Unsupervised Domain Adaptation applied to Agriculture and Urban Forests using Transformers, GANs and Diffusion Models

Alessandro dos Santos Ferreira

Unsupervised Domain Adaptation applied to Agriculture and Urban Forests using Transformers, GANs and Diffusion Models[1]

Alessandro dos Santos Ferreira

Advisor: *Prof Dr Wesley Nunes Gonçalves*

Thesis presented to the Federal University of Mato Grosso do Sul - UFMS as part of the requirements necessary to obtain the Doctorate Degree in Computer Science.

FACOM - UFMS - Campo Grande
September/2024

[1]This work has received financial support from FUNDECT

À minha família

e aos meus amigos
que me acompanharam
nessa jornada.

Agradecimentos

Gostaria inicialmente de agradecer a toda minha família, em especial ao meu pai José de Oliveira, minhas irmãs Kelly, Simone e Ester, e minha sobrinha Amanda, por todo o apoio oferecido ao longo desses anos de estudo.

Agradeço também a todos os amigos do grupo Inovisão que me acompanharam durante o mestrado, em especial ao professor Hemerson Pistori, meu orientador do mestrado, que sempre me auxiliou com comentários nos artigos publicados e no texto da minha qualificação.

Aos professores José Marcato e Farid Melgani, agradeço por toda contribuição no artigo resultante da minha qualificação, aos professores Lucas Ribas, Jonathan Andrade e Lucas Osco pelos comentários importantíssimos no texto e, em especial, ao meu orientador, professor Wesley Nunes Gonçalves, agradeço por todo suporte, direcionamento e acompanhamento durante este doutorado.

Também não posso deixar de mencionar a oportunidade de realizar inúmeros experimentos utilizando a plataforma gratuita oferecida pelo Google Colab, assim como a NVIDIA, por fornecer as GPUs Titan X, que foram de suma importância para esses experimentos.

Agradeço pelo apoio financeiro da Universidade Federal de Mato Grosso do Sul e da Fundação de Apoio ao Desenvolvimento do Ensino, Ciência e Tecnologia do Estado de Mato Grosso do Sul (FUNDECT), assim como o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

Por fim, dedico este trabalho à memória da minha mãe, Florinda do Santos Ferreira, que, embora não teve a chance de estar presente para ver os passos que trilhei para chegar até aqui, foi a principal responsável por me colocar neste caminho.

Abstract

Deep learning architectures, such as ConvNets, represented an impressive advance in the field of machine learning and are continually breaking records in numerous areas of artificial inteligence, such as image recognition. Nevertheless, the success of these architectures depends on a large amount of labeled data. The annotation of training data is a costly process often performed manually. In agricultural and urban forest problems, differences in image acquisition conditions, such as the height of capture, different sensors, soil conditions, crop stages, and lighting, often necessitate retraining models as new acquisitions are made. In this context, domain adaptation presents itself as a promising alternative to deal with this situation. Domain adaptation consists of adapting the knowledge learned from a source domain to apply it in a target domain, different but related to the original. The aim of this work is to use the domain adaptation approach to find solutions that address problems requiring large amounts of annotated data. Our focus consist of problems related to agriculture and urban forests, using recents architectures used in the unsupervised domain adaptation, such as Generative Adversarial Networks, Vision Transformers, and Diffusion Models. In this work, we propose an approach to address the problem of detecting crop rows and gaps using dilation to generate approximate segmentation maps from annotated one-pixel-wide lines. This method speeds up the pixel labeling process and reduces the line detection problem to semantic segmentation. We use the transformer-based model DAFormer to evaluate the ability to transfer the knowledge learned from source datasets to target datasets. Additionally, we propose a method for segmenting trees that integrates domain adaptation with image-to-image translation models and super-resolution networks to enhance the quality of low-resolution aerial images. Our method also aims to address the challenge of limited labeled data by employing data augmentation to generate additional high-resolution training samples from the existing labeled data, thereby improving model performance and reducing the need for costly label-

ing processes.

Resumo

Arquiteturas de aprendizado profundo, como Redes Neurais Convolucionais, representaram um enorme avanço na área de aprendizado de máquina e vèm continuamente quebrando recordes em inúmeras áreas da inteligência artificial como o reconhecimento de imagens. Todavia, o sucesso dessas arquiteturas é dependente de uma grande quantidade de dados rotulados. Essa anotação dos dados de treinamento consiste em um processo dispendioso e frequentemente realizado de forma manual. Nos problemas relacionados à agricultura e florestas urbanas, devido a diferenças nas condições de aquisição das imagens, por fatores como altura de captura, diferentes sensores, condição do solo, estágios da cultura e iluminação, é comum que os modelos precisem ser novamente treinados a medida que são realizadas novas capturas. Nesse contexto, a adaptação de domínio se apresenta como uma alternativa promissora para lidar com esse problema. A adaptação de domínio consiste em adaptar o conhecimento aprendido em um domínio de origem para aplicá-lo a um domínio destino diferente mas relacionado ao original. O objetivo desse trabalho é utilizar a abordagem de adaptação de domínio para encontrar soluções que lidem com problemas que necessitam de grandes quantidades de dados anotados. Nosso foco consiste em problemas relacionados à agricultura e florestas urbanas, utilizando recentes arquiteturas usadas na adaptação de domínio não supervisionada como as Redes Adversárias Generativas e Vision Transformers e Diffusion Models. Neste trabalho, propomos uma abordagem para resolver o problema de detecção de faixas de plantação e falhas em lavouras, usando dilatação para gerar mapas de segmentação aproximados a partir de linhas anotadas com um pixel de largura. Utilizamos DAFormer, um modelo baseado em transformers, para avaliar a capacidade de transferir o conhecimento aprendido em conjuntos de dados de origem para conjuntos de dados de destino. Além disso, propomos um método para segmentação de árvores que integra adaptação de domínio com modelos de tradução de imagem para imagem e redes de super-resolução para melhorar a qualidade de

imagens aéreas de baixa resolução. Nosso método também visa enfrentar o desafio da limitação de dados rotulados, empregando aumento de dados para gerar amostras adicionais de treinamento em alta resolução a partir dos dados rotulados existentes, melhorando assim o desempenho do modelo e reduzindo a necessidade de processos custosos de rotulagem.

Contents

List of Figures

List of Tables

List of Abbreviations

ADDA Adversarial Discriminative Domain Adaptation

ANNs Artificial Neural Networks

CE Cross-Entropy

CNNs Convolutional Neural Networks

CoGAN Coupled Generative Adversarial Networks

DA Domain Adaptation

DDPMs Denoising Diffusion Probabilistic Models

DMs Diffusion Models

ESRGAN Enhanced Super-Resolution Generative Adversarial Network

F1 Farm 1

F2 Farm 2

F3 Farm 3

F4 Farm 4

FD Feature Distance

FN False Negatives

FP False Positives

GANs Generative Adversarial Networks

GLCM Gray Level Co-occurrence Matrix

GPU Graphics Processing Unit

GSD Ground Sample Distance

GT Ground Truth

GTA Grand Traffic Auto

HOG Histogram of Oriented Gradients

HR High Resolution

JPEG Joint Photographic Experts Group

LAION Large-scale Artificial Intelligence Open Network

LBP Local Binary Patterns

LD Latent Diffusion

LDMs Latent Diffusion Models

LR Low Resolution

IoU Intersection over Union

mIoU mean Intersection over Union

MLP MultiLayer Perceptron

MiT Mix Transformer

MiT-B0 Mix Transformer Base 0

MiT-B5 Mix Transformer Base 5

MSE Mean Squared Error

MVS MultiView Stereo

NLP Natural Language Processing

P Prediction

PVT Pyramid Vision Transformer

P20 P20 dataset

P20D P20 dataset upsampled using Latent Diffusion

P20G P20 dataset upsampled using Real-ESRGAN

P20lr P20 dataset low-resolution (32x32)

P20lD P20lr dataset upsampled using Latent Diffusion

P20lG P20lr dataset upsampled using Real-ESRGAN

P20lp P20lr dataset upsampled using pix2pix

P20S P20 dataset upsampled using Stable Diffusion

P20 P50 dataset

P50D P50 dataset upsampled using Latent Diffusion

P50G P50 dataset upsampled using Real-ESRGAN

P50S P50 dataset upsampled using Stable Diffusion

P50-20p P50 dataset translated using pix2pix trained with P20 pairs

P50-50p P50 dataset translated using pix2pix trained with P50 pairs

RCS Rare Class Sampling

Real-ESRGAN Real-Enhanced Super-Resolution Generative Adversarial Network

SegFormer-B0 SegFormer Base 0

SegFormer-B5 SegFormer Base 5

SIFT Scale Invariant Feature Transform

SfM Structure-from-Motion

SR Super-Resolution

SISR Single Image Super-Resolution

SRGAN Super-Resolution Generative Adversarial Network

SVM Support Vector Machine

TP True Positives

UAVs Unmanned Aerial Vehicles

UDA Unsupervised Domain Adaptation

ViTs Vision Transformers

VRAM Video Random Access Memory

CHAPTER 1

Introduction

1.1 Motivation

With the continuous increase in the global population, it is expected that there will be a corresponding increase in food production, maintaining the high quality of products while protecting natural ecosystems using sustainable agricultural procedures. In this context, it's necessary to monitor, measure and analyze agricultural ecosystems, making use of new technologies. Remote sensing can be used to monitor these areas, being carried out by satellites, planes and, more recently, given the cost reduction, by unmanned aerial vehicles (UAVs). With the use of remote sensing it's possible to obtain images of large geographic areas at low cost.

This large amount of images collected can be used to address a wide variety of challenges present in agriculture and urban forests. Computer vision emerges as a great alternative for automating the analysis of these images. In image analysis, the steps of attribute extraction and classification are usually necessary. For attribute extraction, among the most commonly used algorithms are Scale Invariant Feature Transform (SIFT) (Lowe, 2004), Gray Level Co-occurrence Matrix (GLCM) (Soh and Tsatsoulis, 1999), Histogram of Oriented Gradients (HOG) (Dalal and Triggs, 2005), and Local Binary Patterns (LBP) (Ahonen et al., 2006), in addition to color statistics. Regarding classifiers, we can mention Support Vector Machine (SVM) (Cortes and Vapnik, 1995), Random Forests (Liaw and Wiener, 2002), and Artificial Neural Networks (ANNs) (Jain et al., 1996). However, in recent years, the use of deep learning has been standing out and is widely used in the area.

