as input the binary building predictions for an image sample, $P$ and $\tilde{P}$, where $P$ corresponds to the predictions for epoch $A$ and $\tilde{P}$ to the ones for epoch B, and the initial change map $C$. The first step is to create sets $\mathcal{R}$ and $\tilde{\mathcal{R}}$, containing all separate building regions (or building objects) in each predicted mask based on pixel connectivity (two pixels are connected if they are neighbours and have the same value). Next, for each region in the set $\mathcal{R}$, the algorithm retrieves the region's bounding box in $P$ and the corresponding bounding box in mask $\tilde{P}$ and computes the Intersection over Union (IoU) between the two masks. If the IoU value is above a specified threshold (which, for our experiments, was set to 0.5), then the region is considered to be unchanged and is added to the unchanged regions set $\mathcal{U}$. The same process is followed for the regions in $\tilde{\mathcal{R}}$, which results in a second unchanged regions set $\tilde{\mathcal{U}}$. Then, by combining the unchanged regions' information from sets $\mathcal{U}$ and $\tilde{\mathcal{U}}$, we can derive a single mask $U_f$ that contains all pixel locations that should be considered unchanged. Finally,

mask $U_f$ is applied on the original change map $C$ in order to filter out the regions that are likely misclassified as buildings mainly due to misregistration errors between the image pair and minor classification errors in the semantic segmentation of individual buildings.

**Algorithm 2:** Filter Change Predictions

**Data:** Binary Building Predictions $P$ and $\tilde{P}$
Matching threshold $t$
**Result:** Filtered Change Map $C_f$

Initial Change Map $C \leftarrow P \oplus \tilde{P}$ ; /* xor operator */
/* Find separate building regions based on pixel connectivity */
$\mathcal{R} \leftarrow$ regions($P$), where $\mathcal{R} = \{r_1, \ldots, r_n\}$;
$\tilde{\mathcal{R}} \leftarrow$ regions($\tilde{P}$), where $\tilde{\mathcal{R}} = \{\tilde{r}_1, \ldots, \tilde{r}_m\}$;

**foreach** $r_i \in \mathcal{R}$ **do**
  Find $BB_i \leftarrow \text{bbox}(r_i)$ ;
  $iou_i = \text{IoU}(r_i, \tilde{P}_{BB_i})$ ;
  **if** $iou_i > t$ **then**
    Append $r_i$ to the unchanged regions set $\mathcal{U}$ ;
  **end**
**end**
Create binary mask $U$ for unchanged regions of set $\mathcal{U}$ ;

**foreach** $\tilde{r}_i \in \tilde{\mathcal{R}}$ **do**
  Find $BB_i \leftarrow \text{bbox}(\tilde{r}_i)$ ;
  $iou_i = \text{IoU}(\tilde{r}_i, P_{BB_i})$ ;
  **if** $iou_i > t$ **then**
    Append $\tilde{r}_i$ to the unchanged regions set $\tilde{\mathcal{U}}$ ;
  **end**
**end**
Create binary mask $\tilde{U}$ for unchanged regions of set $\tilde{\mathcal{U}}$ ;

Combine $U$ and $\tilde{U}$ into a single mask $U_f = U \vee \tilde{U}$ ; /* logical or */
Filter out False Positive pixel: $C_f = (\neg U_f) \wedge C$ ; /* logical and */

*Algorithm 7.2: Filtering of the change predictions.*

### 7.2.3 Experiments

This section provides an overview of the datasets that were used for both the domain adaptation of building semantic segmentation and the post-classification change detection experiments and describes the dataset preprocessing and setup for each experiment. It also provides some details pertaining to the models' training and validation.

#### 7.2.3.1 Datasets

Our experiments were based on three datasets:

- The Inria Aerial Image Labelling dataset (Maggiori et al., 2017),
- The LEVIR-CD remote sensing Image change detection dataset (H. Chen & Shi, 2020) and
- The SpaceNet Challenge 2 (or SpaceNetV2) dataset (Van Etten et al., 2019).

In all cases, Inria was used as our source domain $\mathcal{D}_S$ and for pretraining the models for the building semantic segmentation task. For the first set of experiments that focused on the domain adaptation of building semantic segmentation, the four different cities from the SpaceNet Challenge 2 were used as four different target domains. Finally, we used the LEVIR-CD dataset in the post-classification change detection experiments.

The Inria Aerial Image Labelling dataset is the building semantic segmentation dataset we introduced in Chapter 6 and used for the experiments presented in Section 6.2.

The LEVIR-CD building change detection dataset was introduced in Chapter 5 and used in the second set of experiments presented in Section 5.3.2.

**SpaceNetV2**

The SpaceNetV2 dataset was developed for the extraction of building footprints from DigitalGlobe WorldView 3 satellite images (SpaceNet Challenge 2). The source imagery comprises the original 30cm resolution panchromatic band, the 1.24m resolution 8-band multi-spectral composite (as 11-bit geotiff files), and a 30cm resolution Pan-Sharpened 3-band and 8-band composites (as 16-bit geotiff files) (Van Etten et al., 2019). For our experiments, we only used the pan-sharpened RGB composite images.

The dataset covers four different cities from around the world – Vegas, Shanghai, Paris, and Khartoum – and includes both urban and suburban areas. The images are made available in tiles of 650 × 650 pixels (200m × 200m) and are accompanied by their respective building footprints in a vector polygon format.

#### 7.2.3.2 Training Details

The ResUNet Decoupled Body and Edge (DEB) architecture was pre-trained on the Inria Aerial Image Labelling dataset as described in Section 6.2.5 and in (Bousias Alexakis & Armenakis, 2022).

For the semi-supervised component of the training, we used the Adam Optimizer with the coefficients of the running average set to 0.9 and 0.999, respectively. The learning rate, $lr$, was initially set to 0.0001 and was reduced by 0.9 every 1,000 iterations ($lr_t = 0.9 \cdot lr_{t-1}$). Preliminary experiments of our domain adaptation training suggested that the training curves converged after about 10,000 iterations. Hence, unless stated otherwise – which is only the case in some ablation studies – the results correspond to models trained for 10,000 iterations. The batch size for both training and evaluation of the models was set to 8. A discussion on the effects of the batch size on the evaluation metrics' values is provided in Appendix E.

For the image perturbations, we used a combination of random horizontal and vertical flips, random 90-degree rotations, random cropping and rescaling and random shifts of the RGB image values in the range $[-20,20]$ and a random brightness and contrast transformation. We used the *Albumentations* python library (Buslaev et al., 2020) in our implementation as it provides an easy way to store the parameters of the applied transformations for later use, a feature that facilitated the implementation of our semi-supervised training algorithm.

The model implementation and training were developed using the PyTorch framework (Paszke et al., 2019) on an NVIDIA Quadro RTX 5000 GPU.

#### 7.2.3.3 Data preprocessing

All models were trained using 512 × 512 pixels image input samples. Since the original image size of the Inria dataset images is 5000 × 5000 pixels, we randomly crop 512 × 512 sample windows that we use as input to our models (this is also a form of data augmentation which is performed before the augmentations described in the previous section). The exact process was also followed for the training set of the SpaceNetV2 dataset, where the original image size is 650 × 650 pixels, and for the LEVIR-CD dataset, where the original image size is 1024 × 1024 pixels.

For the SpaceNetV2 samples, we also transformed the RGB 16-bit geotiff files into 8-bit PNG image files and the corresponding building footprints into binary masks using the rasterio[18] and geopandas[19] libraries.

We created our four target domain datasets for the building semantic segmentation task by randomly sampling 1,000 images from each city of the SpaceNetV2 dataset. However, many samples from the SpaceNet dataset had very large portions of the image left blank (blank/black regions in the image), which was not the case in any of the samples of the Inria dataset, the dataset we used as our Source (or labelled) dataset for our experiments. Thus, before selecting the 1,000 images from each region of the SpaceNet dataset, we first filtered out all images where the percentage of pixels missing ground information exceeded 20% of the total image. It should be noted that the number of available images after the filtering process for Khartoum was 724 and not 1,000.

## 7.3. Results and Discussion

We begin by examining the benefits of the proposed domain adaptation approach on the building semantic segmentation task. Next, we present a series of ablation studies on the domain adaptation training framework, followed by the results of our complete post-classification change detection approach.

### 7.3.1 Domain Adaptation

As a first step, we compare the building semantic segmentation results of the model trained according to the complete Mean Teacher training framework – that includes the decoupled edge and body architecture, spatial dropout and the additional entropy regularization loss term – to the models trained solely on the source domain dataset in a supervised way and to the initially trained models using a simplified Mean Teacher framework (no spatial dropout feature-level perturbations and no entropy regularization loss term).

The results for the final DA approach, as well as for the MT and the supervised training approaches using both the Decoupled Body and Edge (DEB) and a plain ResUNet architectures,

[18] https://github.com/rasterio/rasterio

[19] https://github.com/geopandas/geopandas

are presented in Table 7.1. Figure 7.5 displays the difference between the proposed approach and the rest of the models for each city. Both the table and figure results suggest that the proposed approach (1st row for each dataset/city) performs better than all the other training frameworks for three out of four cities: Vegas, Paris, and Khartoum. In the Shanghai dataset, our approach has the best recall rate, by more than 4% from the second best, which is the MT framework with the DEB architecture, and has the second-best performance on the accuracy, F1 and IoU metrics with a difference smaller than 1%. In general, therefore, it seems that the entropy regularization term and the feature-level perturbations via spatial dropout led to improved performance compared to applying the MT framework with only image/input-level perturbations. A more detailed presentation of the effects of the entropy regularization term and the use of spatial dropout can be found in Section 7.3.2.

The presented average metrics of Table 7.1 also indicate that training using the MT semi-supervised framework leads to significant improvements compared to the model being trained in a fully supervised manner solely on the source domain, with the benefits in a lot of cases exceeding 20% for most metrics in all four different target domains/cities.

*Table 7.1: Model performance on the 4 target datasets for different network architectures and training techniques.*

| City | Architecture | Training | Criterion | dropout | entropy | accuracy | precision | recall | F1 | IoU |
|---|---|---|---|---|---|---|---|---|---|---|
| Vegas | DEB | semi-sup | Teacher | ✓ | ✓ | **0.9475** | **0.8174** | 0.9356 | **0.8720** | **0.7738** |
| | DEB | semi-sup | # iter. | ✗ | ✗ | 0.9433 | 0.8103 | 0.9200 | 0.8608 | 0.7566 |
| | plain | semi-sup | # iter. | ✗ | ✗ | 0.9396 | 0.7957 | 0.9238 | 0.8537 | 0.7464 |
| | DEB | sup | # iter. | ✗ | ✗ | 0.9084 | 0.7029 | 0.9108 | 0.7917 | 0.6575 |
| | plain | sup | # iter. | ✗ | ✗ | 0.8582 | 0.5815 | **0.9426** | 0.7172 | 0.5615 |
| Shanghai | DEB | semi-sup | Teacher | ✓ | ✓ | 0.9101 | 0.6169 | **0.8414** | 0.6983 | 0.5495 |
| | DEB | semi-sup | # iter. | ✗ | ✗ | **0.9163** | 0.6477 | 0.7990 | **0.7031** | **0.5533** |
| | plain | semi-sup | # iter. | ✗ | ✗ | 0.9084 | **0.6726** | 0.6640 | 0.6478 | 0.4955 |
| | DEB | sup | # iter. | ✗ | ✗ | 0.8776 | 0.5460 | 0.4284 | 0.4538 | 0.3108 |
| | plain | sup | # iter. | ✗ | ✗ | 0.8406 | 0.4404 | 0.7238 | 0.5338 | 0.3751 |
| Paris | DEB | semi-sup | Teacher | ✓ | ✓ | **0.9584** | **0.6036** | **0.7588** | **0.6674** | **0.5355** |
| | DEB | semi-sup | # iter. | ✗ | ✗ | 0.9573 | 0.5912 | 0.7490 | 0.6558 | 0.5227 |
| | plain | semi-sup | # iter. | ✗ | ✗ | 0.9560 | 0.6022 | 0.7119 | 0.6410 | 0.5043 |
| | DEB | sup | # iter. | ✗ | ✗ | 0.8814 | 0.3395 | 0.5571 | 0.3979 | 0.2614 |
| | plain | sup | # iter. | ✗ | ✗ | 0.9098 | 0.3943 | 0.5019 | 0.4196 | 0.2812 |
| Khartoum | DEB | semi-sup | Teacher | ✓ | ✓ | **0.8808** | 0.6826 | **0.7610** | **0.7162** | **0.5610** |
| | DEB | semi-sup | # iter. | ✗ | ✗ | 0.8757 | 0.6958 | 0.6746 | 0.6797 | 0.5195 |
| | plain | semi-sup | # iter. | ✗ | ✗ | 0.8564 | **0.7561** | 0.4275 | 0.5373 | 0.3740 |
| | DEB | sup | # iter. | ✗ | ✗ | 0.7681 | 0.4373 | 0.4915 | 0.4539 | 0.2986 |
| | plain | sup | # iter. | ✗ | ✗ | 0.7894 | 0.4900 | 0.8146 | 0.6079 | 0.4399 |

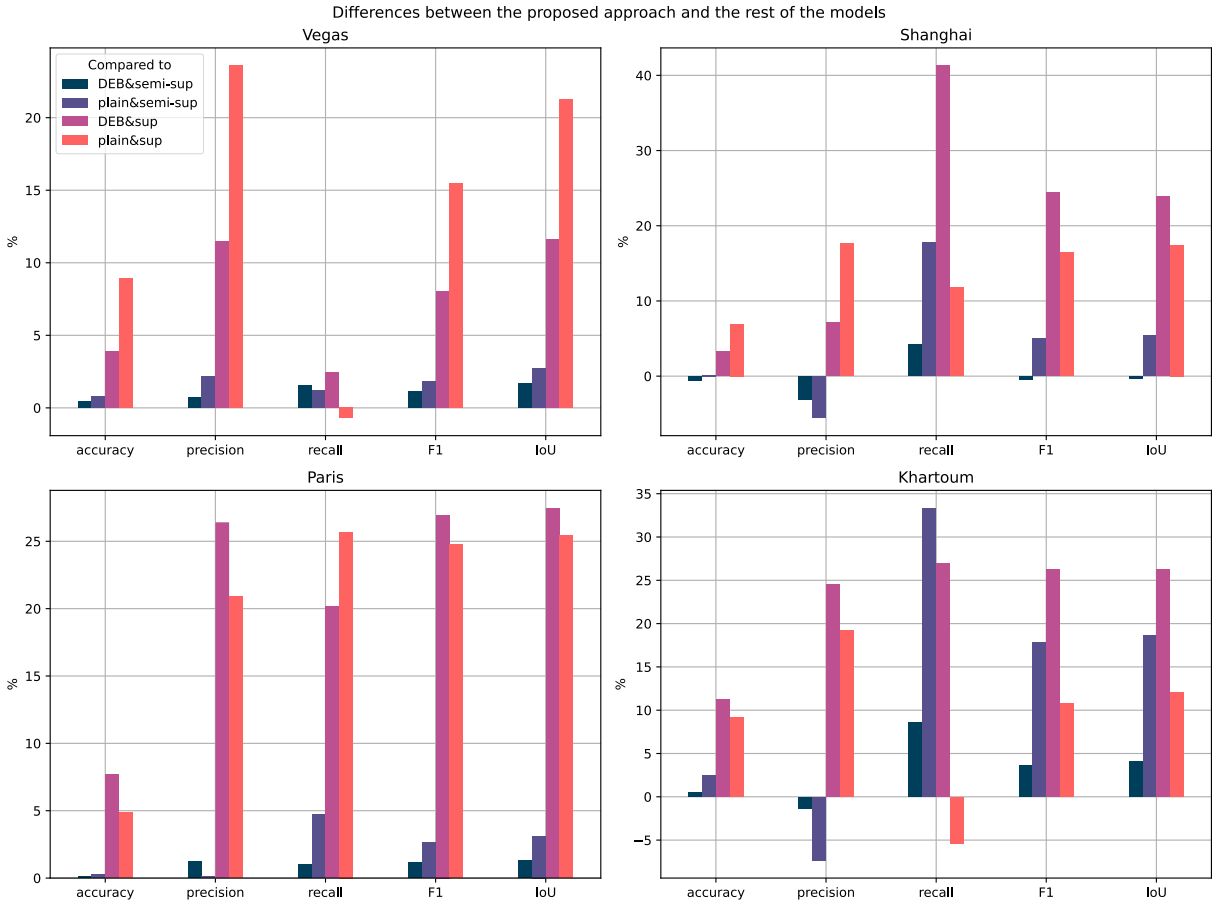



*Figure 7.5: Difference between the proposed semi-supervised DA approach and the rest of the approaches presented in Table 7.1.*

By comparing the results of the DEB and plain architectures for the same target domains and training frameworks (rows 2:3 and 4:5, respectively), we can observe a consistent pattern regarding the effects of the model's architecture on the evaluation metrics and its relation to the training framework. When during training we used the DA self-ensembling approach, the Decoupled Body and Edge architecture consistently outperforms the plain ResUNet architecture, but in the fully supervised case, it is the plain architecture that performs better in all four target datasets. It can thus be suggested that the more complex DEB model tends to overfit the source domain training data more than the simpler ResUNet architecture. The semi-supervised training helps reduce the DEB model's overfitting to the labelled training data thanks to the consistency regularization loss components and the strong image and feature perturbations. It also helps the models to incorporate characteristics of the target domain into the feature representations, leading to significantly better performance on the target datasets.