The use of deep learning architectures represented a revolution in the machine learning field, making impressive advances in the state-of-the-art across a series of tasks and applications (LeCun et al., 2015). The ability to learn in multiple layers using automatically extracted attributes has led this type of learning to achieve breakthroughs in many areas of artificial intelligence. These deep architectures are continually breaking records in areas such as image and speech recognition, in addition to achieving great results in natural language processing (LeCun et al., 2015).

However, despite the enormous success of these architectures, the performance gains tend to occur when a large amount of labeled training data is available, making these works dependent on the costly process of manually labeling the data (Ganin and Lempitsky, 2015; dos Santos Ferreira et al., 2019). In addition to requiring a significant amount of annotated data to train effectively, these architectures often struggle when trying to generalize their learning to unseen datasets (Giuffrida et al., 2019). For problems without labeled data, in some cases, it is possible to obtain sufficiently large training sets using, for example, synthetic data. Nonetheless, this training is hampered due to the different distribution of this data in relation to the real data (Ganin and Lempitsky, 2015).

In the agricultural area, several researchers have conducted studies and achieved excellent results using deep learning architectures such as Convolutional Neural Networks (CNNs or ConvNets). These works involve classification and detection tasks such as weed identification, plant recognition, fruit counting, and crop classification. A survey published in 2018 analyzed 40 papers that used deep learning for a range of agriculture-related problems (Kamilaris and Prenafeta-Boldú, 2018). Among all these articles, 37 were published after 2015, demonstrating the recent adoption of deep learning in agriculture. This research found that the use of deep learning, in the vast majority of these works, presented better performance when compared to traditional techniques of classification and manual extraction of attributes.

Nevertheless, pre-trained models often require retraining on new datasets originating from different experiments, even of the same species (Giuffrida et al., 2019). Furthermore, the problem of manually labeling data is particularly critical in the fields of agriculture and urban forests, as it often requires specialists with limited availability (Kamilaris and Prenafeta-Boldú, 2018). Although deep learning-based algorithms have shown promise in developing automated approaches, the extensive labeling efforts required to capture diverse features across different regions can limit their effectiveness for large-scale problems (Zheng et al., 2020).

Additionally, the lack of publicly available datasets has significantly hin-

dered progress in this research area (Kapil et al., 2024). Despite the significant advances achieved through the use of deep learning in agricultural and urban forest problems, challenges persist in achieving the same high precision in practical applications as demonstrated in controlled experiments. Consequently, bridging the gap between source and target domains through knowledge transfer has become essential (Amirkolaee et al., 2024).

1.2 Objectives

Our proposal in this work is to seek solutions to overcome some of the obstacles mentioned in the previous section. As an example of these challenges, we can mention the need for large image datasets required for training, the high cost of manually annotating images in these datasets, the scarcity of specialists available for manual annotation, the difficulty in generalizing certain aspects of learning, the differences in data distribution among different image datasets focused on the same problem, and finally, the lack of extensive image datasets in the field of agriculture and urban forests (Kamilaris and Prenafeta-Boldú, 2018).

In the absence of labeled data for a given task, domain adaptation is presented as an interesting option. Domain adaptation can be defined as a technique in which the objective is to adapt the knowledge learned in a source domain to apply it to a different but related target domain (Tuia et al., 2016). It has become an important area of study to reduce the cost of data annotation, appealing due to its capability to learn mappings between domains where the target domain data is either completely unlabeled (unsupervised domain adaptation) or has limited labeled samples (semi-supervised domain adaptation) (Ganin and Lempitsky, 2015).

Satellite remote sensing has been crucial for monitoring urban forest resources for many years. Nevertheless, the distribution and varied surfaces of urban forests often make it challenging to accurately identify and detect individual trees due to the limited resolution of satellite imagery (Velasquez-Camacho et al., 2023; Lv et al., 2023)). In recent years, high-resolution aerial RGB imagery, that is easy to use and available at low cost, has become widely accessible. Unlike satellite images, UAV-acquired imagery typically includes only three RGB channels, which, while providing limited spectral information, enables clear visualization and extraction of structural characteristics such as shape, size, and texture of ground objects (Ferreira et al., 2020).

In the agricultural sector, most images used in supervised learning methods are obtained through remote sensing. However, due to potential variations in image acquisition conditions, such as illumination, soil conditions, or phe-

nological stages of vegetation, it is common for models to require retraining using samples collected during each image acquisition. In this context, manual labeling may not keep pace with frequent image acquisitions or may not be feasible due to economic constraints (Tuia et al., 2016).

Domain adaptation solutions emerge as a promising alternative to address these challenges in agriculture. In this work, we propose an approach to use unsupervised domain adaptation for detecting sugarcane crop rows and gaps. Our approach involves generating approximate segmentation maps from annotated one-pixel-wide lines using dilation. This method speeds up the pixel labeling process and reduces the line detection problem to semantic segmentation.

We considered the transformer-based method, SegFormer, and compared it with CNN segmentation models, PSPNet and DeepLabV3+, using datasets consisting of aerial images from four distinct sugarcane farms. To assess the transferability of learned knowledge across datasets, we employed a recent and advanced unsupervised domain adaptation model, DAFormer.

We also propose a novel method for segmenting trees that integrates domain adaptation with image-to-image translation models and super-resolution networks to enhance the quality of low-resolution aerial images. Our method tackles the challenge of limited labeled data by employing data augmentation, using image-to-image translation and super-resolution networks, to generate additional training samples from the existing labeled data, thus improving model performance and reducing the need for costly labeling processes.

This approach not only addresses the difficulties associated with segmenting trees in aerial images of varying resolutions and capture heights but also leverages advanced methods such as Real-ESRGAN, Latent Diffusion, and Stable Diffusion. Moreover, the proposed method is adaptable and can be extended to similar detection problems with minimal adjustments.

1.3 Contributions

Overall, the main contributions of our approach to address the challenge of detecting sugarcane crop rows and gaps are: (1) utilizing dilation to generate approximate segmentation maps and reducing crop rows detection to semantic segmentation, (2) evaluating the transformer-based SegFormer alongside CNN segmentation models for crop rows and gaps detection, and (3) assessing the performance of the state-of-the-art UDA model, DAFormer, in generalizing segmentation knowledge across four distinct sugarcane farms using a generic procedure which could be easily adapted to similar problems in agriculture.

Our primary contributions to the tree detection problem in aerial images

are as follows: (1) employing domain adaptation techniques to address the challenges of segmenting trees in aerial images of different resolutions and captured at different heights; and (2) utilizing data augmentation methods to overcome the scarcity of labeled data in this domain, using image-to-image translation models and the recent super-resolution networks Real-ESRGAN, Latent Diffusion, and Stable Diffusion.

1.4 Organization

This work is organized as follows. Section 2 provides a introduction to the topic of unsupervised domain adaptation and describes the techniques currently employed in this approach. Section 3 introduces the problem of sugarcane crop rows and gaps detection, detailing the methodology used for conducting experiments and presenting the results and their discussion. Section 4 addresses the tree detection problem, outlining the methodology used and discussing the results obtained from the experiments. Section 5 concludes the work and outlines future research directions.

CHAPTER

2

Unsupervised Domain Adaptation

Deep learning architectures, when trained on large-scale datasets, can learn representations that are generally useful across various visual tasks. However, due to a phenomenon known as dataset bias or domain shift (Gretton. et al., 2009), models trained with these representations often do not generalize well to new datasets and tasks. Domain adaptation methods aim to mitigate the impacts caused by domain shift.

There are several approaches to addressing problems using unsupervised domain adaptation. Generating synthetic data is an interesting alternative for cases where image datasets are not large enough to achieve satisfactory accuracy. However, without domain adaptation, there can be a significant disparity in the distribution of real and synthetic data for the same class, impairing performance on real data tests (Ganin and Lempitsky, 2015). Additionally, real image datasets often exhibit substantial differences in data distribution due to various factors in image capture (Giuffrida et al., 2019). Domain adaptation can integrate these diverse image datasets, increasing robustness and generalization across different but similar domains.

New techniques are continually being researched to address these challenges. In the following sections, we will discuss recent algorithms used in unsupervised domain adaptation.

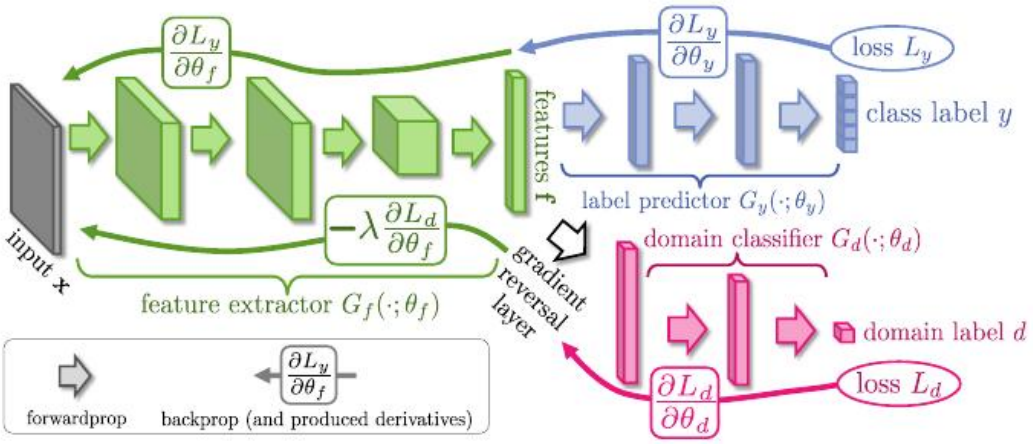

Figure 2.1: Proposed architecture for unsupervised domain adaptation by backpropagation (Ganin and Lempitsky, 2015).

2.1 Unsupervised Domain Adaptation by Backpropagation

Unlike most of the work on domain adaptation published up to that time, this algorithm focuses on combining domain adaptation with deep feature learning in a single training process (Ganin and Lempitsky, 2015). The objective is to incorporate domain adaptation into the learning process so that the final classification is based on attributes that are both discriminative for the problem and invariant to domain changes. This ensures that the resulting neural network can be applied to the target domain without being adversely affected by differences between the two domains.