#### 7.3.1.1 Best Model Criterion

The three different criteria for selecting the final (best) model in the absence of any labelled samples in the target domain were evaluated. Table 7.2 summarizes the results on the various target domains where for each target domain, the evaluation metrics were estimated for the following four models:

- The estimated Mean Teacher model, which as an ensemble of multiple instances of the model is expected to produce more accurate results than the student.
- The instance of the student model that achieved the lowest average entropy value during training on the unlabelled target domain (as described in Equation 7.3).
- The last computed student model during the semi-supervised training, and
- The model with the maximum average IoU value on the target domain. This model would not have been available in an actual unsupervised domain adaptation scenario where the target domain does not have any labelled samples. However, in our case, it can serve as an upper bound for the average IoU.

*Table 7.2: Evaluation metrics for different best model criteria. All the presented results were produced by models trained on the best training scenario using spatial dropout and entropy minimization.*

| City | Criterion | accuracy | precision | recall | F1 | IoU |
|---|---|---|---|---|---|---|
| Vegas | Teacher | 0.9475 | 0.8174 | **0.9356** | 0.8720 | 0.7738 |
| | Min Entropy | 0.9494 | 0.8316 | 0.9223 | 0.8742 | 0.7771 |
| | Max IoU | **0.9513** | **0.8425** | 0.9175 | **0.8780** | **0.7830** |
| | 10000 Iter. | 0.9452 | 0.8103 | 0.9306 | 0.8659 | 0.7641 |
| Shanghai | Teacher | 0.9101 | 0.6169 | 0.8414 | 0.6983 | 0.5495 |
| | Min Entropy | **0.9224** | 0.6801 | 0.7714 | 0.7101 | 0.5621 |
| | Max IoU | 0.9223 | 0.6683 | 0.8145 | 0.7219 | **0.5764** |
| | 10000 Iter. | 0.9190 | 0.6594 | 0.8012 | 0.7106 | 0.5628 |
| Paris | Teacher | 0.9584 | 0.6036 | 0.7588 | 0.6674 | 0.5355 |
| | Min Entropy | 0.9590 | 0.6008 | **0.7776** | **0.6728** | **0.5422** |
| | Max IoU | **0.9613** | **0.6207** | 0.7419 | 0.6721 | **0.5422** |
| | 10000 Iter. | 0.9587 | 0.5951 | 0.7728 | 0.6680 | 0.5376 |
| Khartoum | Teacher | **0.8808** | 0.6826 | 0.7610 | 0.7162 | 0.5610 |
| | Min Entropy | 0.8700 | **0.7243** | 0.5704 | 0.6326 | 0.4672 |
| | Max IoU | 0.8791 | 0.6649 | **0.8044** | **0.7253** | **0.5712** |
| | 10000 Iter. | 0.8687 | 0.6695 | 0.6841 | 0.6713 | 0.5099 |

In addition to Table 7.2, we also include a visual overview of the results, which is presented in Figure 7.6. In general, the IoU values of all three models (MT, Min Entropy, and last iteration

model) are close –they differ less than 2% from each other in almost every case – with the exception of Khartoum, where the MT model performs better than the other two models by around 5 and 10%. However, the minimum entropy model achieves better results in the remaining three cities, albeit only 0.5 to 1% better than the MT model. In Vegas, Shanghai, and Paris, all three models produce IoU values within 1% of the maximum IoU, whereas in Khartoum, only the MT model is close to 1% off. Overall, both the minimum entropy and the Mean Teacher model could be used to produce the results, as no approach outperformed all others in all cases. The MT as an exponential moving average of multiple model weights seems to have a more consistent performance, but the minimum entropy model outperformed MT in three of the four datasets, though slightly.

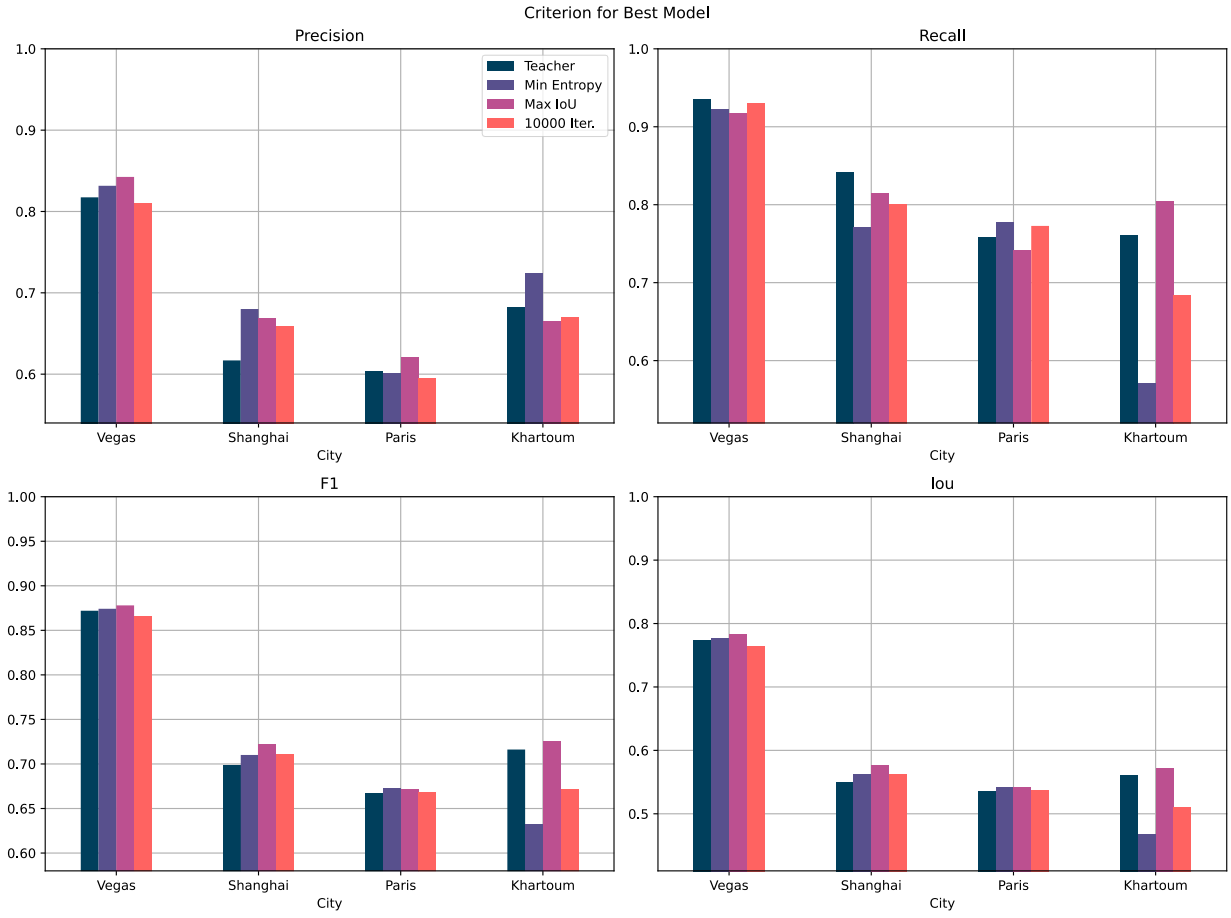



*Figure 7.6: Evaluation metrics for different best model criteria on the four target domain datasets/cities.*

### 7.3.2 Domain Adaptation – Ablation Studies

In the following set of experiments, we investigate the effects of different training techniques and design choices on the performance of the proposed domain adaptation training framework. We experiment with various types of perturbations, namely CutMix, Spatial Dropout, and a newly introduced feature perturbation involving the feature statistics of Batch Normalization layers. We also assess the introduction of two additional consistency regularization components based on the edge and body predictions of the proposed decoupled body and edge ResUNet architecture.

#### 7.3.2.1 CutMix – Spatial Dropout

Many recent works on semi-supervised learning with self-ensembling and the Mean Teacher paradigm have highlighted the importance that image and feature perturbations play in the training process (French et al., 2019; S. Li et al., 2021; X. Li et al., 2018; Tarvainen & Valpola, 2017). First, we investigated the use of spatial dropout and CutMix to improve the performance of our training method.

French et al. (2019) suggest that the use of CutMix improves the model's performance when training using the Mean Teacher framework for semi-supervised semantic segmentation. Thus, inspired by their work, we experimented with applying a CutMix augmentation (our implemented variation of CutMix is presented in Appendix D) in addition to the other image augmentations described in Section 7.2.3.

In the general case, dropout layers (Srivastava et al., 2014) are used during training to improve the generalization performance of a model by preventing feature activations from becoming strongly correlated, which can lead to overfitting the training model. However, Tompson et al. (2015) showed that the application of standard dropout layers, where each activation of a convolution feature map is set to 0 independently with a probability $p_i$, did not prevent overfitting on fully convolutional networks and increased training time. This is due to the strong spatial correlation between neighbouring image pixels and, consequently, between neighbouring feature activations within a feature map. They proposed spatial dropout, a new dropout formulation that applies dropout on entire feature activation maps and increases the generalization performance on fully convolutional networks.

In our implementation, we introduced a Spatial Dropout layer with a probability of 0.3 – meaning that 30% of the feature channels will be randomly turned off during training – to the outputs of the Body and Edge Decoupling sub-module and to the level 0 features – i.e., the activations after the first batch normalization layer that are passed as high-level features together with the output of the Body and Edge Decoupling sub-module to the Decoupled Attention and Feature Fusion sub-module (see Figure 6.2).

Our experiments did not verify the findings of (French et al., 2019). Instead, they suggested that introducing CutMix in the training process has negligible effects on the model performance on the validation (target) set. A comparison of the training and validation IoU curves of a model trained with and without CutMix augmentations (Figure 7.8) shows that CutMix had a slight negative effect on the metric values after the first 10,000 iterations. We also compared the validation metrics for multiple models trained with and without CutMix (Table 7.3) on both the Vegas and Shanghai datasets and were not able to find any significant benefits derived from the application of the CutMix augmentation. The introduction of Spatial Dropout layers (Tompson et al., 2015), on the other hand, led to better IoU values on the Vegas dataset and generally to a more stable IoU validation curve (Figure 7.7).

*Table 7.3: Validation results for the Vegas and Shanghai target domains with and without the use of CutMix augmentation for the final models according to all three best model selection criteria. All models were trained using the DEB architecture, spatial dropout, and the additional entropy regularization term.*

| City | Best Model Selection | CutMix | accuracy | precision | recall | F1 | IoU |
|---|---|---|---|---|---|---|---|
| Vegas | Teacher | ✓ | **0.9483** | **0.8218** | 0.9317 | **0.8728** | **0.7750** |
| Vegas | Teacher | ✗ | 0.9475 | 0.8174 | **0.9356** | 0.8720 | 0.7738 |
| Vegas | 10000 Iter. | ✓ | 0.9414 | 0.7960 | **0.9341** | 0.8589 | 0.7534 |
| Vegas | 10000 Iter. | ✗ | **0.9452** | **0.8103** | 0.9306 | **0.8659** | **0.7641** |
| Vegas | Min Entropy | ✓ | 0.9486 | 0.8271 | **0.9244** | 0.8726 | 0.7745 |
| Vegas | Min Entropy | ✗ | **0.9494** | **0.8316** | 0.9223 | **0.8742** | **0.7771** |
| Shanghai | Teacher | ✓ | 0.9084 | **0.6224** | 0.8048 | 0.6880 | 0.5371 |
| Shanghai | Teacher | ✗ | **0.9101** | 0.6169 | **0.8414** | **0.6983** | **0.5495** |
| Shanghai | 10000 Iter. | ✓ | 0.9053 | 0.5994 | **0.8363** | 0.6871 | 0.5340 |
| Shanghai | 10000 Iter. | ✗ | **0.9190** | **0.6594** | 0.8012 | **0.7106** | **0.5628** |
| Shanghai | Min Entropy | ✓ | 0.9186 | 0.6608 | **0.7726** | 0.7001 | 0.5500 |
| Shanghai | Min Entropy | ✗ | **0.9224** | **0.6801** | 0.7714 | **0.7101** | **0.5621** |

The IoU values of the training and validation curves presented in Figures 7.7 to 7.12 are smoothed by an Exponential Moving Average (EMA) with a smoothing factor of 0.3. The solid lines correspond to the smoothed IoU values using the EMA, while the faint lines represent the initially estimated values. By validation curves, we refer to the results on the target domain (i.e., the domain for which no labels were provided to the models during training).

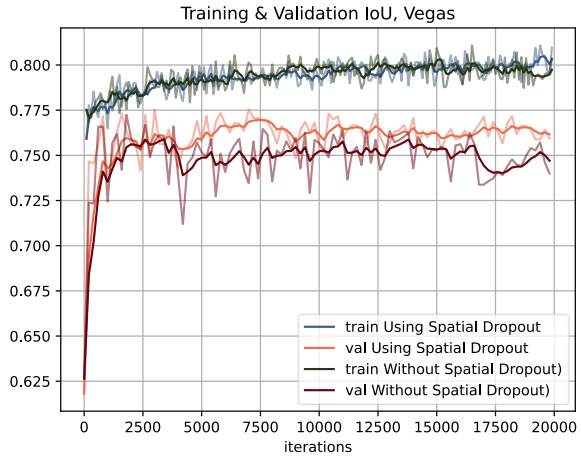



*Figure 7.7: Training and Validation IoU curves for the model trained either with or without spatial dropout layers on Vegas.*

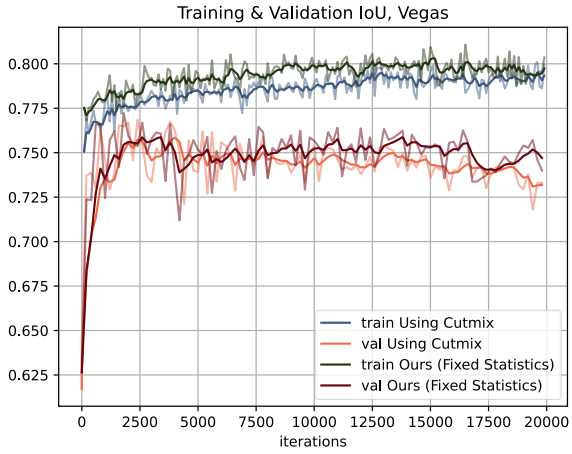



*Figure 7.8 : Training and Validation IoU curves for models trained either with or without using CutMix on the Vegas validation (target) dataset.*

### 7.3.2.2 Batch Normalization

In Appendix A, we introduce batch normalization and review certain proposed adaptations to the batch normalization layers that have been shown to reduce the domain shift in domain adaptation applications. Consistency regularization approaches, including ours, feed samples from both source and target domains to the model and thus, the batch normalization layers' statistics should incorporate information from both domains.

The purpose of this experiment is:

- to test whether batch normalization, as described in Ioffe & Szegedy (2015), allows the model to generalize to the target domain under a consistency regularization semi-supervised training scenario, and

- to compare the results of the traditional batch normalization training to the ones of the implementation we proposed in Appendix A, in which we keep the batch normalization statistics fixed and thus turn the batch normalization layers into learnable linear transformations of the input feature channels.

We performed the experiment twice using Vegas as the target dataset:

- the first time we used the augmentations described in Section 7.2.3.2 (Figure 7.9),
- and the second time, we also introduced CutMix augmentations with a probability of 0.5 (Figure 7.10).

Figures 7.9 and 7.10 indicate that in both experiments, our approach leads to higher and more stable IoU metrics on the validation (Target) curves compared to the application of standard batch normalization layers.

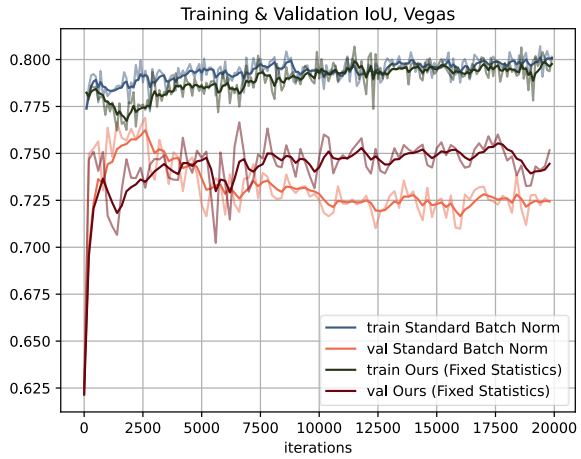



*Figure 7.9: Training & validation IoU curves for a model trained using Batch Nomralization compared to a model trained with all Batch Normalization Layer statistics "frozen". Trained on the Vegas dataset without CutMix.*

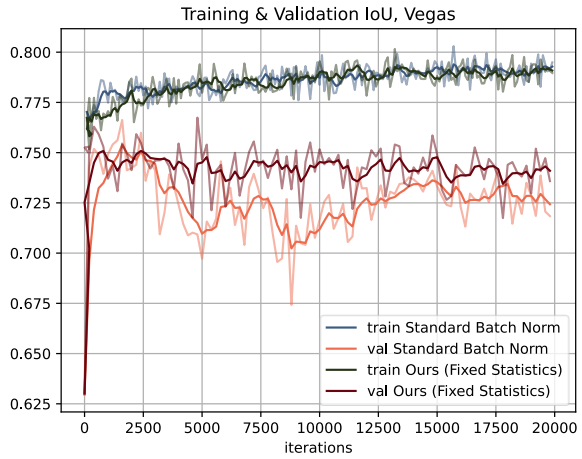



*Figure 7.10: Training & validation IoU curves for a model trained using Batch Nomralization compared to a model trained with all Batch Normalization Layer statistics "frozen". Trained on the Vegas dataset using CutMix as an extra augmentation.*

### 7.3.2.3 Extra Consistency Regularization Loss (CRL) Terms

In the results section of Chapter 6, as well as in (Bousias Alexakis & Armenakis, 2022), it has been shown that separating the body and edge information of semantic segmentation masks and training a model to attend separately to the objects' main body and boundaries besides the original masks improves the model's performance. Thus, a question arises regarding the

utilization of the boundary and body information in the domain adaptation process by introducing additional consistency regularization constraints to the body and edge outputs of the model.

The effects of the additional consistency regularization terms for the models' body and edge outputs were tested on the Vegas dataset. We performed the experiment twice:

- the first time we used the augmentations described in Section 7.2.3.2 (Figure 7.11),
- and the second time we also introduced CutMix augmentations with a probability of 0.5 (Figure 7.12).

The validation IoU curves of both experiments suggest that the additional regularization terms do not affect the solution in any substantial manner. Thus, for the sake of simplicity, we did not include these extra terms in any other experiment.

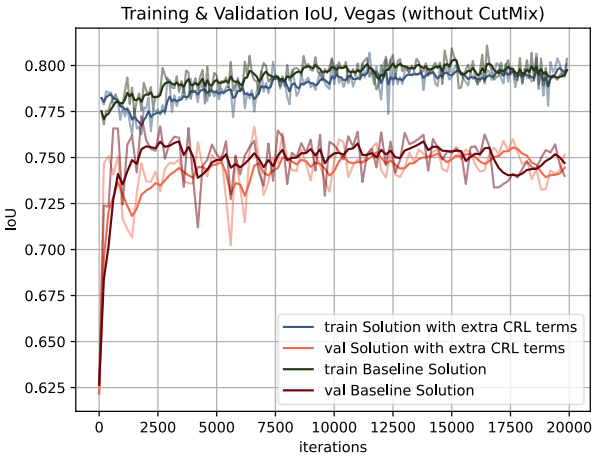



*Figure 7.11: Training & validation IoU curves for a model trained using additional body and edge CRL terms compared to a model trained using CRL only for the fused predictions on the Vegas dataset. Trained without CutMix.*

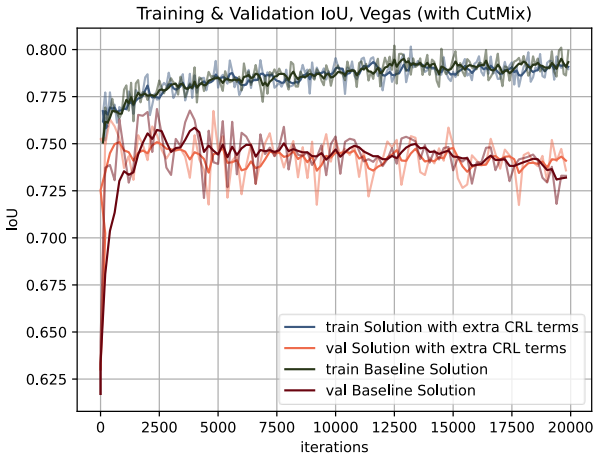



*Figure 7.12: Training & validation IoU curves for a model trained using additional body and edge CRL terms compared to a model trained using CRL only for the fused predictions. Both models were trained on the Vegas dataset using CutMix as an extra augmentation.*

### 7.3.3 Post-Classification Change Detection

In this section, we present and discuss the results of the complete proposed post-classification building change detection approach using an enhanced MT training framework for the domain adaptation of building semantic segmentation.

Table 7.4 summarizes the post-classification change detection evaluation results on the LEVIR-CD target dataset. The No Filter (top) section of the table provides the results for each run without the application of the proposed filtering pipeline. It is followed by the Filter Section, which presents the results for the same runs after filtering. In the last row of the table, we include the results retrieved by training the model using the direct CD approach[20] with full access to the labelled training data of the LEVIR-CD dataset. This final run can be used to compare the performance of our approach, which does not use any labelled training samples from the CD dataset and thus can be considered unsupervised with respect to the CD task, to the performance of a model of similar architecture with full access to the labelled training data.

The results of Table 7.4 highlight the benefits of the proposed DA semi-supervised training framework compared to training the model solely on the Source Domain (Inria dataset) in a fully supervised manner. All metrics improve significantly, with the highest benefits observed on the recall rate, with the difference being higher than 25% for the best semi-supervised model retrieved according to the minimum entropy criterion.

The filtering step also leads to steady benefits, mainly on the precision metric, which consistently increases between 5 and 6 % for all runs compared to the respective results retrieved before the filter application. This was the expected behaviour as the filtering step was designed to reduce the false positive changes caused by misregistration and small misclassification errors.

Figure 7.13 presents the building predictions for each of the two image instances, the change prediction maps and a comparison between the predicted and the ground truth change maps for both the filtered and the unfiltered change results. Based on a visual inspection of the randomly selected samples, it seems that the process is able to detect most of the changed building regions (except for the example in column 2) and that the filtering step successfully removes most of the artifacts in change maps caused by minor misregistration errors between the building predictions of the two images and small errors caused by the partial misclassification of some building footprints in one of the two images.

[20] The fully supervised approach presented in Chapter 3 trained on the LEVIR-CD dataset. We used the ResUNet architecture presented in Section 6.2.1 for the encoder-decoder model.

*Table 7.4: Change Detection results using the post-classification CD approach with and without domain adaptation. All models were trained using the Decoupled Body and Edge (DEB) ResUNet architecture.*

| | Training | Best Model Selection | dropout | entropy | accuracy | precision | recall | F1 | IoU |
|---|---|---|---|---|---|---|---|---|---|
| No filter | Semi-sup | Teacher | ✓ | ✓ | **0.9621** | **0.5614** | 0.8392 | **0.6545** | **0.5055** |
| | Semi-sup | Min Entropy | ✓ | ✓ | 0.9606 | 0.5379 | **0.8753** | 0.6508 | 0.4984 |
| | Semi-sup | # Iter. | ✓ | ✓ | 0.9451 | 0.4455 | 0.8946 | 0.5812 | 0.4226 |
| | Semi-sup | # Iter. | ✗ | ✗ | 0.9621 | 0.5690 | 0.7874 | 0.6407 | 0.4880 |
| | sup | # Iter. | ✗ | ✗ | 0.9547 | 0.5163 | 0.6158 | 0.5350 | 0.3796 |
| Filter | Semi-sup | Teacher | ✓ | ✓ | **0.9682** | **0.6202** | 0.8330 | 0.6940 | **0.5506** |
| | Semi-sup | Min Entropy | ✓ | ✓ | 0.9681 | 0.6001 | **0.8688** | **0.6966** | 0.5494 |
| | Semi-sup | # Iter. | ✓ | ✓ | 0.9539 | 0.4941 | 0.8870 | 0.6227 | 0.4646 |
| | Semi-sup | # Iter. | ✗ | ✗ | 0.9671 | 0.6190 | 0.7839 | 0.6732 | 0.5240 |
| | sup | # Iter. | ✗ | ✗ | 0.9585 | 0.5562 | 0.6131 | 0.5566 | 0.4006 |
| *Model trained in a supervised way on the target domain using the direct CD approach* | | | | | 0.9871 | 0.8751 | 0.8537 | 0.8625 | 0.7610 |

Figure 7.14 displays the change detection predictions compared to the ground truth masks for three models with and without the application of the filtering step. As in Figure 7.13, we can once again see the benefits of the filtering process in removing a lot of False Positive artifacts in all three models. In addition, the MT training approach significantly improves the predictions' quality compared to the model trained in a supervised way on the Source Domain. Finally, the results in columns 1,3 and 7 suggest that the proposed approach, which incorporates the entropy regularization term and the spatial dropout perturbations, performs better than the plain MT training framework based on image-level perturbations.

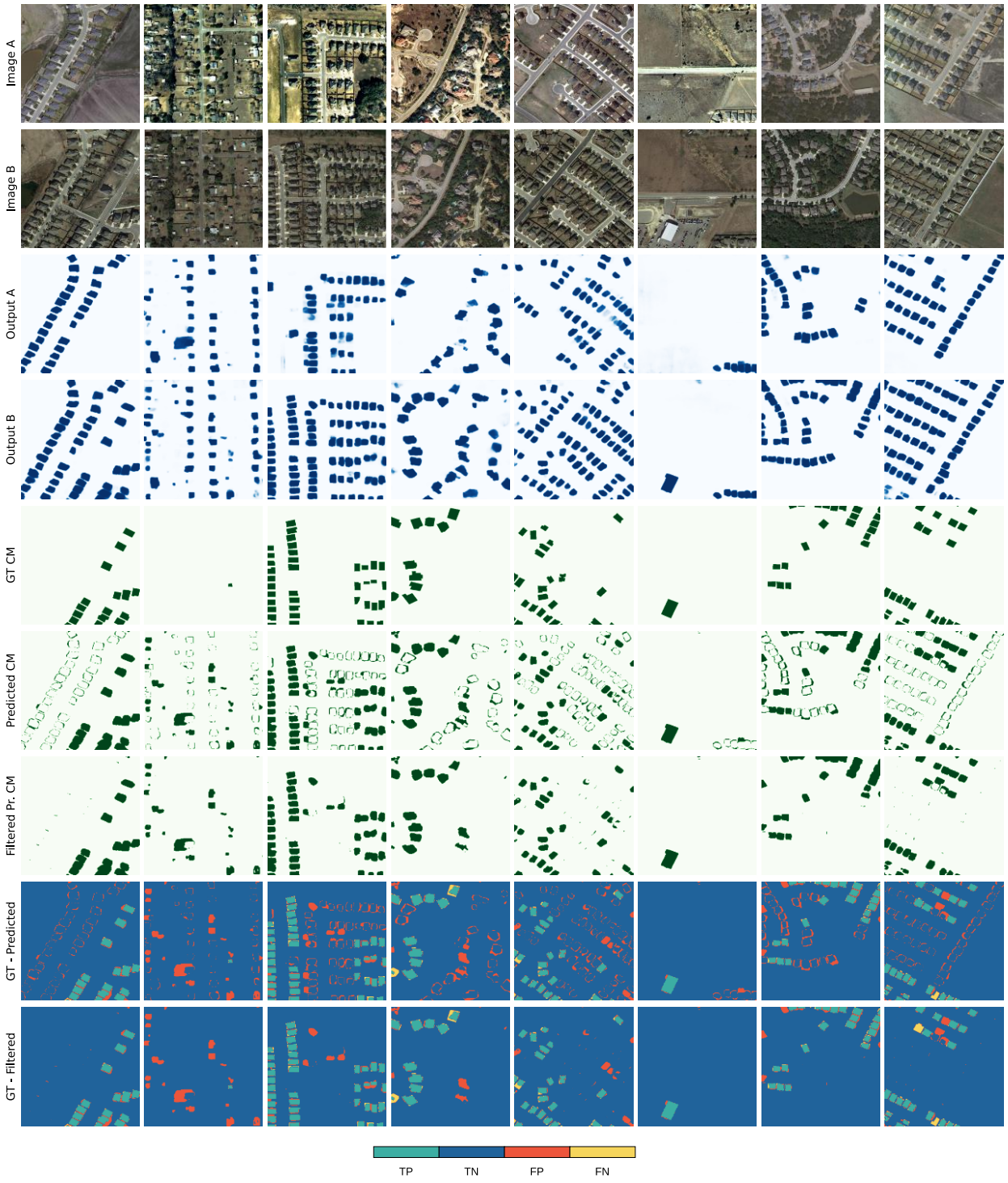



*Figure 7.13: Building predictions, change predictions and comparisons between the change predictions and the ground truth masks for 8 sample image pairs. From top to bottom: Instance A, Instance B, Building Prediction of instance A, Building Prediction of instance B, Ground Truth Change Map, Predicted Change Map, Filtered Predicted Change Map, Comparison Between the Ground Truth and the Predicted Change Maps, Comparison Between the Ground Truth and the Filtered Predicted Change Maps.*

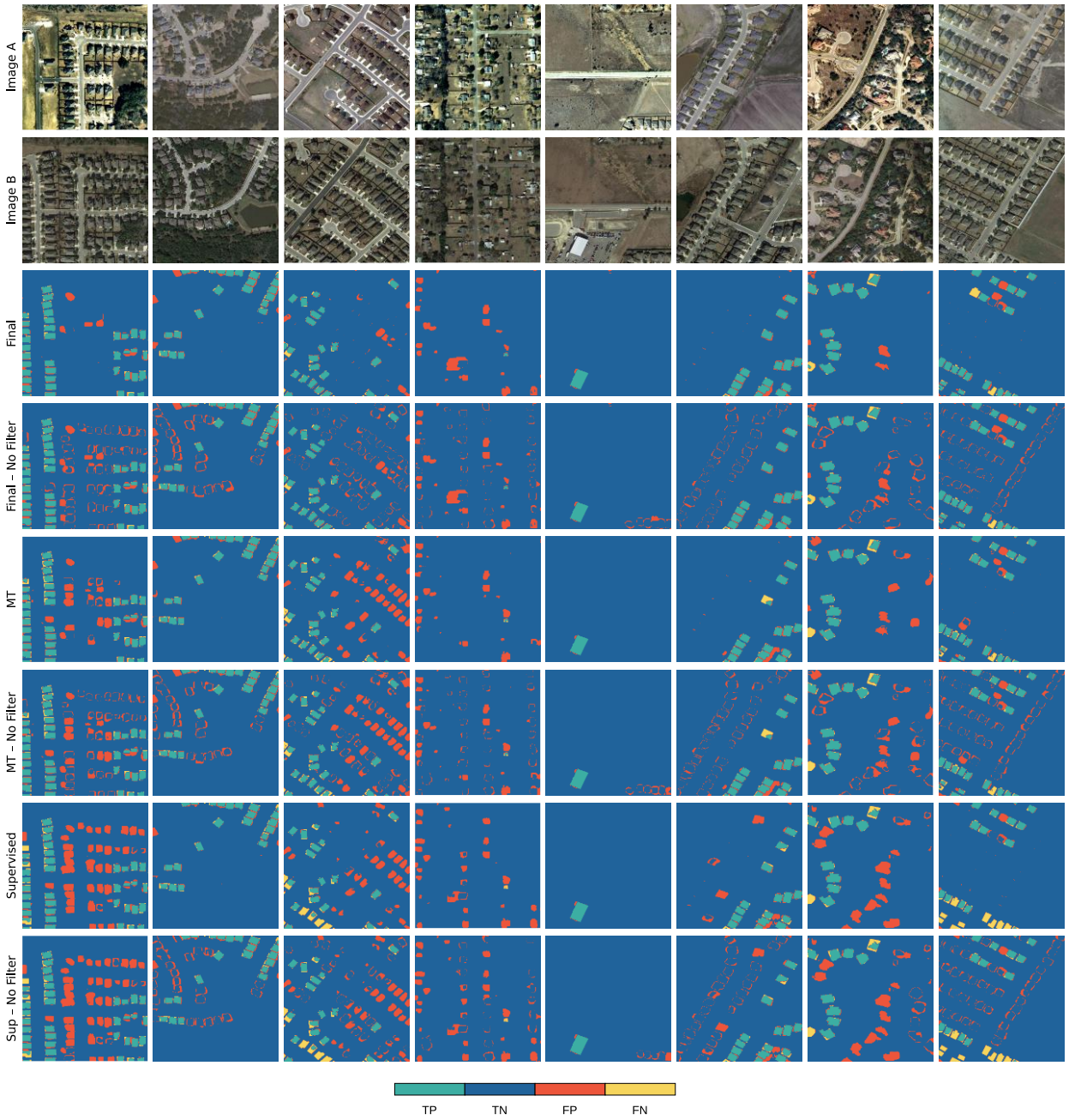



*Figure 7.14: Change Detection predictions compared to the ground truth masks for multiple models with and without the application of the filtering step (random samples from Levir-CD). From top: Instance A, Instance B, Proposed Approach with filtering step, Proposed Approach without filtering step, model trained using MT training with image augmentations – with filtering, model trained using MT training with image augmentations – without filtering, model trained using supervised training on the Source Domain (Inria Datatset) – with filtering, model trained using supervised training on the Source Domain (Inria Datatset) – without filtering.*

## 7.4. Summary

To conclude, in this chapter we developed a novel CNN-based change detection pipeline that eliminates the need for labelled change maps when training the models by utilizing large, readily available pixel-wise classification datasets.

We showed that the proposed Mean Teacher training framework, based on a consistency assumption and enhanced by an additional entropy regularization soft constraint, can be a very effective unsupervised domain adaptation approach that leads to significant improvements both in the case of building semantic segmentation and in the building change detection task. In addition, the introduction of additional perturbations in the models' feature representations, such as the introduction of spatial dropout layers and the use of different statistics in the batch normalization layers of the student and the teacher models, have a significant positive effect on the method's performance.

Both qualitative and quantitative results indicate that all three components of the proposed post-classification change detection pipeline demonstrably contribute to the improvement of the method's performance.

# 8

# Conclusions

This research work focused on developing and assessing novel deep learning-based algorithms for change detection applications from high resolution imagery, inspired by the great success of deep learning and Convolutional Neural Networks in the fields of computer vision and remote sensing.

We investigated and proposed two primary approaches for CNN-based change detection algorithms. A direct change detection approach, where the deep learning model takes as input a multitemporal image pair covering the same region and outputs the change map prediction, and a post-classification change detection approach which first performs semantic segmentation on each instance of the image pair independently and then compares/combines the produced predictions in order to detect the changed regions between the images.

Within these two methods we addressed the lack of training data via the introduction of domain adaptation and semi-supervised training frameworks, we improved the detail and localization accuracy of the predicted changes by incorporating the semantic information edges, and increased the model's robustness to co-registration errors by introducing separate geometric and radiometric augmentations among the image pairs.