Let a model use input samples $x \in X$, in a given space X, and labels y in a given space Y, where Y is a finite set $(Y = 1,2,...,L)$. Let there be two distributions $S(x,y)$ and $T(x,y)$ in $X \otimes Y$, referred to as the source distribution and the target distribution (or the source domain and the target domain). The goal of the algorithm is to predict the labels y for the target distribution, given the input x from both distributions.

During training, it is possible to access the training samples $x_1, x_2, ..., x_N$ from the source and target domains, distributed according to the marginal distributions $S(x)$ and $T(x)$. For the source domain examples, defined as $(d_i = 0)$, the corresponding labels $Y_i \in Y$ are known during training. For the target domain examples, defined as $(d_i = 1)$, the labels are not known during training and must be predicted by the algorithm by aligning the feature distributions of the source and target domains.

The proposed architecture is illustrated in Figure 2.1. It includes a feature extractor, shown in green, and a classifier, shown in blue, which together form

a standard deep learning architecture, such as convolutional neural networks. The loss L_y is calculated only for samples from the source domain for which the labels are known. Unsupervised domain adaptation is achieved by adding a domain classifier, shown in red, which tries to distinguish whether a sample comes from the source or target domain, resulting in the loss L_d. The gradient reversal layer inverts the gradient coming from the domain classifier during backpropagation. This process encourages the model to learn features that are invariant to the domain shift, which helps in achieving good performance on the target domain even though the target domain labels are not used during training.

2.2 Generative Adversarial Networks (GANs)

Although they were not originally designed with domain adaptation in mind, generative adversarial networks (GANs) have become the foundation for some of the most successful architectures in unsupervised domain adaptation in recent years. A generative adversarial network consists of a generative model and a discriminative model. The objective of the generative model is to synthesize images that resemble real images, while the discriminative model aims to distinguish real images from the synthesized ones. Both the generative and discriminative models are typically defined as multilayer perceptrons (Goodfellow et al., 2014).

The generative model can be thought of as analogous to a team of counterfeiters trying to produce fake currency and use it without being discovered, while the discriminative model is analogous to the police trying to detect the counterfeit currency. The competition between these two entities drives both to evolve their methods until the fakes are indistinguishable from the genuine ones. The way these networks measure and minimize the discrepancy between the distribution of real and synthesized data is very similar to how the model used by Ganin and Lempitsky (2015) measures and minimizes the discrepancy between data distributions to perform domain adaptation.

Let x be a genuine image obtained from a distribution p_X and z a random vector in $\mathbb{R}^d$. Let G and D be the generative and discriminative models, respectively. The generative model uses z as input and outputs an image, $G(z)$. Let p_Z be the distribution of $G(z)$. The discriminative model estimates the probability that an input image belongs to p_X. In the ideal scenario, $D(y) = 1$ if y is drawn from p_X (i.e., $y \sim p_X$) and $D(y) = 0$ if if y is drawn from p_Z (i.e., $y \sim p_Z$).

The structure of the generative adversarial network corresponds to a Minimax game for two players, represented by the generative and discriminative

models, which can be trained together through equation 2.1:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_X}[logD(x)] + \mathbb{E}_{z\sim p_Z}[log(1-D(G(z)))] \tag{2.1}$$

In practice, the equation 2.1 is solved by alternating the following two gradient update steps described in 2.2:

$$\begin{aligned} \theta_d^{t+1} &= \theta_d^t - \lambda^t \nabla_{\theta_d} V(D^t, G^t) \\ \theta_g^{t+1} &= \theta_g^t + \lambda^t \nabla_{\theta_g} V(D^{t+1}, G^t) \end{aligned} \tag{2.2}$$

where θ_d and θ_g are the parameters of D and G, λ is the learning rate, and t is the iteration number (Tzeng and Tuzel, 2016). Using this strategy, the parameters of the discriminative model are updated with gradient descent. In the subsequent step, using the updated parameters of the discriminator, the parameters of the generative model are updated with gradient reversal descent to decrease the discriminator's accuracy.

2.2.1 Coupled Generative Adversarial Networks (CoGAN)

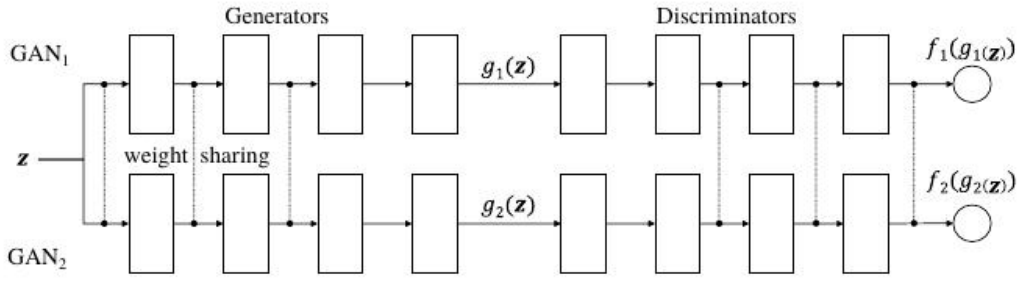

Figure 2.2: Coupled Generative Adversarial Networks (CoGAN) architecture Tzeng and Tuzel (2016).

CoGAN consists of a pair of generative adversarial networks, one for each image domain, with the restriction that they share the weights of some layers (Tzeng and Tuzel, 2016). The CoGAN architecture is inspired by the idea that deep convolutional neural networks learn a hierarchical representation of features. By enforcing sharing on the layers that decode high-level semantics, the two generative adversarial networks are compelled to decode these semantics in the same manner. Layers that decode low-level information map the shared representation to the images in individual domains to mislead the respective discriminative models.

Figure 2.2 shows the architecture of a CoGAN. It consists of a pair of generative adversarial networks: GAN_1 and GAN_2. Each GAN has a generative model for synthesizing realistic images in a domain and a discriminative model for

classifying whether an image is real or synthesized. The weights of the first layers (responsible for decoding high-level semantics) of the generative models, g_1 and g_2, are shared. The weights of the last layers (responsible for encoding high-level semantics) of the discriminative models, f_1 and f_2, are also shared. This weight-sharing constraint allows CoGAN to learn a joint distribution of images without the need for correspondence supervision. A trained CoGAN can be used to synthesize pairs of images that share the same high-level abstraction but have different low-level characteristics.

2.2.2 Adversarial Discriminative Domain Adaptation (ADDA)

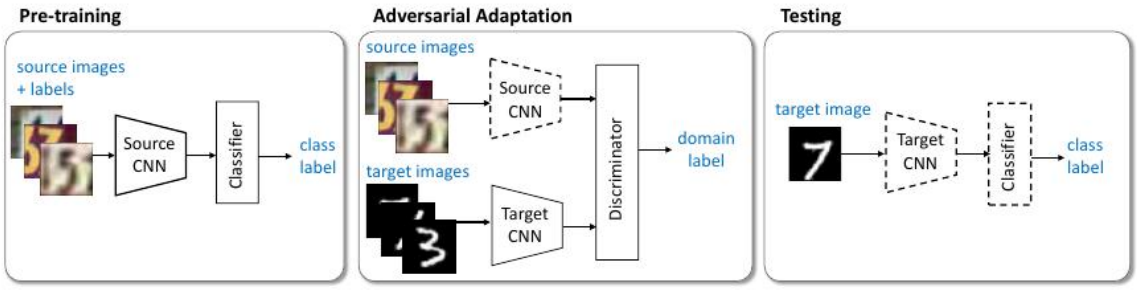

Figure 2.3: Overview of the approach used in Adversarial Discriminative Domain Adaptation (ADDA). Dashed lines indicate layers with fixed parameters (Tzeng et al., 2017).

Domain adaptation methods using adversarial loss have led to approaches that aim to minimize the domain shift distance using an adversarial objective with respect to the domain discriminator. One such approach is the Adversarial Discriminative Domain Adaptation (ADDA) method (Tzeng et al., 2017). An overview of this method can be seen in Figure 2.3.

Initially, a convolutional neural network is trained using labeled images from the source domain. Then, adversarial adaptation is performed by learning the parameters of a convolutional network for the target domain. This step is carried out in such a way that a discriminator, which sees images from both domains encoded by their respective networks, is unable to accurately predict the correct domain of these images. During testing, the images from the target domain are mapped to a vector of attributes resulting from their training network. This attribute vector is then used as input to the classifier trained on the source domain.

2.3 Image to Image Translation

Image-to-image translation is a domain within computer vision that focuses on learning the mapping between an input image and a corresponding output

image using a training set of aligned image pairs. In paired training data, the dataset comprises examples $\{x_i, y_i\}_{i=1}^{N}$, where each input image x_i has a corresponding output image y_i, establishing a clear relationship between the pairs, as illustrated in Figure 2.4.

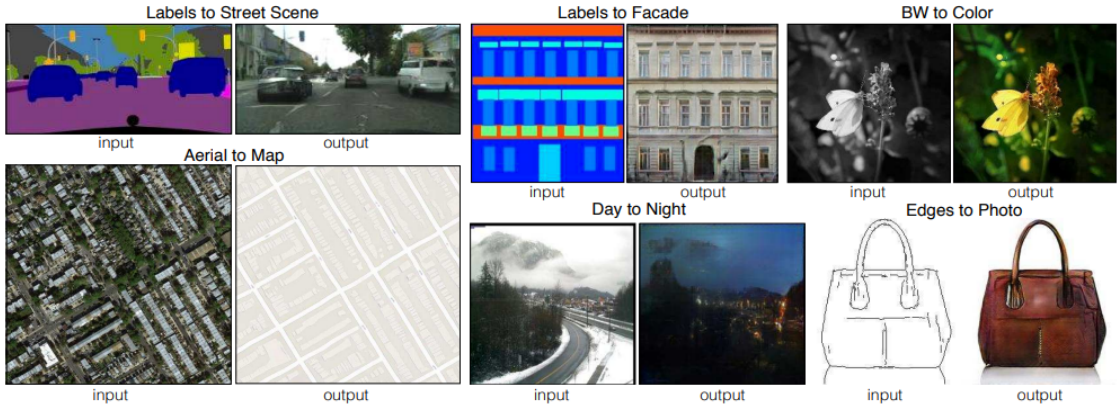

Figure 2.4: Examples of image pairs used in image-to-image translation, as shown by Isola et al. (2017).