## 8.1. Direct Change Detection

First, for the direct change detection approach we experimented with two U-shaped architectures – UNet and Nested UNet – and with the application of data augmentations, deep supervision, and different loss functions – a combination of BCE and Dice loss and the Lovász Hinge Loss – aiming to explore the effects of each design option on the performance of the change detection algorithm and to identify the combination that leads to the best performance. The analysis of the results led to the following conclusions:

- The evaluation results of the UNet++ architecture consistently outperformed the ones retrieved using UNet by a small margin (less than 1%) on all assessment metrics (accuracy, precision, recall, and F1 score).
- The results showed that the choice of loss function had a bigger impact on the metrics' average values compared to the choice of network architecture. The combination of BCE with Dice coefficient loss outperformed Lovász Hinge Loss on both UNet and UNet++ architectures on all four metrics in the case of the UNet architecture and on precision, F1 score and accuracy in the case of the UNet++ architecture.
- Using data augmentation consistently led to better performance on all tested metrics.
- Deep supervision had a positive effect on the evaluation metrics when combined with the Lovász Hinge Loss and data augmentation, but a negative effect otherwise.

To further improve the predictions' localization accuracy and detail of U-shaped encoder-decoder architectures and their robustness to co-registration errors two new approaches/methods are proposed and assessed (Chapter 4). First, to enhance the model's semantic segmentation accuracy, we introduced a second CNN model – based on the DexiNed architecture – that separately predicts the edges for each instance of the image pair and then inserted the predicted boundary information into multiple points of the network. Furthermore, in order to increase the model's robustness to mis-registrations between the instances of the image pair, we introduced additional geometric (random affine transformations or rotations) and radiometric noise into the second instance of the training image pair, thus implicitly teaching the network to ignore small image alignment errors. Further we evaluated our approach on the UNet and UNet++ architectures and experimented with multiple loss functions to find the one that works best for each scenario.

Overall, our findings suggest that both proposed enhancements improved the networks' average prediction performance for each of the tested models. The results' analysis indicates that the use of semantically informed edges in the models improves the average IoU metric by 2%, the average recall rate and F1 score by approximately 1.7%, and the precision metric by about 1%, in comparison to using the plain backbone architectures without any boundary information both for UNet and UNet++ architectures and BCE Dice and Lovasz Hinge loss functions. Hence, the

results corroborate our hypothesis that incorporating boundary information into a U-shaped architecture can improve the models' change detection performance, albeit by a small margin.

The findings regarding the effect of the training process proposed to increase the models' robustness to co-registration errors among the image pair showed that our method improved the performance of all models on average by about 2% on the IoU metric, 1.7% on the recall rate and F1 score, and around 1% on the precision metric. The advantages of the method are most noticeable in image pairs with more significant alignment errors - with the gain in IoU exceeding 5% in some cases - while when considering image pairs with minor alignment errors, many models – excluding most models based on UNet++, which continue to report gains, albeit of a smaller magnitude – exhibit a slight performance degradation of around 1%.

To reduce the need for large-scale training datasets – which require costly and time-consuming manual or semi-automatic annotation processes and constitute a considerable limitation in deep learning applications – a semi-supervised approach is being proposed (Chapter 5). It has been designed to address the need for extensive labelled training data when training deep learning models for change detection applications It uses information from unlabelled image pairs via an additional consistency regularization loss function component, inspired by the Mean Teacher training framework. This extra component derives from comparing the outputs of a student and teacher model that were presented with the same input images but subjected to a different set of perturbations/augmentations, thus enforcing a soft consistency constraint on the models' predictions and helping the models retain useful information from the image pairs without relying on training labels.

We conducted two sets of experiments using different design parameters on two datasets to assess the potential benefits of the proposed semi-supervised approach. Even though the results of the first set of experiments did not suggest any discernible improvements on the evaluation metrics compared to a fully supervised approach, the second set of experiments, performed on the more extensive LEVIR-CD change detection dataset, clearly indicated enhanced semantic segmentation performance when using the proposed training framework. Using the semi-supervised training method resulted in higher evaluation metrics on every training scenario – where we used different portions of the training samples' labels (5, 10, and 20%) to simulate scenarios with varying numbers of labelled samples available for training – and for every metric,

with the most notable improvement (6%) reported on the recall rate. Perhaps the most compelling finding of the second experiment that highlights the potential of the proposed approach was that a model trained according to the Mean Teacher framework was able to approximately match the performance of a fully supervised model while using only 20% of the labelled training samples.

## 8.2. Post-classification Change Detection

Chapters 6 and 7 focus on the development and evaluation of a new post-classification change detection approach. The method was designed to address the lack of labelled training data for change detection as well as the significant degradation in the models' performance when faced with data that have different characteristics from the training domain (known as domain shift). The key strength of this approach lies in the fact that it does not require any change detection labels to train the models but relies instead on the semantic segmentation of each image instance separately. In this way, it can take advantage of much larger training datasets that cover multiple regions around the globe and can lead to models that generalize more effectively. Another advantage of the method is that it can focus on specific types of changes (e.g., building coverage or certain types of vegetation), provided that a respective training dataset for the specified class is available.

The proposed method consists of the following main components:

- A new deep learning model for building semantic segmentation. To refine the accuracy and detail of the building semantic segmentation, we utilize the objects' boundary information and propose a new ResUNet architecture that decouples the features' body and edge information, processes them in discrete streams, and then amalgamates them into a final fused prediction.

- A novel domain adaptation training framework that addresses the domain gap between the training dataset and each of the two sets of images corresponding to one of the periods of the change detection dataset. The proposed unsupervised domain adaptation approach was inspired by the Mean Teacher training method and introduced an additional entropy regularization term that aims to reinforce more confident model predictions.

- A process that filters out False Positive detections from the predicted change masks that are due to small misclassification or misregistration errors.

With regard to the first component of the method, the experimental results on the Inria building semantic segmentation dataset (Section 6.3) showed that the proposed decoupled body and edge (DEB) ResUNet architecture marginally outperformed a baseline ResUNet architecture on the precision, F1, and IoU metrics (by 1.6, 1.12 and 1.22 %, respectively). Similar results were obtained in the building segmentation domain adaptation experiments of Section 7.3.1, where the DEB models trained in a semi-supervised manner consistently outperformed the corresponding regular ResUNet models in the accuracy, precision, F1 score, and IoU metrics. Nevertheless, the plain ResUNet architectures performed better in three of the four datasets when trained solely in a supervised manner, suggesting that the more complex DEB architecture tends to overfit the source domain unless trained according to the proposed domain adaptation framework.

In terms of the second component of our approach, before assessing the method for the change detection task, we conducted a series of experiments devised to finetune and evaluate the proposed domain adaptation training framework. We tested the approach on the building semantic segmentation of four different datasets – corresponding to the four cities of the SpaceNetV2 dataset – and in all four cases, our training method significantly outperformed all other approaches with increases ranging between 7% and 24.5% for the precision metric, between 2.5% and 41.3% for the recall rate, 8% and 26.9% for the F1 score, and 11.6% and 27.4% for the IoU when compared to the results of the typical supervised training approach (Sections 7.3.1).

Similarly, when applied on the change detection task the proposed domain adaptation training framework improves the models' performance on all metrics compared to the supervised approach with the biggest gains reported on the recall rate – which increased by more than 22% – followed by the IoU metric with an increase of approximately 12.6%. Finally, the change predictions' refinement process further improved the results of all the tested models with the greatest benefits being recorded for the precision metric ranging between 5% and 6%.

A quantitative comparison of the results for the two methods suggests that our direct change detection approach leads to better semantic segmentation performance compared to the proposed post-classification pipeline. Our direct CD method when trained using 5% of the labelled training data and the proposed semi-supervised training framework outperforms the post-classification approach on the metrics of precision (by more than 19%), F1 score (by around 9.3%), and IoU (by approximately 10.6%) but can detect fewer changes overall as the recall rate is reduced by about 5.6%. However, a significant advantage of the post-classification method is that it does not require any labelled samples for the change detection task. It can take advantage of datasets developed for pixel-wise land cover classification (such as datasets for building semantic segmentation). In addition, the post-classification approach offers more flexibility when an application requires the detection of class-specific changes as the method can be easily adjusted to identify different types of changes (nevertheless, there is still a need for category/class-specific semantic segmentation training data) and can be used to answer more complex questions with respect to the type of change (class transitions).

## 8.3. Future Work

As deep learning is a relatively new and highly active scientific field there is a plethora of directions for future research.

In this work, we only addressed misregistration errors between the image pair by teaching the network to implicitly model them through additional geometric augmentations, as described in Chapter 4. However, a more sophisticated approach would involve the use of spatial transformers (Jaderberg et al., 2015) that can learn the parameters of a global (rigid) affine transformation or a non-rigid deformation field and warp the network's hidden feature maps according to the learned transformation. Recent research works (Vakalopoulou et al., 2019; Papadomanolaki et al., 2021, 2022) have used deforming autoencoders (Shu et al., 2018) – architectures that combine rigid and deformable transformations – to co-register satellite images in an unsupervised manner. This type of enhancement could be incorporated in both the direct and the post-classification approaches and increase their robustness to misregistration errors.

An interesting extension of the post-classification change detection approach would involve incorporating style transfer techniques such as the generative adversarial approach introduced in

(Hoffman et al., 2017) to explore whether reducing the domain shift between the two epochs could further improve the models' performance. In addition, it would be interesting to compare our proposed Mean Teacher inspired domain adaptation approach to generative adversarial approaches (Hoffman et al., 2017; Vu et al., 2019; Wittich & Rottensteiner, 2021) in the context of building semantic segmentation and change detection.

Further research might also explore ways to combine our post-classification change detection approach with unsupervised change detection methods based on generative adversarial networks and anomaly detection of the image reconstruction loss functions (Noh et al., 2022). Such methods learn to reconstruct the instances of an epoch in the style of another epoch without using any CD labelling, then compare the reconstructed image to the actual one and classify the image regions with high reconstruction error as changed. The building (or potentially any other class) predictions estimated using our domain adaptation method could be used to introduce a semantic loss component ensuring the semantic consistency between the generated and the original images, which has been shown to improve the quality of the generated images significantly (Hoffman et al., 2017).

Furthermore, a promising extension of the proposed post-classification change detection pipeline would involve changing the semantic segmentation task performed on each epoch to instance segmentation using models such as Mask R-CNN (K. He et al., 2017) and its extensions (e.g., PointRend (Kirillov et al., 2020)). Applying instance segmentation would also simplify the filtering step of the pipeline as the building instances would be directly available, thus eliminating the need for extracting instances based on pixel connectivity from the building segmentation predictions. The building predictions could be further improved by introducing refinement steps that also polygonise the building predictions and refine the outputs based on learned building shapes (Girard et al., 2021; K. Zhao et al., 2020). In addition, an instance segmentation pipeline would make it easier to provide the capability of detecting changes between objects of the same class – e.g., a new building replaced an older one between the two epochs – a feature that the current algorithm cannot easily provide.

In addition, further investigation is deemed necessary to fully understand the implications of batch normalization/group normalization on the proposed self-ensembling Mean Teacher framework, which relates to the methods described in Chapters 5 and 7. The research would

involve more experiments on both the semi-supervised and domain adaptation learning contexts, as well as a more in-depth theoretical analysis of the interaction of batch normalization/group normalization layers with the Mean Teacher training method. Another compelling upgrade would involve estimating different batch normalization statistics for each dataset, similar to (Carlucci et al., 2017; Y. Li et al., 2016), within the Mean Teacher training framework.

Finally, it would be interesting to repeat the experiments of both the direct and the post-classification approach with the proposed U-shaped CNN models replaced by architectures that either incorporate multi-headed self-attention layers into CNN architectures (e.g., the Dual Attention Networks (Fu et al., 2019)) or are primarily based on visual transformers and their variations (Z. Liu et al., 2021; Strudel et al., 2021; S. Zheng et al., 2021). These transformer-based architectures have been successfully applied to semantic segmentation tasks and have been shown to match and, in many cases, exceed the performance of state-of-the-art CNN models. However, the transformer-based models are significantly larger than the proposed CNN architectures, and their application will require addressing hardware constraints – with the most critical limitation being the GPU memory resources – and applying specialized pretraining approaches such as masked autoencoders (MAE) (K. He et al., 2022).

# References


Arbeláez, P., Maire, M., Fowlkes, C., & Malik, J. (2011). Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(5), 898–916. https://doi.org/10.1109/TPAMI.2010.161

Bachman, P., Alsharif, O., & Precup, D. (2014). Learning with pseudo-ensembles. *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 3365–3373.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(12), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

Bai, T., Wang, L., Yin, D., Sun, K., Chen, Y., Li, W., & Li, D. (2022). Deep learning for change detection in remote sensing: A review. *Geo-Spatial Information Science*, *0*(0), 1–27. https://doi.org/10.1080/10095020.2022.2085633

Bengio, Y., Lecun, Y., & Hinton, G. (2021). Deep learning for AI. *Communications of the ACM*, *64*(7), 58–65. https://doi.org/10.1145/3448250

Berman, M., Rannen Triki, A., & Blaschko, M. B. (2018). The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4413–4421.

Bousias Alexakis, E., & Armenakis, C. (2020). Evaluation of UNet and UNet++ Architectures in High Resolution Image Change Detection Applications. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLIII-B3-2020*, 1507–1514. https://doi.org/10.5194/isprs-archives-XLIII-B3-2020-1507-2020

Bousias Alexakis, E., & Armenakis, C. (2021a). Evaluation of Semi-Supervised Learning for CNN-Based Change Detection. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLIII-B3-2021*, 829–836. https://doi.org/10.5194/isprs-archives-XLIII-B3-2021-829-2021

Bousias Alexakis, E., & Armenakis, C. (2021b). Performance Improvement of Encoder/Decoder-Based CNN Architectures for Change Detection from Very High-Resolution Satellite Imagery. *Canadian Journal of Remote Sensing*, *47*(2), 309–336. https://doi.org/10.1080/07038992.2021.1922880

Bousias Alexakis, E., & Armenakis, C. (2022). Improving CNN-Based Building Semantic Segmentation Using Object Boundaries. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLIII-B3-2022*, 41–48. https://doi.org/10.5194/isprs-archives-XLIII-B3-2022-41-2022

Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., & Krishnan, D. (2017). Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 95–104. https://doi.org/10.1109/CVPR.2017.18

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: Fast and Flexible Image Augmentations. *Information*, *11*(2), Article 2. https://doi.org/10.3390/info11020125

Cao, C., Dragićević, S., & Li, S. (2019). Land-Use Change Detection with Convolutional Neural Network Methods. *Environments*, *6*(2), 25. https://doi.org/10.3390/environments6020025

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. In A. Vedaldi, H. Bischof, T. Brox, & J.-M. Frahm (Eds.), *Computer Vision – ECCV 2020* (Vol. 12346, pp. 213–229). Springer International Publishing. https://doi.org/10.1007/978-3-030-58452-8_13

Carlucci, F. M., Porzi, L., Caputo, B., Ricci, E., & Bulo, S. R. (2017). AutoDIAL: Automatic Domain Alignment Layers. *2017 IEEE International Conference on Computer Vision (ICCV)*, 5077–5085. https://doi.org/10.1109/ICCV.2017.542

Caye Daudt, R., Le Saux, B., & Boulch, A. (2018). Fully Convolutional Siamese Networks for Change Detection. *2018 25th IEEE International Conference on Image Processing (ICIP)*, 4063–4067. https://doi.org/10.1109/ICIP.2018.8451652

Caye Daudt, R., Le Saux, B., Boulch, A., & Gousseau, Y. (2019). Multitask learning for large-scale semantic change detection. *Computer Vision and Image Understanding*, *187*, 102783. https://doi.org/10.1016/j.cviu.2019.07.003

Chang, W.-G., You, T., Seo, S., Kwak, S., & Han, B. (2019). Domain-Specific Batch Normalization for Unsupervised Domain Adaptation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7346–7354. https://doi.org/10.1109/CVPR.2019.00753

Chang, X., Xiang, T., & Hospedales, T. M. (2018). Scalable and Effective Deep CCA via Soft Decorrelation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1488–1497. https://doi.org/10.1109/CVPR.2018.00161

Chatterjee, B., & Poullis, C. (2019). *Semantic Segmentation from Remote Sensor Data and the Exploitation of Latent Learning for Classification of Auxiliary Tasks*. https://arxiv.org/abs/1912.09216v1

Chen, H., & Shi, Z. (2020). A Spatial-Temporal Attention-Based Method and a New Dataset for Remote Sensing Image Change Detection. *Remote Sensing*, *12*(10), Article 10. https://doi.org/10.3390/rs12101662

Chen, J., Yuan, Z., Peng, J., Chen, L., Huang, H., Zhu, J., Liu, Y., & Li, H. (2021). DASNet: Dual Attentive Fully Convolutional Siamese Networks for Change Detection in High-Resolution Satellite Images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *14*, 1194–1206. https://doi.org/10.1109/JSTARS.2020.3037893

Chen, L.-C., Barron, J. T., Papandreou, G., Murphy, K., & Yuille, A. L. (2016). Semantic Image Segmentation with Task-Specific Edge Detection Using CNNs and a Discriminatively Trained Domain Transform. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4545–4554. https://doi.org/10.1109/CVPR.2016.492

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(4), 834–848. https://doi.org/10.1109/tpami.2017.2699184

Chen, L.-C., Papandreou, G., Schroff, F., & Adam, H. (2017). *Rethinking Atrous Convolution for Semantic Image Segmentation*. https://doi.org/10.48550/arXiv.1706.05587

Cheng, G., Wang, G., & Han, J. (2022). ISNet: Towards Improving Separability for Remote Sensing Image Change Detection. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1–11. https://doi.org/10.1109/TGRS.2022.3174276

Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1800–1807. https://doi.org/10.1109/CVPR.2017.195

Cohen, T., & Welling, M. (2016). Group Equivariant Convolutional Networks. *Proceedings of The 33rd International Conference on Machine Learning*, 2990–2999. https://proceedings.mlr.press/v48/cohenc16.html

Cramer, M. (2010). The DGPF-Test on Digital Airborne Camera Evaluation Overview and Test Design. *Photogrammetrie - Fernerkundung - Geoinformation*, 73–82. https://doi.org/10.1127/1432-8364/2010/0041

Daudt, R. C., Le Saux, B., Boulch, A., & Gousseau, Y. (2018). Urban Change Detection for Multispectral Earth Observation Using Convolutional Neural Networks. *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, 2115–2118. https://doi.org/10.1109/IGARSS.2018.8518015

DeVries, T., & Taylor, G. W. (2017). *Improved Regularization of Convolutional Neural Networks with Cutout* (arXiv:1708.04552). arXiv. https://doi.org/10.48550/arXiv.1708.04552

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale* (arXiv:2010.11929). arXiv. http://arxiv.org/abs/2010.11929

Dwibedi, D., Misra, I., & Hebert, M. (2017). Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, 1310–1319. https://doi.org/10.1109/ICCV.2017.146

Foody, G. M., Pal, M., Rocchini, D., Garzon-Lopez, C. X., & Bastin, L. (2016). The Sensitivity of Mapping Methods to Reference Data Quality: Training Supervised Image Classifications with Imperfect Reference Data. *ISPRS International Journal of Geo-Information*, *5*(11), Article 11. https://doi.org/10.3390/ijgi5110199

French, G., Laine, S., Aila, T., Mackiewicz, M., & Finlayson, G. (2019). *Semi-supervised semantic segmentation needs strong, varied perturbations*. https://doi.org/10.48550/arXiv.1906.01916

Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., & Lu, H. (2019). Dual attention network for scene segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3146–3154.