In contrast, unpaired training involves two separate sets: a source set $\{x_i\}_{i=1}^{N}$, where $x \in X$, and a target set $\{y_j\}_{j=1}^{M}$, where $y \in Y$, without any explicit information linking a specific source image x_i to a specific target image y_j (Zhu et al., 2017). It is important to note that the target set Y in the context of image-to-image translation is distinct from the labels Y used in supervised learning tasks.

This problem can be broadly characterized as converting an image from one representation of a given scene, A, to another representation, B. Examples include transforming a grayscale image to a color image, converting an image to semantic labels, or generating a photograph from an edge-map.

Acquiring paired training data for these tasks can be challenging and costly. For instance, there are only a few datasets available for tasks like semantic segmentation, and they are relatively small. To address this issue, various methods have been developed to perform both unpaired and paired image-to-image translation.

2.3.1 CycleGAN (Impaired)

Adversarial training theoretically enables learning mappings G and F that generate outputs with distributions matching those of the target domains. However, given sufficient network capacity, a model might map the same set of input images to any random permutation of images in the target domain. Consequently, the learned mappings might produce an output distribution that aligns with the target distribution, but adversarial losses alone do not

ensure that a specific input x_i will correspond to a particular output y_i.

To further constrain the space of possible mapping functions, Zhu et al. (2017) proposed that learned mappings should be cycle-consistent. This means that if, for instance, a sentence is translated from English to French and then back to English, it should return to the original sentence. Similarly, for each image x in domain X, the translation cycle should ideally return x to its original form, i.e., $x \to G(x) \to F(G(x)) \approx x$. This concept, known as cycle consistency, sets the foundation for the CycleGAN method Zhu et al. (2017).

The goal of the method is to learn mapping functions between two domains, X and Y, using training samples $\{x_i\}_{i=1}^{N}$ where $(x \in X)$ and $\{y_j\}_{j=1}^{M}$ where $(y \in Y)$. The data distributions are denoted as $x \sim p_{\text{data}}(x)$ and $y \sim p_{\text{data}}(y)$. Mathematically, if we have a translator $G : X \to Y$ and another translator $F : Y \to X$, then G and F should act as inverses of each other, with both mappings being bijections.

CycleGAN enforces this structural assumption by simultaneously training both mappings, G and F, and incorporating a cycle consistency loss. This loss function encourages the conditions $F(G(x)) \approx x$ and $G(F(y)) \approx y$, ensuring that the mappings are consistent with the original images. The cycle consistency loss is formalized in Equation 2.3:

$$\begin{aligned} \mathcal{L}_{\text{cyc}}(G,F) = \mathbb{E}_{x\sim p_{\text{data}}(x)}\left[\|F(G(x)) - x\|_1\right] \\ + \mathbb{E}_{y\sim p_{\text{data}}(y)}\left[\|G(F(y)) - y\|_1\right] \end{aligned} \tag{2.3}$$

where $\mathbb{E}$ denotes the expectation operator and and $\|_1$ represents the L_1 norm, which measures the absolute differences between the generated and original images.

In addition, CycleGAN employs two adversarial discriminators, D_X and D_Y. The discriminator D_X is tasked with differentiating between real images x and translated images $F(y)$, while D_Y distinguishes between real images y and generated images $G(x)$. The full objective contains two key components: adversarial losses (Equation 2.1), which align the distribution of generated images with the data distribution of the target domain, and cycle consistency losses, which ensure that the mappings G and F do not contradict each other, as illustrated in Equation 2.4:

$$\begin{aligned} \mathcal{L}(G,F,D_X,D_Y) = \mathcal{L}_{\text{GAN}}(G,D_Y,X,Y) + \mathcal{L}_{\text{GAN}}(F,D_X,Y,X) \\ + \lambda\mathcal{L}_{\text{cyc}}(G,F) \end{aligned} \tag{2.4}$$

where λ is a hyperparameter that balances the relative importance of the cycle consistency loss $\mathcal{L}_{\text{cyc}}$ with the adversarial losses $\mathcal{L}_{\text{GAN}}$.

The model can be conceptualized as training two autoencoders (Kingma

and Welling, 2013): one autoencoder $F \circ G : X \to X$ and another $G \circ F : Y \to Y$. These autoencoders, however, have distinctive structures; they map an image to itself through an intermediate representation, which involves translating the image to another domain. This setup can be regarded as a specific instance of adversarial autoencoders (Makhzani et al., 2015), which use an adversarial loss to train the bottleneck layer of an autoencoder to approximate an arbitrary target distribution. In our case, the target distribution for the $X \to X$ autoencoder is the distribution of domain Y.

2.3.2 Pix2pix (Paired)

Although unpaired methods like CycleGAN often succeed in translation tasks involving color and texture changes, Figure 2.5 shows some typical failure cases. Moreover, while the network may produce visually appealing results that preserve local content in natural scenes, it is not specifically designed for end tasks and may not always preserve semantic information. This lack of semantic preservation can be problematic if the translation is used as a preliminary step in a segmentation task (Hoffman et al., 2018).

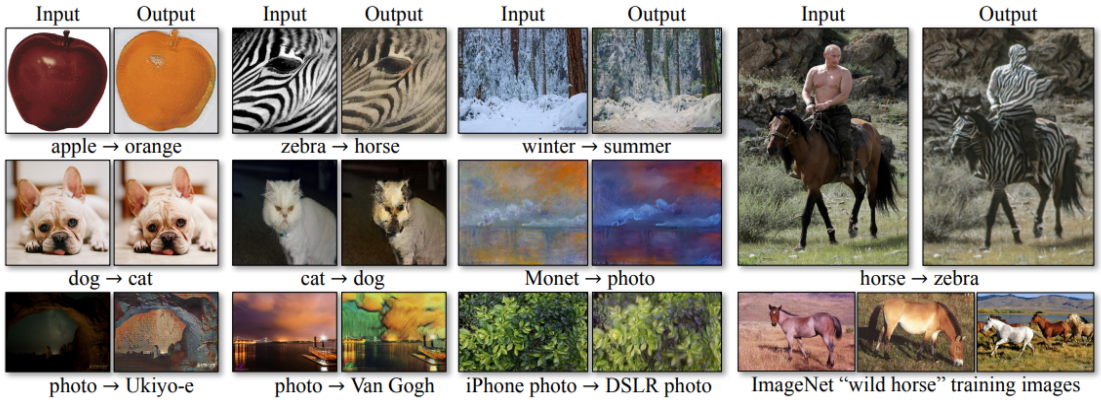

Figure 2.5: Typical failure cases of CycleGAN. According to Zhu et al. (2017), CycleGAN fails, for example, in this horse → zebra pair because the model has not encountered images of horseback riding during training.

On the other hand, in paired image-to-image translation, each input image has a corresponding target image in the dataset. This clear correspondence enables the model to learn a direct mapping from the input to the target domain, often resulting in more accurate and higher-quality transformations (Isola et al., 2017). Since the model is explicitly trained on exact data transformations and does not need to infer correspondences between domains, paired image-to-image translation can yield more consistent and reliable outputs.

This is particularly beneficial for tasks requiring precise mapping, such as segmenting aerial images. However, if a naive approach is taken, such

as training a CNN to minimize the Euclidean distance between predicted and ground truth pixels, it often results in blurry outputs. This blurring occurs because minimizing Euclidean distance averages all possible outputs, leading to a lack of sharpness.

Generative Adversarial Networks (GANs) address this issue by learning a loss function that classifies whether the output image is real or fake, while concurrently training a generative model to minimize this adversarial loss. Blurry images will not be tolerated as they look obviously fake. GANs address this issue by learning a loss function that adapts to the data, making them applicable to a wide range of tasks that traditionally require different loss functions.

Figure 2.6: Different losses induce different quality of results in pix2pix. Each column shows results after being trained under a different loss by Isola et al. (2017).

Building on this concept, Isola et al. (2017) introduced pix2pix, which utilizes a conditional GAN loss for image translation between paired images x and y, as described in Equation 2.5:

$$\mathcal{L}_{\text{cGAN}}(G,D) = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,z}[\log(1 - D(x,G(x,z)))] \tag{2.5}$$

where the generator G aims to minimize the loss, while the adversarial discriminator D seeks to maximize it, with z representing the noise vector.

However, Instead of combining the conditional GAN loss with a traditional loss such as L_2 distance, pix2pix uses L_1 distance. This choice is made because L_1 distance tends to produce less blurring compared to L_2 distance. The L_1 loss is defined in Equation 2.6:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}\left[\|y - G(x,z)\|_1\right] \tag{2.6}$$

The final pix2pix objective is outlined in Equation 2.7:

$$\mathcal{L}(G,D) = \mathcal{L}_{\text{cGAN}}(G,D) + \lambda \mathcal{L}_{L1}(G) \tag{2.7}$$

where λ is a hyperparameter that controls the relative importance of the L_1 loss compared to the adversarial loss. Figure 2.6 demonstrates the results

obtained by training with these different types of losses.

2.4 Vision Transformers (ViTs)

Transformers were proposed by Vaswani et al. (2017) for machine translation and have demonstrated remarkable performance in language tasks such as text classification, machine translation, and question answering. These impressive results with transformer models in the natural language processing (NLP) domain have attracted the attention of the vision community, leading to efforts to adapt these models for vision and multi-modal learning tasks.

Transformer architectures are based on a self-attention mechanism that learns the relationships between elements of a sequence. The model was first developed for language translation tasks, where an input sequence of words in one language is required to be converted into an output sequence in another language (Khan et al., 2021). The most commonly used approach with transformers in NLP tasks involves pre-training on a large text corpus followed by fine-tuning on a smaller, task-specific dataset.

Subsequently, Dosovitskiy et al. (2020) proposed the Vision Transformer (ViT) for image classification. Vision Transformer was the first work to demonstrate how transformers can replace standard convolutions in deep neural networks for large-scale image datasets. Following the original transformer design in NLP (Vaswani et al., 2017), they applied transformers to a sequence of images flattened into vectors.