Girard, N., Smirnov, D., Solomon, J., & Tarabalka, Y. (2021). *Polygonal Building Segmentation by Frame Field Learning* (arXiv:2004.14875; Issue arXiv:2004.14875). arXiv. https://doi.org/10.48550/arXiv.2004.14875

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). *Explaining and Harnessing Adversarial Examples* (arXiv:1412.6572). arXiv. http://arxiv.org/abs/1412.6572

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, *63*(11), 139–144. https://doi.org/10.1145/3422622

He, C., Li, S., Xiong, D., Fang, P., & Liao, M. (2020). Remote Sensing Image Semantic Segmentation Based on Edge Information Guidance. *Remote Sensing*, *12*(9), Article 9. https://doi.org/10.3390/rs12091501

He, H., Li, X., Cheng, G., Shi, J., Tong, Y., Meng, G., Prinet, V., & Weng, L. (2021). *Enhanced Boundary Learning for Glass-Like Object Segmentation*. 15859–15868. https://openaccess.thecvf.com/content/ICCV2021/html/He_Enhanced_Boundary_Learning_for_Glass-Like_Object_Segmentation_ICCV_2021_paper.html

He, J., Zhang, S., Yang, M., Shan, Y., & Huang, T. (2019). Bi-Directional Cascade Network for Perceptual Edge Detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3823–3832. https://doi.org/10.1109/CVPR.2019.00395

He, J., Zhang, S., Yang, M., Shan, Y., & Huang, T. (2022). BDCN: Bi-Directional Cascade Network for Perceptual Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(1), 100–113. https://doi.org/10.1109/TPAMI.2020.3007074

He, K., Chen, X., Xie, S., Li, Y., Dollar, P., & Girshick, R. (2022). Masked Autoencoders Are Scalable Vision Learners. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 15979–15988. https://doi.org/10.1109/CVPR52688.2022.01553

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2980–2988. https://doi.org/10.1109/ICCV.2017.322

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Hobley, B. L., Arosio, R., French, G., Bremner, J., Dolphin, T., & Mackiewicz, M. (2021). *Semi-supervised segmentation for coastal monitoring seagrass using RPA imagery*. https://doi.org/10.20944/preprints202103.0780.v1

Hoffman, J., Tzeng, E., Park, T., Zhu, J.-Y., Isola, P., Saenko, K., Efros, A. A., & Darrell, T. (2017). *CyCADA: Cycle-Consistent Adversarial Domain Adaptation*. https://doi.org/10.48550/arXiv.1711.03213

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, 448–456.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-Image Translation with Conditional Adversarial Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5967–5976. https://doi.org/10.1109/CVPR.2017.632

Jaderberg, M., Simonyan, K., Zisserman, A., & Kavukcuoglu, K. (2015). Spatial transformer networks. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, 2017–2025.

Ji, S., Wei, S., & Lu, M. (2019). Fully Convolutional Networks for Multisource Building Extraction From an Open Aerial and Satellite Imagery Data Set. *IEEE Transactions on Geoscience and Remote Sensing*, *57*(1), 574–586. https://doi.org/10.1109/TGRS.2018.2858817

Jung, H., Choi, H.-S., & Kang, M. (2022). Boundary Enhancement Semantic Segmentation for Building Extraction From Remote Sensed Image. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1–12. https://doi.org/10.1109/TGRS.2021.3108781

Ke, Z., Wang, D., Yan, Q., Ren, J., & Lau, R. (2019). Dual Student: Breaking the Limits of the Teacher in Semi-Supervised Learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 6727–6735. https://doi.org/10.1109/ICCV.2019.00683

Khelifi, L., & Mignotte, M. (2020). Deep Learning for Change Detection in Remote Sensing Images: Comprehensive Review and Meta-Analysis. *IEEE Access*, *8*, 126385–126400. https://doi.org/10.1109/ACCESS.2020.3008036

Kirillov, A., Wu, Y., He, K., & Girshick, R. (2020). PointRend: Image Segmentation As Rendering. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9796–9805. https://doi.org/10.1109/CVPR42600.2020.00982

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In F. Pereira, C. J. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 25). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

Laine, S., & Aila, T. (2017). *Temporal Ensembling for Semi-Supervised Learning* (arXiv:1610.02242). arXiv. https://doi.org/10.48550/arXiv.1610.02242

Lebedev, M. A., Vizilter, Y. V., Vygolov, O. V., Knyaz, V. A., & Rubis, A. Y. (2018). Change Detection in Remote Sensing Images Using Conditional Adversarial Networks. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII–2*, 565–571. https://doi.org/10.5194/isprs-archives-XLII-2-565-2018

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), Article 7553. https://doi.org/10.1038/nature14539

Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., & Tu, Z. (2015). Deeply-Supervised Nets. *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, 562–570. https://proceedings.mlr.press/v38/lee15a.html

Li, S., Zhao, Z., Xu, K., Zeng, Z., & Guan, C. (2021). Hierarchical Consistency Regularized Mean Teacher for Semi-supervised 3D Left Atrium Segmentation. *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 3395–3398. https://doi.org/10.1109/EMBC46164.2021.9629941

Li, X., Li, X., Zhang, L., Cheng, G., Shi, J., Lin, Z., Tan, S., & Tong, Y. (2020). Improving Semantic Segmentation via Decoupled Body and Edge Supervision. In A. Vedaldi, H. Bischof, T. Brox, & J.-M. Frahm (Eds.), *Computer Vision – ECCV 2020* (pp. 435–452). Springer International Publishing. https://doi.org/10.1007/978-3-030-58520-4_26

Li, X., Yu, L., Chen, H., Fu, C.-W., & Heng, P.-A. (2018). *Semi-supervised Skin Lesion Segmentation via Transformation Consistent Self-ensembling Model* (arXiv:1808.03887). arXiv. http://arxiv.org/abs/1808.03887

Li, X., Yu, L., Chen, H., Fu, C.-W., Xing, L., & Heng, P.-A. (2021). Transformation-Consistent Self-Ensembling Model for Semisupervised Medical Image Segmentation. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(2), 523–534. https://doi.org/10.1109/TNNLS.2020.2995319

Li, Y., Wang, N., Shi, J., Liu, J., & Hou, X. (2016). *Revisiting Batch Normalization For Practical Domain Adaptation* (arXiv:1603.04779). arXiv. http://arxiv.org/abs/1603.04779

Lim, H., Kim, B., Choo, J., & Choi, S. (2023). *TTN: A Domain-Shift Aware Batch Normalization in Test-Time Adaptation* (arXiv:2302.05155). arXiv. http://arxiv.org/abs/2302.05155

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature Pyramid Networks for Object Detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 936–944. https://doi.org/10.1109/CVPR.2017.106

Liu, M.-Y., Breuel, T., & Kautz, J. (2017). Unsupervised image-to-image translation networks. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 700–708.

Liu, M.-Y., & Tuzel, O. (2016). Coupled Generative Adversarial Networks. *Advances in Neural Information Processing Systems*, *29*. https://papers.nips.cc/paper/2016/hash/502e4a16930e414107ee22b6198c578f-Abstract.html

Liu, Y., Tian, Y., Chen, Y., Liu, F., Belagiannis, V., & Carneiro, G. (2022). Perturbed and Strict Mean Teachers for Semi-supervised Semantic Segmentation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4248–4257. https://doi.org/10.1109/CVPR52688.2022.00422

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 9992–10002. https://doi.org/10.1109/ICCV48922.2021.00986

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3431–3440. https://doi.org/10.1109/CVPR.2015.7298965

Lu, D., Mausel, P., Brondízio, E., & Moran, E. (2004). Change detection techniques. *International Journal of Remote Sensing*, *25*(12), 2365–2401. https://doi.org/10.1080/0143116031000139863

Lyu, H., Fu, H., Hu, X., & Liu, L. (2019). Esnet: Edge-Based Segmentation Network for Real-Time Semantic Segmentation in Traffic Scenes. *2019 IEEE International Conference on Image Processing (ICIP)*, 1855–1859. https://doi.org/10.1109/ICIP.2019.8803132

Maggiori, E., Tarabalka, Y., Charpiat, G., & Alliez, P. (2017). Can semantic labeling methods generalize to any city? The inria aerial image labeling benchmark. *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 3226–3229. https://doi.org/10.1109/IGARSS.2017.8127684

Marmanis, D., Schindler, K., Wegner, J. D., Galliani, S., Datcu, M., & Stilla, U. (2018). Classification with an edge: Improving semantic image segmentation with boundary detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, *135*, 158–172. https://doi.org/10.1016/j.isprsjprs.2017.11.009

Mehta, S., Rastegari, M., Shapiro, L., & Hajishirzi, H. (2019). ESPNetv2: A Light-Weight, Power Efficient, and General Purpose Convolutional Neural Network. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9182–9192. https://doi.org/10.1109/CVPR.2019.00941

Melekhov, I., Tiulpin, A., Sattler, T., Pollefeys, M., Rahtu, E., & Kannala, J. (2019). DGC-Net: Dense Geometric Correspondence Network. *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1034–1042. https://doi.org/10.1109/WACV.2019.00115

Mi, P., Lin, J., Zhou, Y., Shen, Y., Luo, G., Sun, X., Cao, L., Fu, R., Xu, Q., & Ji, R. (2022). Active Teacher for Semi-Supervised Object Detection. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 14462–14471. https://doi.org/10.1109/CVPR52688.2022.01408

Miyato, T., Maeda, S.-I., Koyama, M., & Ishii, S. (2019). Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*(8), 1979–1993. https://doi.org/10.1109/TPAMI.2018.2858821

Murez, Z., Kolouri, S., Kriegman, D., Ramamoorthi, R., & Kim, K. (2018). Image to Image Translation for Domain Adaptation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4500–4509. https://doi.org/10.1109/CVPR.2018.00473

Noa, J., Soto, P. J., Costa, G. a. O. P., Wittich, D., Feitosa, R. Q., & Rottensteiner, F. (2021). Adversarial Discriminative Domain Adaptation for Deforestation Detection. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *V-3–2021*, 151–158. https://doi.org/10.5194/isprs-annals-V-3-2021-151-2021

Noh, H., Ju, J., Seo, M., Park, J., & Choi, D.-G. (2022). *Unsupervised Change Detection Based on Image Reconstruction Loss*. 1352–1361. https://openaccess.thecvf.com/content/CVPR2022W/EarthVision/html/Noh_Unsupervised_Change_Detection_Based_on_Image_Reconstruction_Loss_CVPRW_2022_paper.html

Ouali, Y., Hudelot, C., & Tami, M. (2020a). *An Overview of Deep Semi-Supervised Learning* (arXiv:2006.05278). arXiv. http://arxiv.org/abs/2006.05278

Ouali, Y., Hudelot, C., & Tami, M. (2020b). Semi-Supervised Semantic Segmentation With Cross-Consistency Training. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12671–12681. https://doi.org/10.1109/CVPR42600.2020.01269

Papadomanolaki, M., Christodoulidis, S., Karantzalos, K., & Vakalopoulou, M. (2021). Unsupervised Multistep Deformable Registration of Remote Sensing Imagery Based on Deep Learning. *Remote Sensing*, *13*(7), Article 7. https://doi.org/10.3390/rs13071294

Papadomanolaki, M., Vakalopoulau, M., & Karantzalos, K. (2022). Deep Learning based Multistep Registration Focusing on Regions of Change. *IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium*, 1948–1951. https://doi.org/10.1109/IGARSS46834.2022.9884237

Papadomanolaki, M., Vakalopoulou, M., & Karantzalos, K. (2020). Urban Change Detection Based on Semantic Segmentation and Fully Convolutional LSTM Networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *V-2–2020*, 541–547. https://doi.org/10.5194/isprs-annals-V-2-2020-541-2020

Park, J.-M., Kim, U.-H., Lee, S.-H., & Kim, J.-H. (2022). Dual Task Learning by Leveraging Both Dense Correspondence and Mis-Correspondence for Robust Change Detection With

Imperfect Matches. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13739–13749. https://doi.org/10.1109/CVPR52688.2022.01338

Park, T., Efros, A. A., Zhang, R., & Zhu, J.-Y. (2020). Contrastive Learning for Unpaired Image-to-Image Translation. In A. Vedaldi, H. Bischof, T. Brox, & J.-M. Frahm (Eds.), *Computer Vision – ECCV 2020* (pp. 319–345). Springer International Publishing. https://doi.org/10.1007/978-3-030-58545-7_19

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Peláez-Vegas, A., Mesejo, P., & Luengo, J. (2023). *A Survey on Semi-Supervised Semantic Segmentation* (arXiv:2302.09899). arXiv. https://doi.org/10.48550/arXiv.2302.09899

Peng, D., Zhang, Y., & Guan, H. (2019). End-to-End Change Detection for High Resolution Satellite Images Using Improved UNet++. *Remote Sensing*, *11*(11), Article 11. https://doi.org/10.3390/rs11111382

Peng, X., Zhong, R., Li, Z., & Li, Q. (2021). Optical Remote Sensing Image Change Detection Based on Attention Mechanism and Image Difference. *IEEE Transactions on Geoscience and Remote Sensing*, *59*(9), 7296–7307. https://doi.org/10.1109/TGRS.2020.3033009

Rakhlin, A., Davydow, A., & Nikolenko, S. (2018). *Land Cover Classification From Satellite Imagery With U-Net and Lovasz-Softmax Loss*. 262–266. http://openaccess.thecvf.com/content_cvpr_2018_workshops/w4/html/Rakhlin_Land_Cover_Classification_CVPR_2018_paper.html

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (pp. 234–241). Springer International Publishing. https://doi.org/10.1007/978-3-319-24574-4_28

Ru, L., Du, B., & Wu, C. (2021). Multi-Temporal Scene Classification and Scene Change Detection With Correlation Based Fusion. *IEEE Transactions on Image Processing*, *30*, 1382–1394. https://doi.org/10.1109/TIP.2020.3039328

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., & Bethge, M. (2020). Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, *33*, 11539–11551. https://proceedings.neurips.cc/paper/2020/hash/85690f81aadc1749175c187784afc9ee-Abstract.html

Seydi, S. T., Hasanlou, M., & Amani, M. (2020). A New End-to-End Multi-Dimensional CNN Framework for Land Cover/Land Use Change Detection in Multi-Source Remote Sensing Datasets. *Remote Sensing*, *12*(12), Article 12. https://doi.org/10.3390/rs12122010

Shafique, A., Cao, G., Khan, Z., Asad, M., & Aslam, M. (2022). Deep Learning-Based Change Detection in Remote Sensing Images: A Review. *Remote Sensing*, *14*(4), Article 4. https://doi.org/10.3390/rs14040871

Shao, Z., Tang, P., Wang, Z., Saleem, N., Yam, S., & Sommai, C. (2020). BRRNet: A Fully Convolutional Neural Network for Automatic Building Extraction From High-Resolution Remote Sensing Images. *Remote Sensing*, *12*(6), Article 6. https://doi.org/10.3390/rs12061050

Shu, Z., Sahasrabudhe, M., Alp Güler, R., Samaras, D., Paragios, N., & Kokkinos, I. (2018). Deforming Autoencoders: Unsupervised Disentangling of Shape and Appearance. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018* (Vol. 11214, pp. 664–680). Springer International Publishing. https://doi.org/10.1007/978-3-030-01249-6_40

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations (ICLR 2015)*. https://ora.ox.ac.uk/objects/uuid:60713f18-a6d1-4d97-8f45-b60ad8aebbce

Singh, A. (1989). Review Article Digital change detection techniques using remotely-sensed data. *International Journal of Remote Sensing*, *10*(6), Article 6. https://doi.org/10.1080/01431168908903939

Soria, X., Riba, E., & Sappa, A. (2020). Dense Extreme Inception Network: Towards a Robust CNN Model for Edge Detection. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1912–1921. https://doi.org/10.1109/WACV45572.2020.9093290

Soto, P. J., Costa, G. a. O. P., Feitosa, R. Q., Happ, P. N., Ortega, M. X., Noa, J., Almeida, C. A., & Heipke, C. (2020). Domain Adaptation with CycleGAN gor Change Detection in the Amazon Forest. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLIII-B3-2020*, 1635–1643. https://doi.org/10.5194/isprs-archives-XLIII-B3-2020-1635-2020

Soto Vega, P. J., Costa, G. A. O. P. da, Feitosa, R. Q., Ortega Adarme, M. X., Almeida, C. A. de, Heipke, C., & Rottensteiner, F. (2021). An unsupervised domain adaptation approach for change detection and its application to deforestation mapping in tropical biomes. *ISPRS Journal of Photogrammetry and Remote Sensing*, *181*, 113–128. https://doi.org/10.1016/j.isprsjprs.2021.08.026

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*(56), 1929–1958.