The model was pre-trained on a large proprietary JFT-300M dataset (Sun et al., 2017), which contains 300 million images, and then fine-tuned for downstream recognition tasks on other datasets, such as ImageNet (Deng et al., 2009). This pre-training was necessary because, unlike convolutional or recurrent architectures, transformers assume minimal prior knowledge of the problem structure. Consequently, they typically need to be pre-trained on large-scale (unlabeled) datasets before being fine-tuned on the target task with a smaller labeled dataset (Khan et al., 2021).

With Vision Transformer, Dosovitskiy et al. (2020) achieved excellent results in image recognition benchmarks compared to state-of-the-art ConvNets, demonstrating that transformers can be competitive with convolutional neural networks on large-scale image datasets. Since then, ViTs have gained significant attention from researchers, and several recent approaches have been proposed based on ViTs, including SegFormer, a semantic segmentation architecture that we used and evaluated in this research.

2.4.1 SegFormer

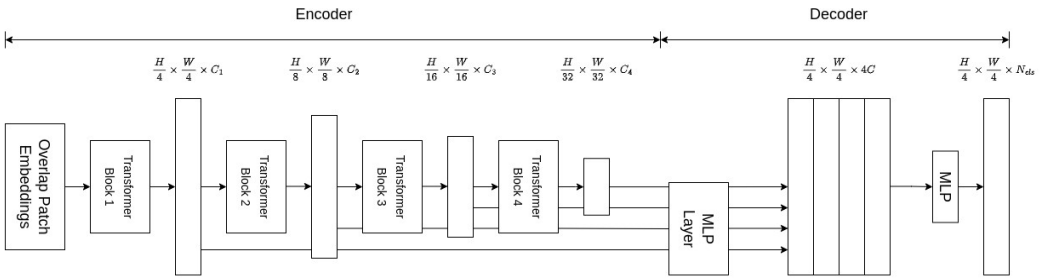

Figure 2.7: Simplified overview of the SegFormer framework, based on the illustration in the original article (Xie et al., 2021). The framework consists of two main modules: a hierarchical transformer encoder for feature extraction and a lightweight MLP decoder for predicting the semantic segmentation mask.

SegFormer is a semantic segmentation framework that unifies transformers with lightweight multilayer perceptron (MLP) decoders (Xie et al., 2021). Semantic segmentation can be viewed as an extension of image classification, as it produces predictions at the pixel level rather than the image level. For this reason, many semantic segmentation frameworks are variants of popular architectures for image classification on ImageNet and use CNNs as their backbone.

Despite its good performance, Vision Transformer (ViT) has two significant limitations: it outputs single-scale low-resolution features and incurs a very high computational cost when processing large images. To address these limitations, Wang et al. (2021a) proposed the Pyramid Vision Transformer (PVT), demonstrating the potential of a pure transformer backbone compared to CNN counterparts in dense prediction tasks.

SegFormer employs a Mix Transformer (MiT) as its backbone. The architecture consists of two main modules: a hierarchical transformer encoder that outputs multiscale features without needing positional encoding, and a lightweight MLP decoder that predicts the final mask by aggregating information from different layers. This combination of local and global attention yields powerful representations (Xie et al., 2021). Given an image of size $H \times W \times 3$, unlike ViT, which uses patches of size 16×16, SegFormer splits the image into patches of size 4×4 because fine-grained patches favor semantic segmentation.

These patches are input to the hierarchical transformer encoder, producing multi-level features at resolutions of $\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$ of the original image. These features are then fed into the MLP decoder to predict the segmentation mask at a resolution of $\frac{H}{4} \times \frac{W}{4} \times N_{cls}$, where N_{cls} represents the number of categories,

as illustrated in Figure 2.7.

The Mix Transformers (MiT) models used as encoders in SegFormer range from MiT-B0 to MiT-B5. They differ primarily in the number of transformer blocks and embedding dimensions. MiT-B0 is the smallest and most efficient model, while MiT-B5 is the largest, providing the best performance with more transformer blocks and higher dimensions, though it requires increased computational resources. While SegFormer-B0, which uses MiT-B0, is a compact and efficient model showing competitive performance, SegFormer-B5, which uses MiT-B5, is the largest model and has achieved state-of-the-art results on tested datasets, demonstrating the potential of the Mix Transformer encoder.

2.4.2 DAFormer

Acquiring pixel annotations of real-world images for semantic segmentation is a costly process; for instance, it can take up to 3.3 hours to annotate a single image from the Cityscapes dataset (Cordts et al., 2016) under adverse weather conditions. Taking advantage of recent advances promoted by transformers in computer vision to generalize knowledge, DAFormer (Hoyer et al., 2021) is an unsupervised domain adaptation architecture based on SegFormer. It was developed with the aim of adapting more accessible synthetic data, such as from the Grand Theft Auto (GTA) dataset (Richter et al., 2016), to real images without requiring annotations.

In unsupervised domain adaptation, a neural network is trained using only source domain images $x \in X$, where X represents the input space, and labels y in a space Y, with Y being a finite set ($Y = 1, 2, \ldots, L$), under a distribution $S(x,y)$ defined over $X \otimes Y$. The goal is to achieve good performance on target images $x \in X$ from a distribution $T(x,y)$, even though the target labels $y \in Y$ are not available. However, training the neural network with a categorical cross-entropy (CE) loss on the source domain usually results in low accuracy on the target images due to the lack of network generalization (Hoyer et al., 2021).

To address the low performance associated with naive training, most unsupervised domain adaptation (UDA) methods employ strategies based on adversarial training or self-training approaches. Adversarial training methods aim to align the distributions of the source and target domains, whereas self-training networks use pseudo-labels for the target domain. DAFormer employs a self-training approach, considering it to be more stable and currently more effective than adversarial training.

To transfer knowledge from the source to the target domain, self-training approaches utilize student and teacher models, where the teacher network generates pseudo-labels for the target domain data. Additionally, a confidence estimate is produced for the pseudo-labels based on the ratio of pixels exceed-

ing a threshold of the maximum softmax probability. These pseudo-labels, along with their quality estimates, are then used to further train the student network on the target domain.

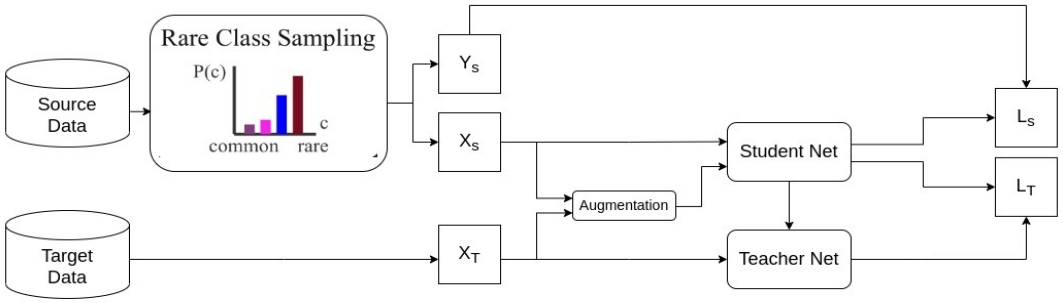

Figure 2.8: Simplified overview of the DAFormer network with rare class sampling (RCS), based on the illustration in the original article (Hoyer et al., 2021). Rare Class Sampling uses images with rare classes from the source domain more often in order to learn them better and earlier.

The DAFormer network consists of a transformer encoder and a multi-level context-aware feature fusion decoder. The DAFormer architecture follows the design of Mix Transformers proposed in SegFormer, dividing the image into patches of size 4×4. The transformer encoder is designed to produce multi-level feature maps $F_i \in \mathbb{R}^{\frac{H}{2^{i+1}} \times \frac{W}{2^{i+1}} \times C_i}$, where H and W are, respectively, the height and width of the images and C represents the number of classes. The architecture incorporates three key strategies to mitigate overfitting to the source domain: Rare Class Sampling (RCS), Feature Distance (FD), and learning rate warmup.

RCS targets uncommon classes in the training process to balance class distribution and enhance model performance on marginalized categories. Feature Distance (FD) measures the similarity between features extracted from the target domain and pre-trained ImageNet features to ensure effective domain alignment. Learning rate warmup gradually increases the learning rate at the beginning of training to stabilize convergence and improve model robustness. A simplified overview of the DAFormer architecture is shown in Figure 2.8.

2.5 Super-Resolution Models

2.5.1 Generative Adversarial Networks for Image Super-Resolution

Single image super-resolution (SR or SISR) is a fundamental low-level vision problem focused on reconstructing a high-resolution (HR) image from its

low-resolution (LR) counterpart. SR has received significant attention from the computer vision research community and has a wide range of applications (Wang et al., 2018). Typically, the objective of supervised SR algorithms is to minimize the mean squared error (MSE) between the recovered HR image and the ground truth.

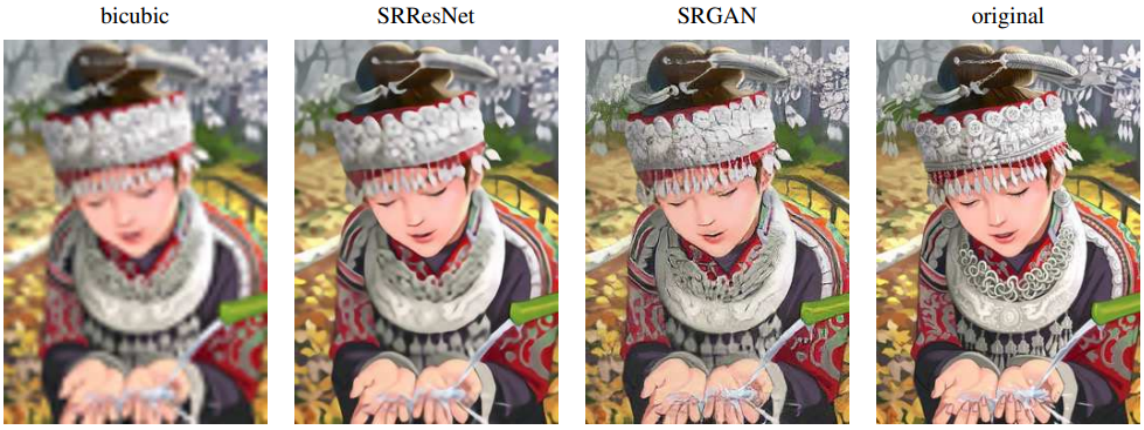

Figure 2.9: Comparison of SRGAN (Ledig et al., 2017) with bicubic interpolation, deep residual network (He et al., 2016) optimized for MSE, and the original HR image.

However, MSE's capacity to capture perceptually relevant differences, such as fine texture details, is quite limited because it is defined based on pixel-wise image differences (Ledig et al., 2017). Pixel-wise loss functions like MSE struggle to account for the uncertainty involved in recovering lost high-frequency details, such as texture. Minimizing MSE often leads to pixel-wise averages of plausible solutions, which tend to be overly smooth and thus exhibit poor perceptual quality.

Ledig et al. (2017) proposed the Super-Resolution Generative Adversarial Network (SRGAN), which employs a deep residual network (ResNet) (He et al., 2016) with skip connections and diverge from MSE as the sole optimization target. A super-resolution image generated using this method can be seen in Figure 2.9.

Unlike previous methods, SRGAN introduces a novel perceptual loss that leverages high-level feature maps from the VGG network (Simonyan and Zisserman, 2014), combined with a discriminator that encourages the generated images to be perceptually indistinguishable from high-resolution (HR) reference images.

At the core of the very deep generator network G, Ledig et al. (2017) employed two convolutional layers with 3×3 kernels and 64 feature maps, followed by batch normalization (BN) layers and Parametric ReLU (PReLU) as the activation function. To distinguish real high-resolution (HR) images from

generated super-resolution (SR) samples, the discriminator network consists of eight convolutional layers. These layers use 3×3 filter kernels, increasing from 64 to 512 kernels in powers of 2, similar to the architecture of the VGG network. The discriminator utilizes Leaky ReLU and avoids max-pooling throughout the network.

2.5.1.1 Real-ESRGAN

To further improve the recovered image quality, Wang et al. (2018) proposed the Enhanced Super-Resolution Generative Adversarial Network (ESRGAN). This model introduces two key modifications to the generator structure: removing all Batch Normalization (BN) layers and replacing the original basic block with the Residual-in-Residual Dense Block (RRDB). The RRDB integrates multi-level residual networks with dense connections. The removal of BN layers has been shown to improve performance and reduce computational complexity, as BN layers can introduce unpleasant artifacts and limit generalization when there is a significant discrepancy between the statistics of training and testing datasets.

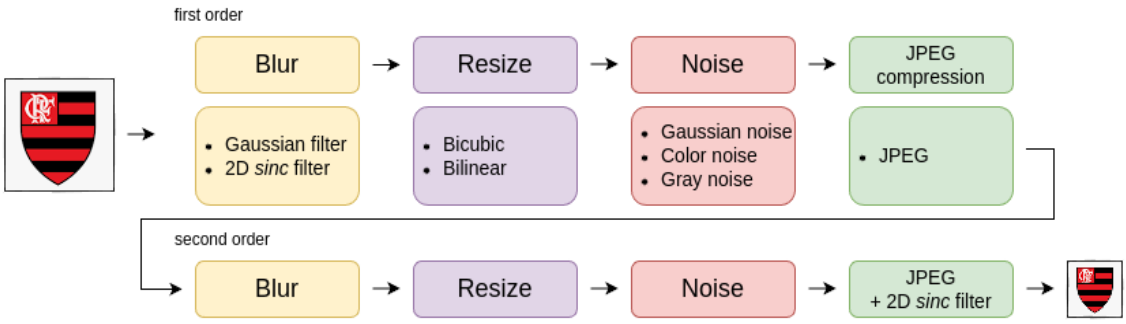

Figure 2.10: Overview of the pure synthetic data generation adopted in Real-ESRGAN. It utilizes a second-order degradation process to model more practical degradations. The *sinc* filter is also used to synthesize common ringing and overshoot artifacts.

Building upon ESRGAN, Wang et al. (2021b) introduced Real-ESRGAN, an extension designed for practical restoration applications and trained with purely synthetic data. Classical degradation model which includes blurring, downsampling, noise addition, and JPEG compression, is widely adopted in explicit modeling methods. Specifically, the ground-truth image y is first convolved with a blur kernel k. Next, a downsampling operation with a scale factor r is applied. The resulting low-resolution image x is obtained by adding noise n. Finally, JPEG compression is applied, as it is commonly used in real-world

images (Wang et al., 2021b). This process is described in Equation 2.8:

$$x = D(y) = [(y \circledast k) + \downarrow_r + n]_{\text{JPEG}} \tag{2.8}$$

where D denotes the degradation process.

However, this straightforward combination of multiple degradations cannot address more complex real-world scenarios, particularly those involving unknown noises and intricate artifacts. Real-world complex degradations often arise from convoluted combinations of various processes, such as camera imaging systems, image editing, and Internet transmission. Real-ESRGAN extends ESRGAN by restoring general real-world low-resolution images through synthesizing training pairs with a more practical degradation process.

Specifically, a high-order degradation modeling process is introduced to more accurately simulate complex real-world degradations. This approach utilizes an n-order model, involving n repeated degradation processes, where each process employs the classical degradation model as illustrated in Figure 2.10. Additionally, a UNet discriminator (Ronneberger et al., 2015), enhanced with spectral normalization to boost its capability and stabilize training dynamics, is introduced to improve upon the VGG-style discriminator used in ESRGAN.

2.5.2 Diffusion Models

Deep generative models have shown remarkable capability in producing high-quality samples across various domains. In the realm of image generation, generative adversarial networks (GANs) can exhibit higher sample quality. However, GANs necessitate careful selection of optimization techniques and architectural choices to stabilize training. Additionally, they often struggle to capture the full data distribution (Song et al., 2020).

In 2015, Sohl-Dickstein et al. (2015) introduced the concept of Diffusion Models (DMs) in their paper "Deep Unsupervised Learning using Nonequilibrium Thermodynamics." The core idea, derived from non-equilibrium statistical physics, involves systematically and gradually destroying the structure within a data distribution through an iterative forward diffusion process. A reverse diffusion process is then learned to restore the structure in the data, resulting in a highly flexible and tractable generative model.

The method employs a Markov chain to gradually transform one distribution into another, a concept borrowed from non-equilibrium statistical physics. This transformation is achieved through a generative Markov chain that converts a simple known distribution (e.g., a Gaussian) into a target data distribution via a diffusion process. The probabilistic model is explicitly defined as the

endpoint of the Markov chain. Since each step in the diffusion process has an analytically evaluable probability, the entire chain can also be evaluated analytically.

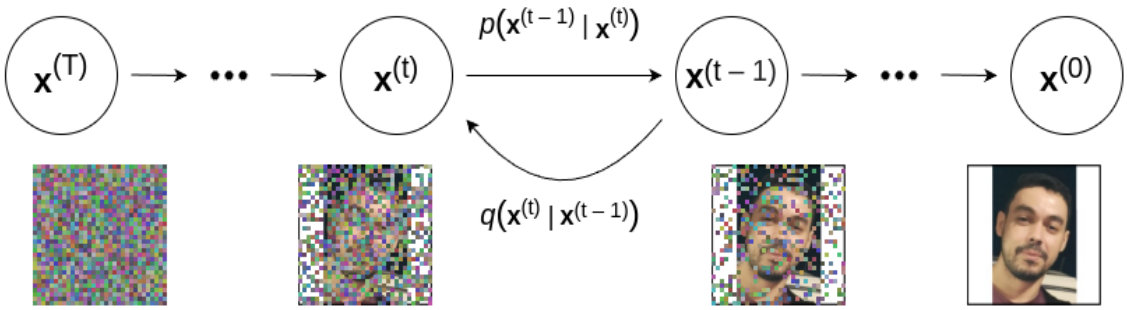

Figure 2.11: Illustration of the diffusion process, showing how data is destroyed by adding noise and the subsequent reverse denoising process.

The method aims to define a forward (or inference) diffusion process which converts any complex data distribution into a simple, tractable distribution, and then learns a finite-time reversal of this diffusion process which defines the generative model distribution. Let $q\left(\mathbf{x}^{(0)}\right)$ be the data distribution. The forward trajectory, starting at the data distribution and performing T steps of diffusion, is defined in Equation 2.9:

$$q\left(\mathbf{x}^{(0\cdots T)}\right) = q\left(\mathbf{x}^{(0)}\right) \prod_{t=1}^{T} q\left(\mathbf{x}^{(t)} \mid \mathbf{x}^{(t-1)}\right) \tag{2.9}$$

where $q\left(\mathbf{x}^{(t)} \mid \mathbf{x}^{(t-1)}\right)$ corresponds to either Gaussian diffusion into a Gaussian distribution with identity covariance, or binomial diffusion into an independent binomial distribution.

The generative distribution will be trained to describe the same trajectory, but in reverse, as shown in Equation 2.10:

$$p\left(\mathbf{x}^{(0\cdots T)}\right) = p\left(\mathbf{x}^{(T)}\right) \prod_{t=1}^{T} p\left(\mathbf{x}^{(t-1)} \mid \mathbf{x}^{(t)}\right) \tag{2.10}$$

where $p\left(\mathbf{x}^{(0)}\right)$ represents the probability that the generative model assigns to the data. The graphical illustration of this process can be seen in Figure 2.11.

Although Diffusion models were initially considered straightforward to define and train, it was not until 2020 that Ho et al. (2020) demonstrated their capability to generate high-quality samples, sometimes surpassing the performance of other generative models. They introduced Denoising Diffusion Probabilistic Models (DDPMs), which are constructed from a hierarchy of denoising autoencoders (Kingma and Welling, 2013).

2.5.2.1 Latent and Stable Diffusion

Although DDPMs achieve high-quality image generation without adversarial training, they are computationally demanding. This is due to the requirement for repeated function evaluations and gradient computations in the high-dimensional space of RGB images. Specifically, the generative process, which approximates the reverse of the forward diffusion process, may involve thousands of steps. Iterating through all these steps to produce a single sample is significantly slower compared to GANs, which require only one pass through a network (Song et al., 2020).