Strudel, R., Garcia, R., Laptev, I., & Schmid, C. (2021). Segmenter: Transformer for Semantic Segmentation. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 7242–7252. https://doi.org/10.1109/ICCV48922.2021.00717

Su, L., Gong, M., Zhang, P., Zhang, M., Liu, J., & Yang, H. (2017). Deep learning and mapping based ternary change detection for information unbalanced images. *Pattern Recognition*, *66*, 213–228. https://doi.org/10.1016/j.patcog.2017.01.002

Su, S., Nawata, T., & Fuse, T. (2020). Building Change Detection from Bitemporal Aerial Images Using Deep Learning. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *V-2–2020*, 565–571. https://doi.org/10.5194/isprs-annals-V-2-2020-565-2020

Sun, B., Feng, J., & Saenko, K. (2016). Return of Frustratingly Easy Domain Adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, *30*(1), Article 1. https://doi.org/10.1609/aaai.v30i1.10306

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., & Hardt, M. (2020). Test-Time Training with Self-Supervision for Generalization under Distribution Shifts. *Proceedings of the 37th International Conference on Machine Learning*, 9229–9248. https://proceedings.mlr.press/v119/sun20b.html

Tarvainen, A., & Valpola, H. (2017, February 17). *Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results*. NIPS. https://www.semanticscholar.org/paper/Mean-teachers-are-better-role-models%3A-consistency-Tarvainen-Valpola/1342c1e1684620c019972e2679d5131f1e8a4a13?sort=is-influential&page=7

Tewkesbury, A. P., Comber, A. J., Tate, N. J., Lamb, A., & Fisher, P. F. (2015). A critical synthesis of remotely sensed optical image change detection techniques. *Remote Sensing of Environment*, *160*, 1–14. https://doi.org/10.1016/j.rse.2015.01.006

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., & Bregler, C. (2015). Efficient object localization using Convolutional Networks. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 648–656. https://doi.org/10.1109/CVPR.2015.7298664

Tong, X.-Y., Xia, G.-S., Lu, Q., Shen, H., Li, S., You, S., & Zhang, L. (2020). Land-cover classification with high-resolution remote sensing images using transferable deep models. *Remote Sensing of Environment*, *237*, 111322. https://doi.org/10.1016/j.rse.2019.111322

Truong, P., Danelljan, M., & Timofte, R. (2020). GLU-Net: Global-Local Universal Network for Dense Flow and Correspondences. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6257–6267. https://doi.org/10.1109/CVPR42600.2020.00629

Tzeng, E., Hoffman, J., Saenko, K., & Darrell, T. (2017). Adversarial Discriminative Domain Adaptation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2962–2971. https://doi.org/10.1109/CVPR.2017.316

Vakalopoulou, M., Christodoulidis, S., Sahasrabudhe, M., Mougiakakou, S., & Paragios, N. (2019). Image Registration of Satellite Imagery with Deep Convolutional Neural

Networks. *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, 4939–4942. https://doi.org/10.1109/IGARSS.2019.8898220

Van Etten, A., Lindenbaum, D., & Bacastow, T. M. (2019). *SpaceNet: A Remote Sensing Dataset and Challenge Series* (arXiv:1807.01232). arXiv. https://doi.org/10.48550/arXiv.1807.01232

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems*, *30*. https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Verma, V., Kawaguchi, K., Lamb, A., Kannala, J., Solin, A., Bengio, Y., & Lopez-Paz, D. (2022). Interpolation consistency training for semi-supervised learning. *Neural Networks*, *145*, 90–106. https://doi.org/10.1016/j.neunet.2021.10.008

Vu, T.-H., Jain, H., Bucher, M., Cord, M., & Perez, P. (2019). ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2512–2521. https://doi.org/10.1109/CVPR.2019.00262

Wiratama, W., Lee, J., Park, S.-E., & Sim, D. (2018). Dual-Dense Convolution Network for Change Detection of High-Resolution Panchromatic Imagery. *Applied Sciences*, *8*(10), 1785. https://doi.org/10.3390/app8101785

Wittich, D. (2020). Deep Domain Adaptation by Weighted Entropy Minimization for the Classification of Aerial Images. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *V-2–2020*, 591–598. https://doi.org/10.5194/isprs-annals-V-2-2020-591-2020

Wittich, D., & Rottensteiner, F. (2019). Adversarial Domain Adaptation for the Classification of Aerial Images and Height Data Using Convolutional Neural Networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *IV-2-W7*, 197–204. https://doi.org/10.5194/isprs-annals-IV-2-W7-197-2019

Wittich, D., & Rottensteiner, F. (2021). Appearance based deep domain adaptation for the classification of aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, *180*, 82–102. https://doi.org/10.1016/j.isprsjprs.2021.08.004

Xie, S., & Tu, Z. (2015). Holistically-Nested Edge Detection. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1395–1403. https://doi.org/10.1109/ICCV.2015.164

Xu, Y., Wu, L., Xie, Z., & Chen, Z. (2018). Building Extraction in Very High Resolution Remote Sensing Imagery Using Deep Learning and Guided Filters. *Remote Sensing*, *10*(1), Article 1. https://doi.org/10.3390/rs10010144

Yan, T., Wan, Z., & Zhang, P. (2023). Fully Transformer Network for Change Detection of Remote Sensing Images. In L. Wang, J. Gall, T.-J. Chin, I. Sato, & R. Chellappa (Eds.), *Computer Vision – ACCV 2022* (Vol. 13842, pp. 75–92). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-26284-5_5

Yang, L., Zhuo, W., Qi, L., Shi, Y., & Gao, Y. (2022). ST++: Make Self-trainingWork Better for Semi-supervised Semantic Segmentation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4258–4267. https://doi.org/10.1109/CVPR52688.2022.00423

Yuan, J. (2018). Learning Building Extraction in Aerial Scenes with Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(11), 2793–2798. https://doi.org/10.1109/TPAMI.2017.2750680

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., & Choe, J. (2019). CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 6022–6031. https://doi.org/10.1109/ICCV.2019.00612

Zagoruyko, S., & Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4353–4361. https://doi.org/10.1109/CVPR.2015.7299064

Zhang, C., Wang, L., Cheng, S., & Li, Y. (2022). SwinSUNet: Pure Transformer Network for Remote Sensing Image Change Detection. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1–13. https://doi.org/10.1109/TGRS.2022.3160007

Zhang, C., Wei, S., Ji, S., & Lu, M. (2019). Detecting Large-Scale Urban Land Cover Changes from Very High Resolution Remote Sensing Images Using CNN-Based Classification. *ISPRS International Journal of Geo-Information*, *8*(4), Article 4. https://doi.org/10.3390/ijgi8040189

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). *mixup: Beyond Empirical Risk Minimization* (arXiv:1710.09412). arXiv. http://arxiv.org/abs/1710.09412

Zhang, W., & Lu, X. (2019). The Spectral-Spatial Joint Learning for Change Detection in Multispectral Imagery. *Remote Sensing*, *11*(3), Article 3. https://doi.org/10.3390/rs11030240

Zhang, X., Shi, W., Lv, Z., & Peng, F. (2019). Land Cover Change Detection from High-Resolution Remote Sensing Imagery Using Multitemporal Deep Feature Collaborative Learning and a Semi-supervised Chan–Vese Model. *Remote Sensing*, *11*(23), Article 23. https://doi.org/10.3390/rs11232787

Zhang, Z., Liu, Q., & Wang, Y. (2018). Road Extraction by Deep Residual U-Net. *IEEE Geoscience and Remote Sensing Letters*, *15*(5), Article 5. https://doi.org/10.1109/LGRS.2018.2802944

Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid Scene Parsing Network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6230–6239. https://doi.org/10.1109/CVPR.2017.660

Zhao, K., Kamran, M., & Sohn, G. (2020). Boundary Regularized Building Footprint Extraction from Satellite Images Using Deep Neural Networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *V-2–2020*, 617–624. https://doi.org/10.5194/isprs-annals-V-2-2020-617-2020

Zhao, K., Kang, J., Jung, J., & Sohn, G. (2018). Building Extraction from Satellite Images Using Mask R-CNN with Building Boundary Regularization. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 242–2424. https://doi.org/10.1109/CVPRW.2018.00045

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H. S., & Zhang, L. (2021). Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6877–6886. https://doi.org/10.1109/CVPR46437.2021.00681

Zheng, Z., Ma, A., Zhang, L., & Zhong, Y. (2021). Change is Everywhere: Single-Temporal Supervised Object Change Detection in Remote Sensing Imagery. *2021 IEEE/CVF*

*International Conference on Computer Vision (ICCV)*, 15173–15182. https://doi.org/10.1109/ICCV48922.2021.01491

Zhou, H., Jiang, F., & Lu, H. (2023). SSDA-YOLO: Semi-supervised domain adaptive YOLO for cross-domain object detection. *Computer Vision and Image Understanding*, *229*, 103649. https://doi.org/10.1016/j.cviu.2023.103649

Zhou, Z., Rahman Siddiquee, M. M., Tajbakhsh, N., & Liang, J. (2018). UNet++: A Nested U-Net Architecture for Medical Image Segmentation. In D. Stoyanov, Z. Taylor, G. Carneiro, T. Syeda-Mahmood, A. Martel, L. Maier-Hein, J. M. R. S. Tavares, A. Bradley, J. P. Papa, V. Belagiannis, J. C. Nascimento, Z. Lu, S. Conjeti, M. Moradi, H. Greenspan, & A. Madabhushi (Eds.), *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (pp. 3–11). Springer International Publishing. https://doi.org/10.1007/978-3-030-00889-5_1

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2242–2251. https://doi.org/10.1109/ICCV.2017.244

Zhu, Y., Sapra, K., Reda, F. A., Shih, K. J., Newsam, S., Tao, A., & Catanzaro, B. (2019). Improving Semantic Segmentation via Video Propagation and Label Relaxation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8848–8857. https://doi.org/10.1109/CVPR.2019.00906

# Appendix A.

# Batch Normalization in Domain Adaptation

While studying the domain adaptation for building semantic segmentation, we noticed that preliminary experiments' training and validation IoU curves indicated that the initial IoU improvements that the Mean Teacher method achieved on the validation (target) set gradually diminished as the training progressed. The performance of the models in the training set did not show a similar drop but was stable/converging to an upper bound IoU value. Examples of the described model behaviour can be found in Section 7.3.2.2 (Figures 7.9 and 7.10).

The training curves on the Source and Target domains resembled typical overfitting on the training data. Thus, the first step was the introduction of extra perturbations on the feature space. Although introducing spatial dropout layers mitigated the issue to a certain extent, it did not eliminate the apparent deterioration of the IoU values during training. Hence, in an attempt to explain the algorithm's behaviour, we turned our attention to the role and setup of the Batch Normalization layers of the model.

A review of relevant research works suggested that by using the feature mean and variance statistics of the training population, batch normalization layers are adapted to the characteristics of the source domain and may hamper the model performance on the target domain when the domain shift is substantial (W.-G. Chang et al., 2019; Schneider et al., 2020).

## A.1. Introduction to Batch Normalization

Batch normalization was introduced by Ioffe & Szegedy (2015) as a way to address the internal covariate shift phenomenon, which refers to complications caused by the changes in the distributions of the inputs of each of the networks' layers due to the change in network parameters during training. Batch normalization allows the use of much higher learning rates, reduces the need for careful initializations and has a regularization effect on training that helps reduce overfitting.

Provided that the training is performed in batches using stochastic gradient decent and a batch sample size of $s$, a batch normalization layer placed after an activation layer $X = \{x^{(1)} \dots x^{(c)}\}$

with $c$ number of chanels, processes separately each activation channel $x^{(j)}$, to which we wil simply refer as $x$ in Equations A.1 to A.3 for clarity.

During training the algorithm first computes the mean and standard deviation of the minibatch for channel $x$ (Eq. A.1 & A.2) and then uses the estimated values to normalize the activations (Eq. A.3). As a final step the normalized activations $\hat{x}$ are scaled and shifted by a pair of learnable parameters per activation channel $\gamma^{(j)}$ and $\beta^{(j)}$(Eq. A.4).

$$\mu_B = \frac{1}{s}\sum_{i=1}^{s} x_i \tag{A.1}$$

$$\sigma_B^2 = \frac{1}{s}\sum_{i=1}^{s} (x_i - \mu_B)^2 \tag{A.2}$$

$$\widehat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{A.3}$$

$$y^{(j)} = \gamma^{(j)}\hat{x}^{(j)} + \beta^{(j)} \tag{A.4}$$

At inference time, the batch normalization layer uses the population mean and variance instead of the minibatch statistics (Eq. A.5 instead of Eq. A.3) mainly to ensure that the output/results depend on the input deterministically and will not change if the samples are reshuffled into new minibatches.

$$\widehat{x_i} = \frac{x_i - E(x_i)}{\sqrt{Var(x_i) + \epsilon}} \tag{A.5}$$

## A.2. Batch Normalization in Domain Adaptation

However, when used in domain adaptation applications, sharing the population mean and variance of the source domain for both source and target domains will hurt performance if the

domain shift is significant (W.-G. Chang et al., 2019; Schneider et al., 2020) as the statistics of the BN layers contain the traits of the source/training domain (Li et al., 2016).

During the past few years, multiple research works on domain adaptation have proposed improvements to the batch normalization layer as a quick and relatively simple way to improve the model's performance on the target domain (W.-G. Chang et al., 2019; Y. Li et al., 2016; Lim et al., 2023; Schneider et al., 2020).

Li et al. (2016) propose using different population estimates for the mean and variance of each batch normalization layer, which are computed separately for the source and the target domain. Even though this simple adjustment of the Batch Normalization layers, called Adaptive Batch Normalization (AdaBN), requires no extra learning parameters, it leads to significant accuracy improvements when applied to two benchmark DA datasets for image classification compared to the baseline. A similar approach that uses different population statistics for the target and source domains in adjusted batch normalization layers, named Domain Alignment Layers, was described by Carlucci et al. (2017).

Chang et al. (2019) propose using separate batch normalization layers for the source and target domains. The authors expect that the separately estimated batch statistics and learned affine parameters of the Domain Specific Batch Normalization (DSBN) layers will be able to capture domain-specific information and thus help the network learn domain-invariant features. The proposed approach is tested by incorporating the Domain Specific Batch Normalization layers to two unsupervised domain adaptation algorithms, the Moving Semantic Transfer Network (MSTN) and the Class Prediction Uncertainty Alignment (CPUA), and testing the approach on three standard benchmark domain adaptation datasets for image classification. Replacing the BN layers with DSBN layers led to improved classification accuracies in all datasets.

Improvements to batch normalization have also been proposed for test-time adaptation (Y. Sun et al., 2020) applications, which try to reduce the domain shift between source and target domain during test time. Lim et al. (2023) developed a test-time normalization (TTN) strategy that combines the source and target batch statistics of a batch normalization layer using per-channel interpolating weights that depend on the sensitivity of each channel to the domain shift. This

way, the enhanced BN layers can flexibly adapt to new target domains while preserving the knowledge from the source domain.

According to Y. Sun et al. (2020, Section A4.1) Group Normalization significantly improves model robustness compared to Batch Normalization when using small mini-batches in the proposed Test-Time Training approach. Test-Time Training differs from Domain Adaptation in that it learns from distribution shifts at test time, instead of trying to learn and “anticipate” the changes in distribution between the source and target domains.

## A.3. Using Different BN Feature Mean and Variance Values for the Student and Teacher Models

In our experiments, we addressed the batch normalization sensitivity to the domain shift by permanently setting the batch normalization layers of the student network to evaluation (validation) mode after a few iterations on both datasets. As mentioned earlier in the section, during validation, the BN layer uses the statistics of the population (which are estimated via a moving average (Ioffe & Szegedy, 2015) during training) and does not compute the mean and variance for each batch. Thus, the BN layers degenerate into a learnable linear transformation for each channel of the activation layer used as input to the BN. Meanwhile, the BN layers of the teacher model are still using the statistics of the population. In this way, we add an extra source of perturbations in the feature space, which should help the models learn feature representations that are more expressive and more robust to feature noise.

Unless stated otherwise, all the results relating to the Mean Teacher training framework have been trained using this BN setup.

# Appendix B.

# Effects on the Change Detection Process using boundaries trained on the Vaihingen Dataset

In this appendix the CD performance of edge enhanced networks when using retrained DexiNeD weights on Vaihingen dataset is compared to the corresponding performance of a network enhanced with predictions from DexiNeD trained on the general-purpose edge detection BIPED dataset.

Table B.1 presents the results retrieved from the edge enhanced models when using edge predictions using (a) DexiNeD with its original weights as presented in (Soria et al, 2020) (rows 2 and 5) and (b) DexiNeD trained on the boundary detection dataset that we created based on the Vaihingen dataset (rows 1 and 4), as well as the results retrieved when using only the backbone architecture without any edge information (rows 3 and 6). The results are computed for both UNet and UNet++ architectures trained using the BCE Dice loss. Based on Table A1, UNet++ using boundaries predicted from the retrained version of DexiNeD performs better than UNet++ using boundaries predicted using the paper version of DexiNeD (1.5% or higher improvement on recall, F1 score and IoU metrics and 0.4% improvement on precision) while UNet++ with added, but not semantically enhanced, edges performs marginally better than the plain UNet++ architecture (with the improvement ranging from 0.35% on the recall and F1 score metrics to 0.6% on the precision and IoU metrics). Similar behaviour is observed with respect to the UNet architecture with the improvement between using non-semantically informed edges and using the plain architecture ranging between 0.9% and 1.4%) and a further improvement of 0.6% to 1.2% being noted when instead of the untrained edges we use the model with the retrained DexiNeD weights on the modified Vaihingen dataset.

*Table B.1: Comparison of boundary enhanced networks using DexiNeD trained on the BIPED dataset and DexiNeD trained on the boundary detection dataset created from the Vaihingen dataset. The results retrieved by using solely the backbone architecture are also presented in rows 3 and 6.*

| Backbone Architecture | Boundaries | Trained on Vaihingen | Accuracy | Precision | Recall | F1 | IoU |
|---|---|---|---|---|---|---|---|
| UNet++ | ✓ | ✓ | **0.9893** | **0.9509** | **0.8332** | **0.8737** | **0.8035** |
| UNet++ | ✓ | ✗ | 0.9881 | 0.9470 | 0.8165 | 0.8596 | 0.7849 |
| UNet++ | ✗ | ✗ | 0.9872 | 0.9406 | 0.8129 | 0.8561 | 0.7788 |
| UNet | ✓ | ✓ | **0.9877** | **0.9423** | **0.8142** | **0.8566** | **0.7803** |
| UNet | ✓ | ✗ | 0.9864 | 0.9363 | 0.8057 | 0.8501 | 0.7687 |
| UNet | ✗ | ✗ | 0.9859 | 0.9273 | 0.7956 | 0.8377 | 0.7549 |

# Appendix C.

# Comparison of BCE and MSE loss functions for the Consistency Regularization Constraint

For the Consistency Regularization Constraint (CRL), we tested both the BCE and MSE loss functions. We conducted the experiment twice: first, on the Vegas target domain (Figures C.1 and C.2), and then, on the Shanghai dataset (Figures C.3 and C.4).

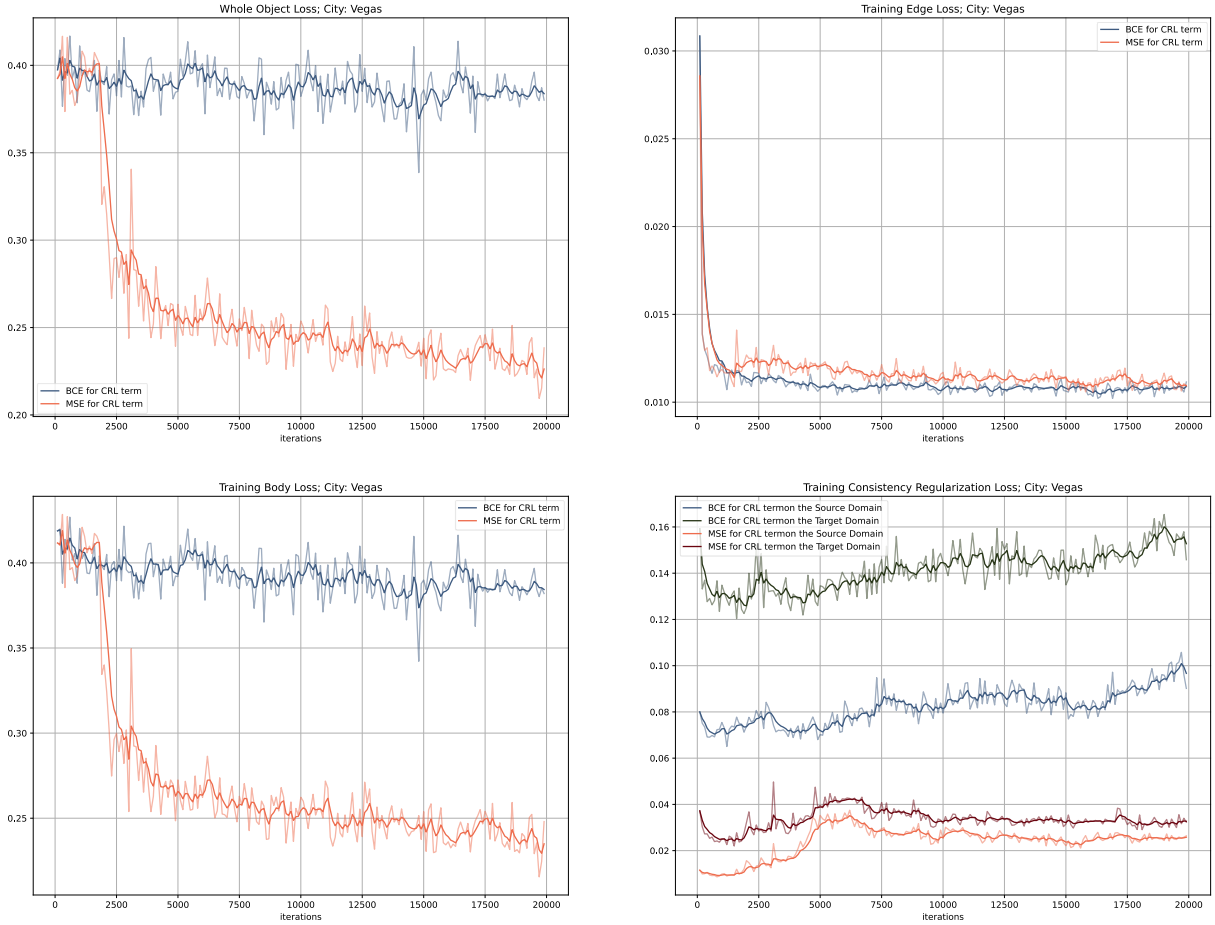



*Figure C.1:BCE versus MSE training curves for the Vegas dataset.*

Figure C.1 presents the training curves for the various components of the loss function for the model trained using either BCE (blue) or MSE (orange) as the consistency regularization loss (CRL). In the top left corner, we present the loss values corresponding to the entire/fused output

of the model (whole object loss). The top right corner shows the edge component of the supervised loss function, and the bottom left corner shows the loss component corresponding to the object body. The bottom right corner shows the values of the regularization terms relating to the Source Domain and the Target Domain, respectively.

The validation curves both for the loss function and the IoU values are illustrated in Figure C.2. Similarly to Figure C.1, the top right corner provides the Fused supervised Loss and the top left and bottom right corners show the losses that correspond to the buildings' edge and body components, respectively. The bottom right corner presents the IoU values on the target domain for the two models trained using each of the two functions as the CRL term. In both figures, the prominent lines are the smoothed training/validation values produced via an Exponential Moving Average with a smoothing factor of 0.3. In contrast, the faint lines represent each model's estimated training/validation values.

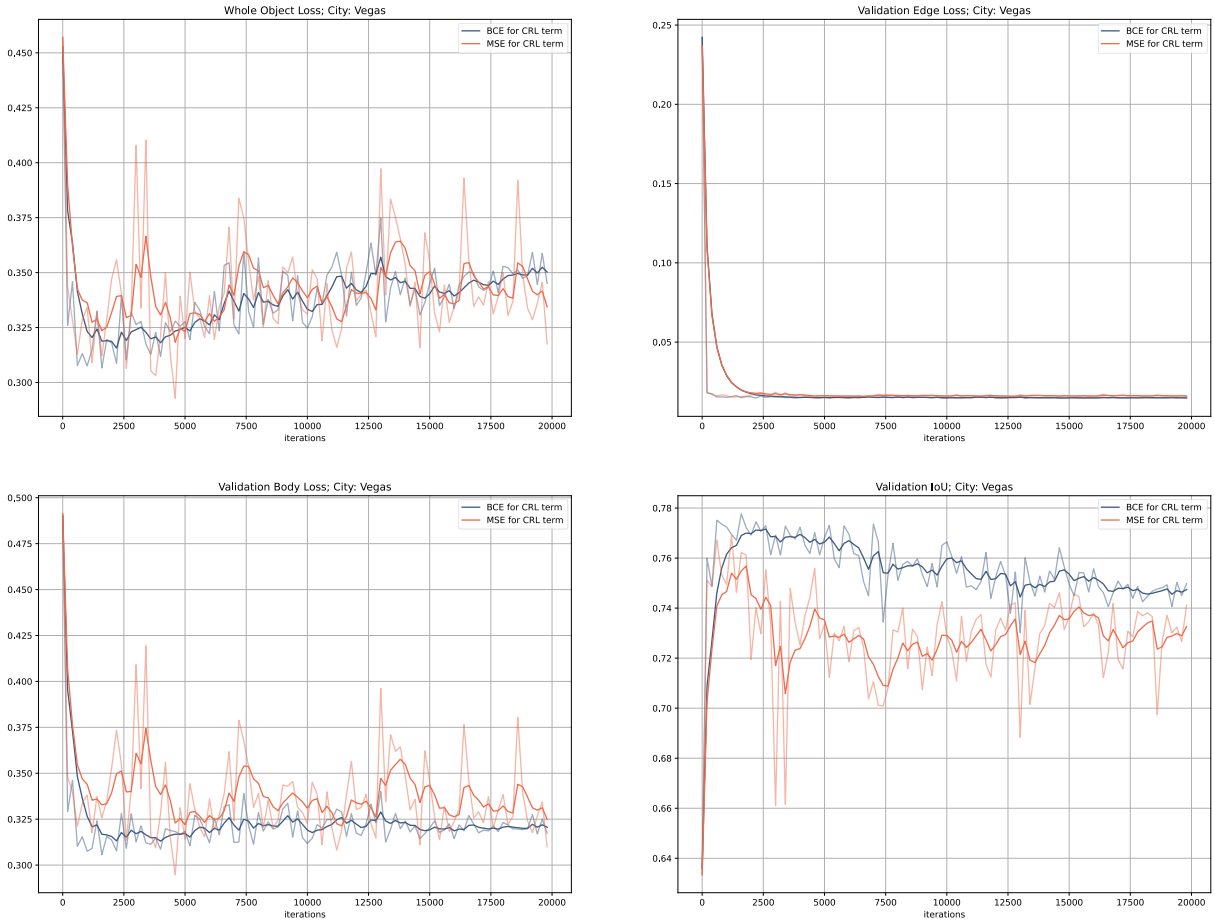



*Figure C.2: BCE versus MSE validation curves for the Vegas dataset.*

Figure C.3 compares the validation and training IoUs on the Source and Target domains, respectively, for both CRL functions.

The results indicate that even though the fused and body training losses retrieved using the MSE loss are significantly lower, the use of BCE improves the model's IoU on the target domain. A reasonable explanation would be that the higher regularization loss values (Figure C.1-bottom left) produced when applying the BCE loss may further prevent the model from overfitting the training data.

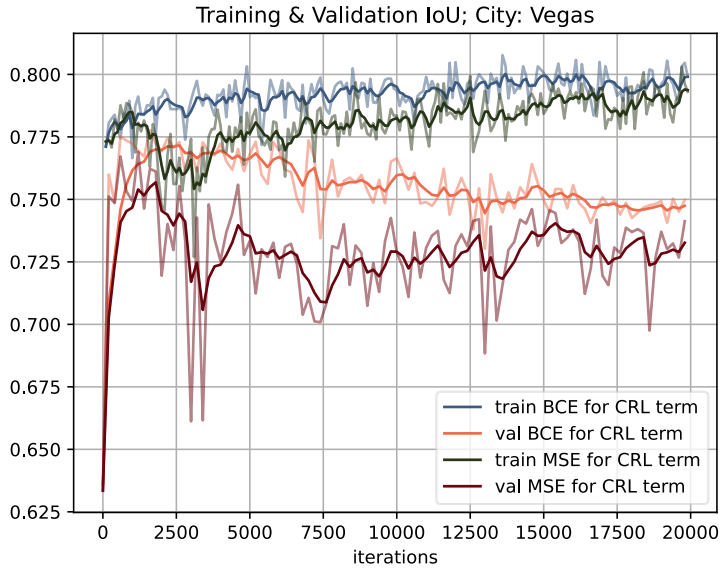



*Figure C.3:Training and Validation IoU for models trained with BCE and MSE as CRL on the Vegas dataset.*

The experiment was repeated for the Shanghai target domain and led to similar results. Figure C.4 presents the training curves for the various supervised components and the CRL terms on the source and target domain for both models. Figure C.5 provides the loss and IoU learning curves on the target domain and Figure C.6 compares the training (source domain) and validation (target domain) IoU curves for both models. The results corroborate the findings of the first experiment, with the model trained using the MSE CRL term performing better on the training dataset but worse on the target domain. Once again, the regularization terms produced with the BCE loss function are higher compared to the MSE CRL terms.