For example, generating 50,000 images of size 32×32 from a DDPM takes approximately 20 hours, whereas the same task with a GAN requires less than a minute on an Nvidia 2080 Ti GPU. This disparity has two main consequences for the research community and users: Firstly, training a DDPM demands substantial computational resources, available only to a small fraction of researchers, and contributes significantly to the carbon footprint (Rombach et al., 2022). Secondly, evaluating a pre-trained model is also time and memory intensive, as the model architecture must sequentially process a large number of steps.

To increase the accessibility of this powerful model class while reducing its significant resource consumption, Rombach et al. (2022) introduced Latent Diffusion Models (LDMs). They achieved this by separating training into two distinct phases: first, they trained an autoencoder to provide a lower-dimensional, yet perceptually equivalent, representational space. Next, they trained the diffusion models in this learned latent space, which reduces computational load and enhances performance, making it feasible to handle high-resolution images or complex data.

This reduced complexity also allows for efficient image generation from the latent space with a single network pass. A notable advantage of this approach is that the universal autoencoding stage needs to be trained only once, making it reusable for multiple diffusion model trainings or for exploring entirely different tasks. This enables efficient exploration of a wide range of diffusion models for various image-to-image and text-to-image applications.

The neural backbone of the model is realized as a time-conditional UNet (Ronneberger et al., 2015). Similar to other types of generative models, diffusion models are in principle capable of modeling conditional distributions of the form $p(z \mid y)$. This can be implemented with a conditional denoising autoencoder $\varepsilon_\theta(z_t, t, y)$, with z_t denoting the latent representation of the data at step t, which paves the way to controlling the synthesis process through inputs y, such as text, semantic maps, or other image-to-image translation tasks. LDMs can be efficiently trained for super-resolution by directly conditioning

low-resolution images, where the low-resolution image y is concatenated with the inputs to the UNet.

By augmenting their underlying UNet backbone with a cross-attention mechanism (Vaswani et al., 2017), which is effective for learning attention-based models of various input modalities, diffusion models (DMs) can be transformed into more flexible conditional image generators. This mechanism is used to preprocess y from various modalities, such as language prompts.

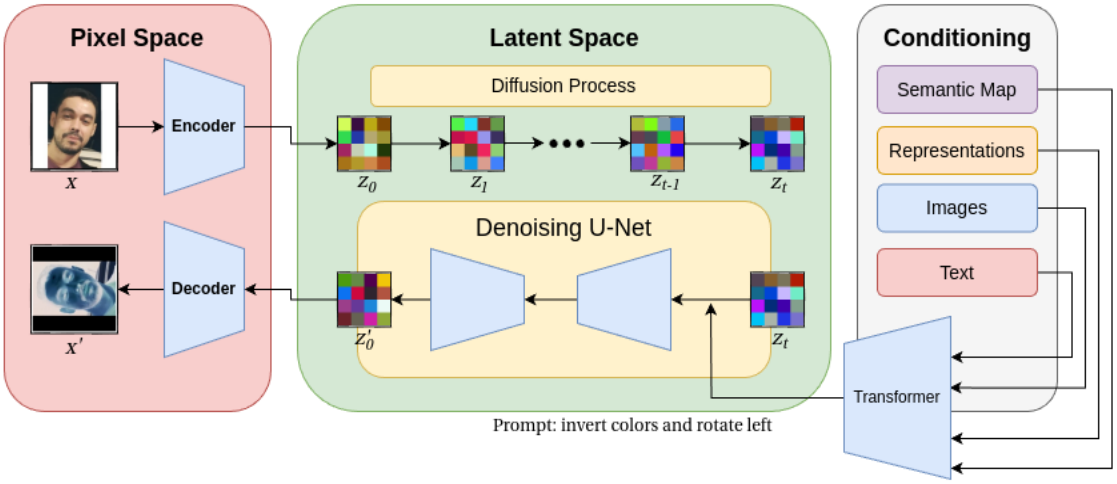

Figure 2.12: Simplified view of Stable Diffusion: the Diffusion Model is trained in the learned latent space, which is smaller but equivalent to the RGB space. This approach reduces computational load and improves performance. A text prompt can be conditioned to the Latent Diffusion Model through a cross-attention mechanism, resulting in a flexible image generator.

With financial support from Stability AI and assistance from LAION, the Latent Diffusion authors were able to train a Latent Diffusion Model on 512×512 images from a subset of the LAION-5B dataset (Schuhmann et al., 2022) and developed Stable Diffusion, a latent text-to-image diffusion model. This model features a 60M UNet and a 123M text encoder. A simplified view of this model can be seen in Figure 2.12.

CHAPTER

3

Domain Adaptation using Transformers for Sugarcane Rows and Gaps Detection

3.1 Introduction

Crop rows detection is a problem that consists of identifying the plantation lines in a given image captured by unmanned aerial vehicles (UAVs) or autonomous terrain vehicles. This procedure is important for crop planning, production estimation, plant counting, harvesting and early correction of planting failures (Soares et al., 2018; Osco et al., 2021). In addition, our approach also detects the gaps present in these rows. These gaps, which can appear during planting and harvesting operations, can reduce the productivity of planting and the profitability of the cultivated area (Rocha et al., 2022).

The Hough transform method (Hough, 1962) is the most commonly used strategy for identifying crop rows (Jiang et al., 2015; Bah et al., 2019; Chen et al., 2021). Recently, other approaches combine the Hough transform with other methods, such as superpixel and CNN (Bah et al., 2019). Although most of these methods do not rely on labeled data, they generally fail to detect curves commonly present in rows of crop images captured by UAVs, thus lacking sufficient accuracy (García-Santillán et al., 2017; Soares et al., 2018; Chen et al., 2021). Curve detection also remains challenging for other recent methods (García-Santillán et al., 2017; Rocha et al., 2022).

The approach we present here reduces sugarcane row and gap detection

Figure 3.1: Example of challenging images for the methods used for crop rows and gaps detection, most using Hough transform based approaches. In (a) and (b) we have curves in the plantations rows, in (c) non-parallel lines and in (d) shadows and large gaps.

to a common semantic segmentation problem to enable the use of state-of-the-art methods capable of addressing challenges such as curves, shadows, and non-parallel lines, as illustrated in Figure 3.1. Bah et al. (2019) employed a similar approach in their work, using segmentation for crop row detection. They developed CRowNet, a modified CNN combined with Hough transform. Due to the time-consuming nature of manually annotating images required for a supervised CNN, they simplified the annotation process, proceeding in a semi-supervised manner.

Compared to (Bah et al., 2019), our experiments have two main differences: (1) we adapted our ground truth using dilation to enable the use of state-of-the-art semantic segmentation architectures for general purposes, without relying on Hough transform to capture specific features of line detection, and (2) to address the expensive annotation process and common lack of annotated data, we evaluated Vision Transformer (Dosovitskiy et al., 2020) and applied unsupervised domain adaptation to generalize our models to similar but different domains without labeled data.

3.2 Methodology

3.2.1 Method

Our proposed method combines a strategy of reducing labeling efforts based on dilations with the application of transformers (Vaswani et al., 2017; Dosovitskiy et al., 2020) and unsupervised domain adaptation for crop row and gap detection. Dilation, a fundamental morphological operation, involves adding pixels at object boundaries. This operation gradually expands the boundaries of foreground pixels, increasing their area size (Chudasama et al., 2015). In our approach, dilation was used to generate approximate segmentation maps

from manually labeled one-pixel-wide lines by specialists. The process used to generate segmentation maps is illustrated in Figure 3.2.

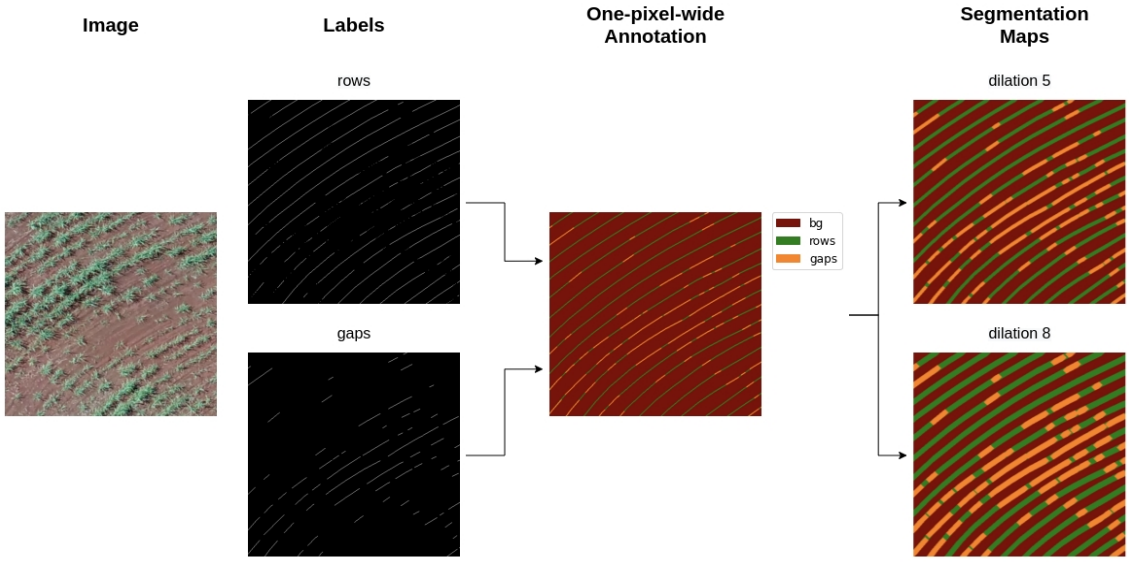

Figure 3.2: Semi-automatic process of generating segmentation maps using dilation in one-pixel-wide annotations manually labeled by specialists.

The use of these segmentation maps enabled the detection of crop rows and gaps using standard semantic segmentation networks, without relying on the Hough transform. However, while fully supervised architectures have achieved significant success in semantic segmentation, they necessitate a large number of fully annotated images for the training set, a process that is time-consuming and costly (Vezhnevets et al., 2011; Huang et al., 2018). To mitigate this drawback, we evaluated three different approaches in our experiments: supervised semantic segmentation, source model only, and unsupervised domain adaptation.