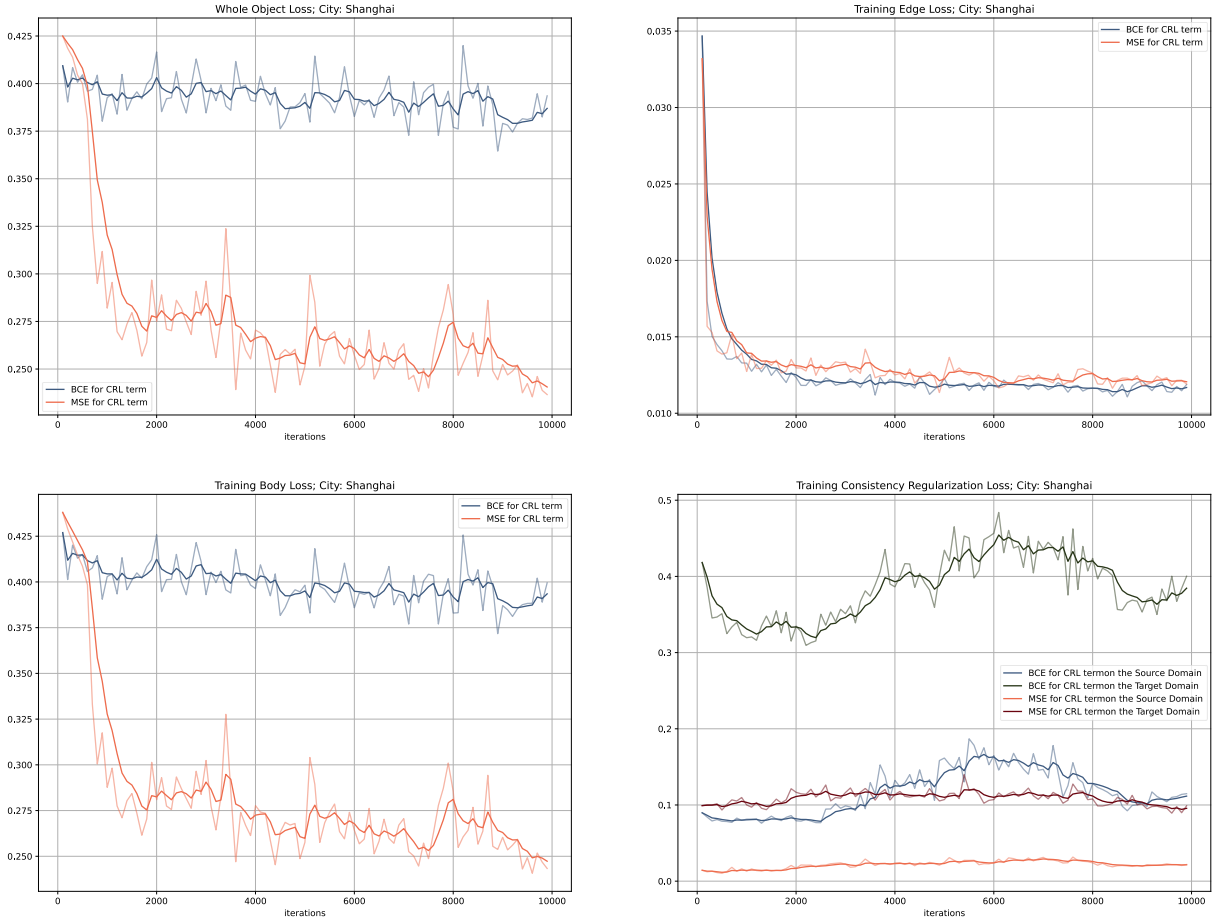



*Figure C.4: BCE versus MSE training curves for the Shanghai dataset.*

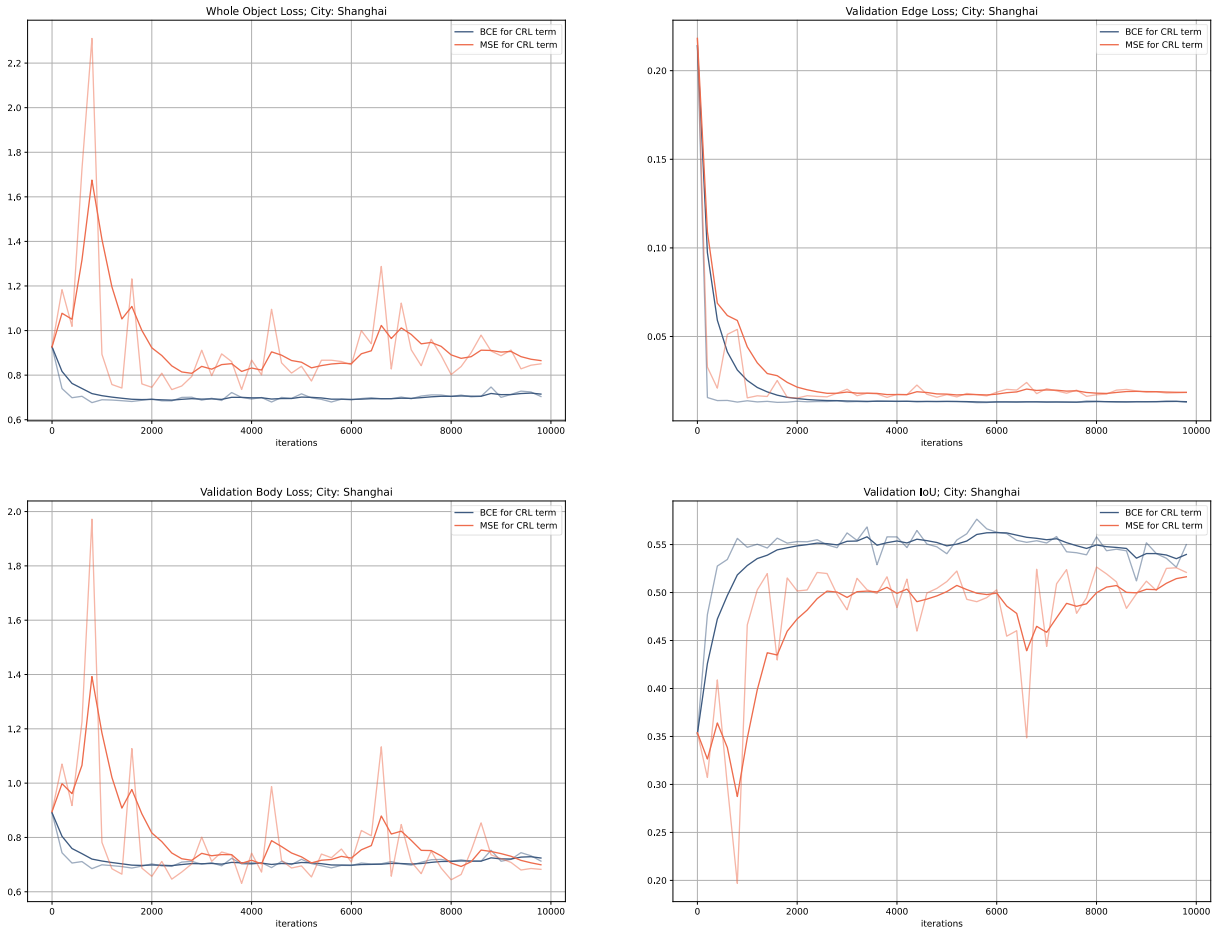



*Figure C.5: BCE versus MSE validation curves for the Vegas dataset.*

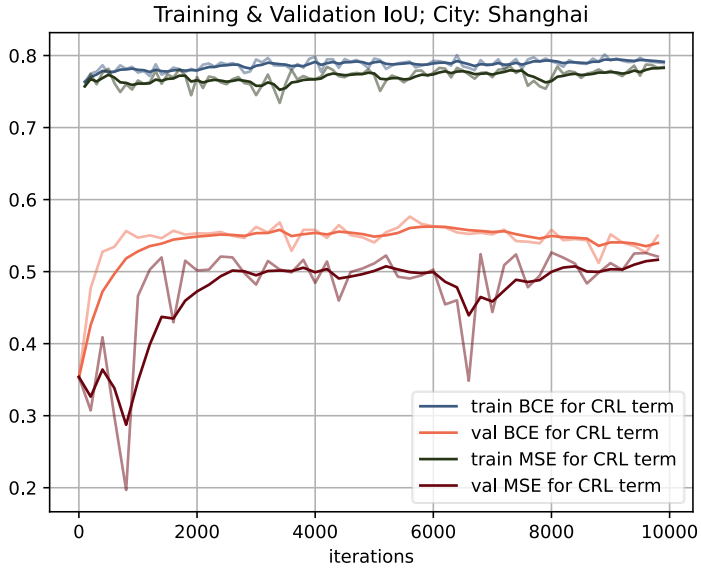



*Figure C.6: Training and Validation IoU for models trained with BCE and MSE as CRL on the Shanghai dataset.*

# Appendix D.

# CutMix Implementation Details

In Chapter 7, we experimented with CutMix augmentations following the conclusions presented by French et al. (2019) that highlight the benefits of using strong and varied augmentations, such as CutMix (Yun et al., 2019) and Cutout (DeVries & Taylor, 2017) in the context of semi-supervised learning. We implemented a CutMix variation with varying patch window dimensions and constraints on the aspect ratio of the patch and the maximum area of the initial image – also expressed as a ratio over the total area – that it can cover. Given the width, $W$ and the height $H$ of the input image the random shape of the CutMix window is regulated by three parameters:

- The maximum area to be cut as a ratio of the original image, $r_{cut}$ (Eq. D.1), and
- The maximum and minimum aspect ratio of the CutMix rectangle $r_{min}$ and $r_{max}$ of dimensions $w$ and $h$ (Eq. D.2).

$$w \cdot h \leq r_{cut} \cdot W \cdot H \tag{D.1}$$

$$r_{min} \cdot w \leq h \leq r_{max} \cdot w \tag{D.2}$$

For all our experiments we set $r_{cut} = 0.25$, $r_{min} = 0.5$ and $r_{max} = 2$. Algorithm D.1 describes the proposed steps to create random CutMix bounding boxes as a Python function.

*Algorithm D.1: Implementation of the proposed CutMix variation in Python.*

```
import numpy as np

# size is the shape of the images' batch: (N, C, H, W)
# where N=number of images, C=number of channels, W=image width and H=image height

def rand_bbox(size, cut_rat=0.25, r_min=0.5, r_max=2):
    W = size[3]
    H = size[2]

    w_min = int(np.sqrt(W * H * cut_rat * r_min))

    w_max = int(np.sqrt(r_max * W * H * cut_rat))

    cut_w = np.random.randint(w_min, w_max)

    cut_h = np.random.randint(int(cut_w / 2), int(cut_w * 2))
```

```
    # uniform
    cx = np.random.randint(W)
    cy = np.random.randint(H)

    bbx1 = np.clip(cx - cut_w // 2, 0, W)
    bby1 = np.clip(cy - cut_h // 2, 0, H)
    bbx2 = np.clip(cx + cut_w // 2, 0, W)
    bby2 = np.clip(cy + cut_h // 2, 0, H)

    return bbx1, bby1, bbx2, bby2
```

Figure D.1 shows an example of our CutMix implementation on a batch of 8 images.


*Figure D.1: CutMix examples. The first row shows the original image, followed by the original image overlayed by the ground truth building footprints in red. In the third row we can see the new images with the random CutMix patch and finally in the last row we present the final augmented images with the building annotations of the CutMix patch shown in purple.*

# Appendix E.

# Examination of the Effects of Batch Size

While processing the building semantic segmentation results using different batch sizes for the target domain metrics, we noticed significant variations of the metrics on certain city datasets when using the exact same model and only varying the batch size used to compute the metrics. Since we use batch normalization in all our models, we use a batch size of 8 for both training and validation purposes. However, when computing the validation metrics, the calculations can be performed in one of the following ways:

- on a per image basis
- or on a per batch size basis, where batch size was set to 8,
- or globally.

In the first two cases the final metrics are the averaged metric values across all images / batches while in the last case the metrics are calculated globally by aggregating the total number of true positives, true negatives, false positives and false negatives for the entire dataset. In this subsection we present an analysis of the results on the various target domains for the decoupled edge and body model trained using the MT framework, dropout layers and the additional entropy regularization term when varying the batch size.

**Vegas**

**Error! Reference source not found.** presents the evaluation metrics for different batch sizes on the Vegas dataset. We can see that the difference between the metrics calculated using a batch size of eight and using the entire dataset (all) is less than 0.2 % for all metrics while the difference between the values for batch size equal to 1 and for batch size equal to 8 can be higher than 3 % (F1 score). This is likely due to the fact that certain images have a very small number of pixels classified as buildings – which equals the sum of TP and FN – if any at all as can be seen in Figure E.1. The precision, recall, F1 score and IoU metrics in these marginal cases can be sensitive to even a very small number of FP or FN and lead to values close to 0 or to 1 that do not necessarily reflect the actual performance of the model on a more balanced sample. The

effect would be negligible if the number of such cases was relatively small, but as we can see in Figure E.1 it is around 10% (more than 90 out of 1000 images where TP + FN = 0). By increasing the batch size to 8 there are no longer such marginal cases (Fig. E.1 right) and the metric values look more stable, as can be seen by the much smaller value differences between columns 3 and 4 of Table E.1.

*Table E.1: Evaluation metrics on the target dataset for various batch sizes. City: Vegas.*

| | batch = 1 | batch = 8 | batch = all |
|---|---|---|---|
| accuracy | 0.9452 | 0.9452 | 0.9452 |
| precision | 0.7859 | 0.8103 | 0.8119 |
| recall | 0.9198 | 0.9306 | 0.9316 |
| F1 | 0.8304 | 0.8659 | 0.8677 |
| IoU | 0.7338 | 0.7641 | 0.7663 |

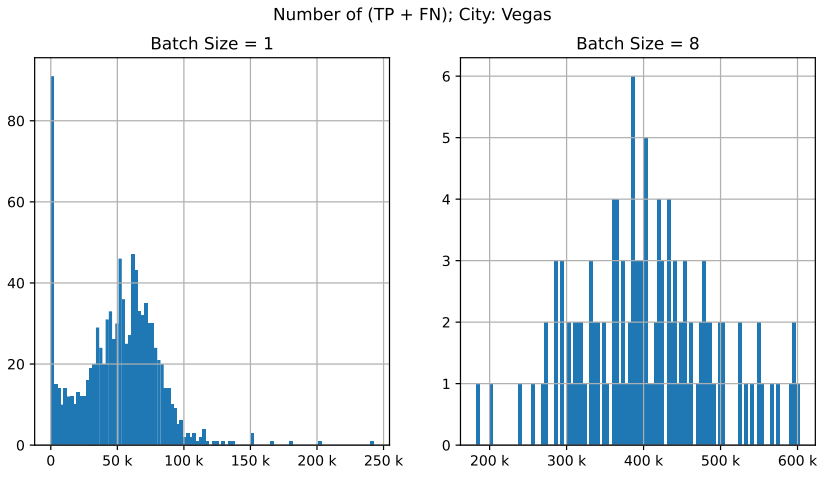



*Figure E.1: Histogram of the number of positive pixels in the ground truth mask for the Vegas dataset.*

## Shanghai

*Table E.2: Evaluation metrics on the target dataset for various batch sizes. City: Shanghai.*

| | batch = 1 | batch = 8 | batch = all |
|---|---|---|---|
| accuracy | 0.9190 | 0.9190 | 0.9190 |
| precision | 0.6768 | 0.6594 | 0.6524 |
| recall | 0.7857 | 0.8012 | 0.8085 |
| F1 | 0.6802 | 0.7106 | 0.7221 |
| IoU | 0.5677 | 0.5628 | 0.5651 |

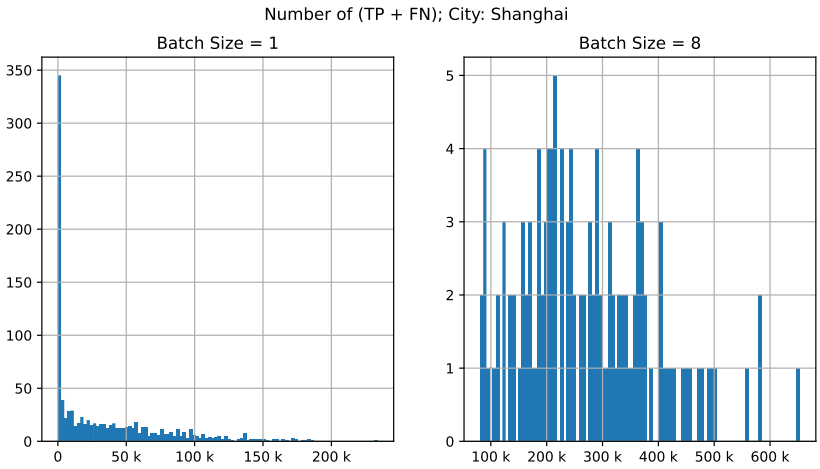



*Figure E.2: Histogram of the number of positive pixels in the ground truth mask for the Shanghai dataset.*

## Paris

*Table E.3: Evaluation metrics on the target dataset for various batch sizes. City: Paris.*

| | batch = 1 | batch = 8 | batch = 8 (filtered) | batch = all |
|---|---|---|---|---|
| accuracy | 0.9584 | 0.9584 | 0.9540 | 0.9584 |
| precision | 0.7862 | 0.6036 | 0.6677 | 0.6761 |
| recall | 0.9061 | 0.7588 | 0.8394 | 0.8667 |
| F1 | 0.8011 | 0.6674 | 0.7383 | 0.7596 |
| IoU | 0.7287 | 0.5355 | 0.5924 | 0.6124 |

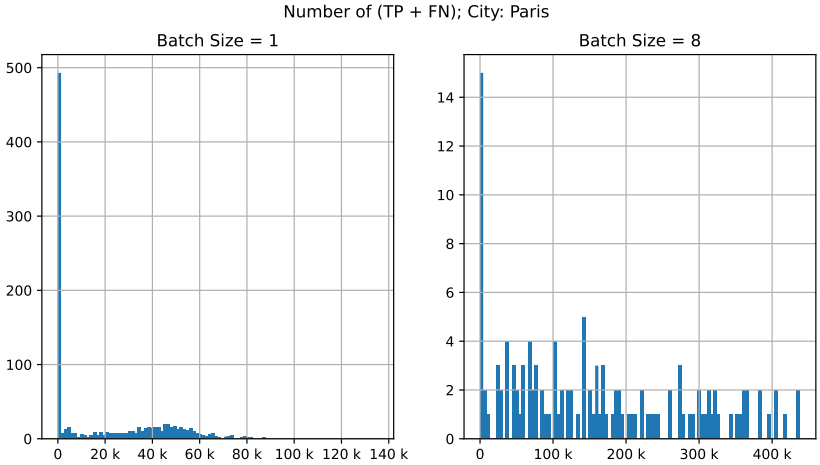



*Figure E.3: Histogram of the number of positive pixels in the ground truth mask for the Paris dataset.*

A good example of a highly imbalanced dataset is Paris, where about half of the 1000 images do not contain any buildings (Fig. E.3 left). In this case the differences between the results with batch size 1 and the ones with batch size 8 can be as high as 15% (Table E.3) and the metrics differ considerably even between the batch size 8 case and the globally computed metrics. This can be explained by the fact that even when using 8 images per batch we still get a high number of samples where TP + FN=0 (Figure E.3). By filtering out these marginal samples (column 4 in Table E.3) we get results that are much closer to the globally computed metrics.

**Khartoum**

The results pertaining to the Shanghai (Table E.2 and Figure E.2) and Khartoum (Table E.4 and ) datasets lead to conclusions similar to the ones presented for the Vegas dataset. The analysis of all datasets suggests that using a batch size of 8 images when computing the evaluation metrics, produces stable results (except for the Paris dataset) in the sense that they are within 1% from the "globally" computed results.

*Table E.4: Evaluation metrics on the target dataset for various batch sizes. City: Khartoum.*

| | batch = 1 | batch = 8 | batch = all |
|---|---|---|---|
| accuracy | 0.8687 | 0.8687 | 0.8687 |
| precision | 0.6912 | 0.6695 | 0.6735 |
| recall | 0.6355 | 0.6841 | 0.6937 |
| F1 | 0.6149 | 0.6713 | 0.6834 |
| IoU | 0.4735 | 0.5099 | 0.5191 |

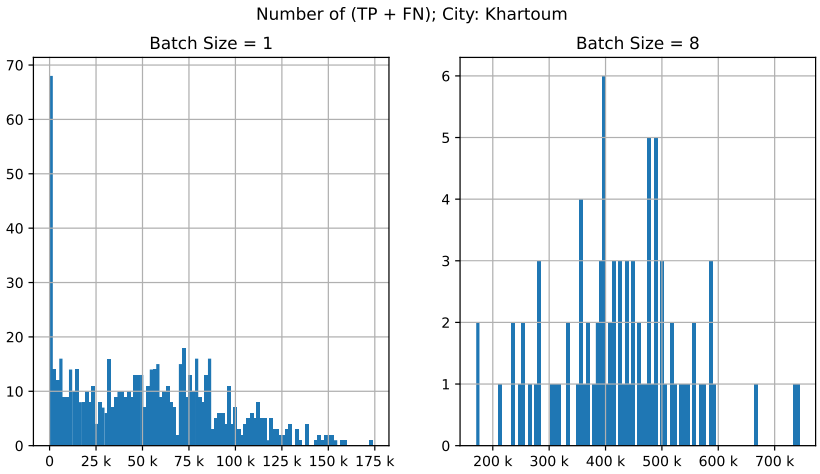



*Figure E.4: Histogram of the number of positive pixels in the ground truth mask for the Khartoum dataset.*