In supervised semantic segmentation, source images and labels are used to train the model, which is evaluated using source images from a different subset, the test set, which contains images not used in training. In source model only (Src-Only) (Liang et al., 2020), the same supervised model is evaluated using images from different but related target domains. While this strategy has the advantage of not relying on annotated images for the target dataset, a decrease in model performance is expected due to domain shift. For both evaluations, we used the recent transformer-based model, SegFormer (Xie et al., 2021), and compared its results with robust ConvNet models.

Unlike source model only, in unsupervised domain adaptation, target images without annotations are used jointly with source images and their annotations in the training step to generalize model learning to target images, reducing source overfitting. In our experiments, we utilized DAFormer (Hoyer

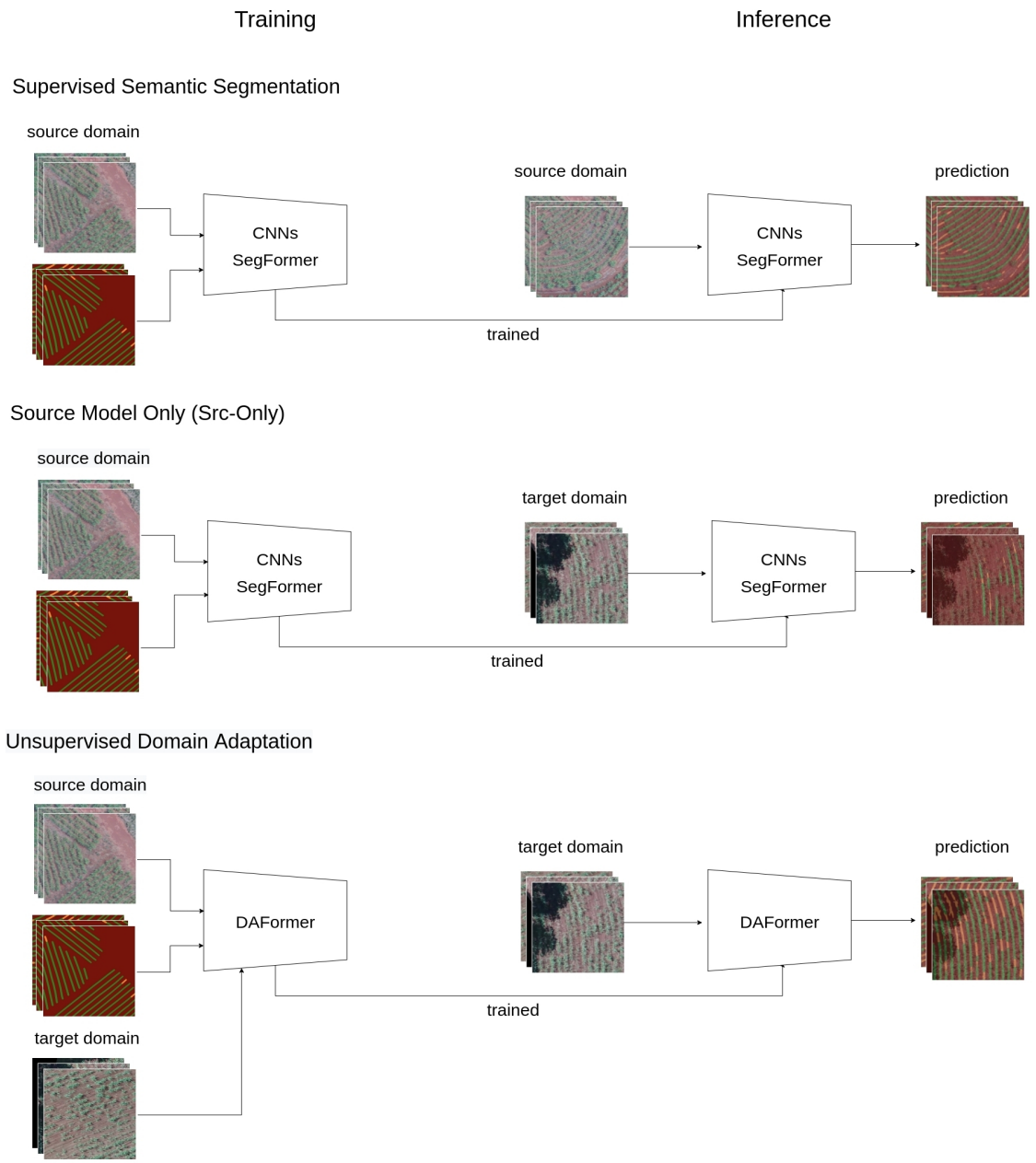

Figure 3.3: In Supervised Semantic Segmentation, source images and labels are used to train the model which is evaluated using source images from test set. In Source Model Only, the same supervised model is evaluated using images from different but related target domain. In Unsupervised Domain Adaptation, target images without annotations are also used in the training step.

et al., 2021), an unsupervised domain adaptation architecture based on SegFormer. The pipeline of our experiments can be seen in Figure 3.3.

3.2.2 Dataset

The images were acquired on four different sugarcane farms with the same camera using an UAV. The orthoimages were generated using Agisoft Metashape software, which is based on structure-from-motion (SfM) and MultiView Stereo

(MVS) computer vision techniques. A ground sample distance (GSD) of 5 centimeters was considered, which enabled the identification of the sugarcane plantation rows and gaps greater than 50 centimeters.

Farm	Train	Validation	Test	Total
Farm 1 (F1)	973	162	486	1621
Farm 2 (F2)	1250	208	626	2084
Farm 3 (F3)	592	99	296	987
Farm 4 (F4)	674	112	338	1124

Table 3.1: Total of images of train (60%), validation (20%) and test (30%) sets for each farm.

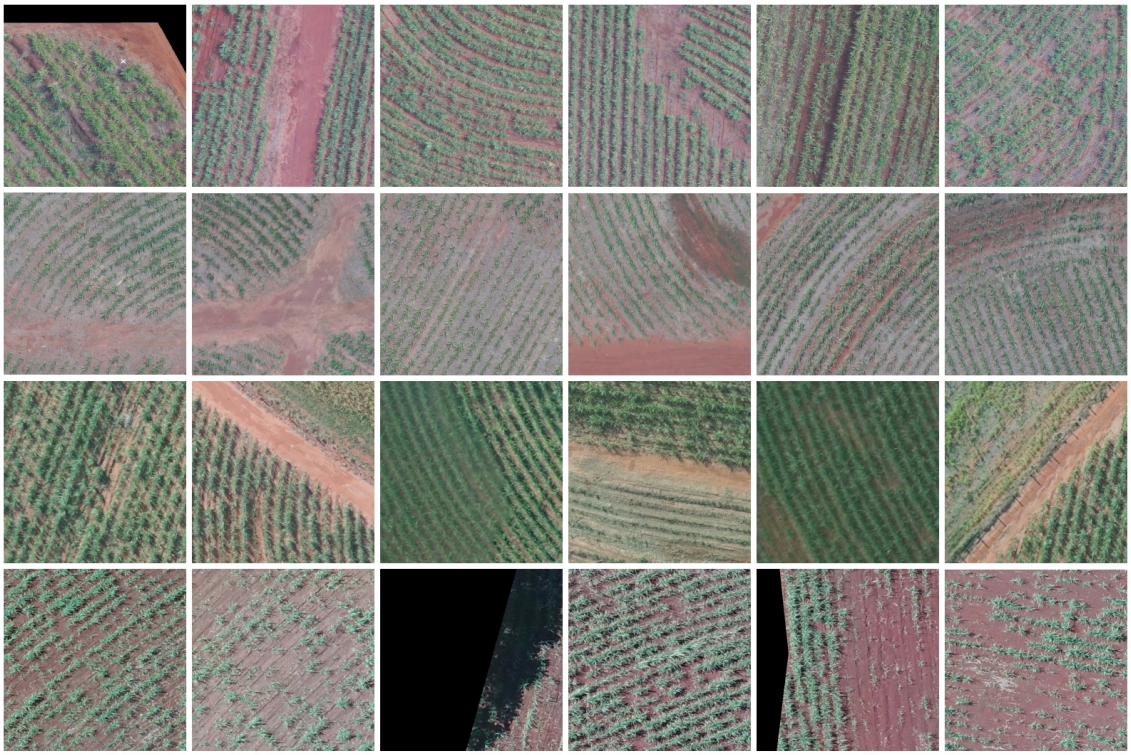

Figure 3.4: From top to bottom, in the first row, images from Farm 1 (F1), in the second row, images from Farm 2 (F2), in the third row, images from Farm 3 (F3), and in the last row images from Farm 4 (F4).

The images of each dataset were divided into training, validation, and test sets, with the number of images for each set detailed in Table 3.1. In general, all images used in the experiment had many pixels annotated as rows. However, most images had few or no pixels annotated as gaps. In Table 3.2, we show the number of images, for each farm, containing at least 10, 100, 1000 and 2000 pixels annotated as rows and gaps. We considered the pixels of the

original annotations, without using dilation. These data helped us to discuss the results of this research.

	+10	+100	+1000	+2000
Rows				
Farm 1 (F1)	1621 (1.00)	1620 (0.99)	1558 (0.96)	1533 (0.95)
Farm 2 (F2)	2084 (1.00)	2073 (0.99)	1998 (0.96)	1941 (0.93)
Farm 3 (F3)	987 (1.00)	986 (0.99)	956 (0.97)	931 (0.94)
Farm 4 (F4)	1124 (1.00)	1124 (1.00)	1112 (0.99)	1074 (0.96)
Gaps				
Farm 1 (F1)	1436 (0.89)	1088 (0.67)	171 (0.11)	40 (0.02)
Farm 2 (F2)	1421 (0.68)	891 (0.43)	223 (0.11)	76 (0.04)
Farm 3 (F3)	950 (0.96)	799 (0.81)	269 (0.27)	48 (0.05)
Farm 4 (F4)	1124 (1.00)	1122 (1.00)	874 (0.78)	362 (0.32)

Table 3.2: For each farm, the number of images containing at least 10, 100, 1000 and 2000 pixels annotated as rows and gaps. In parentheses, the proportion in relation to all images in the training, validation and test sets.

3.2.1.1 Pixel Labeling (Dilation)

Each RGB image from our four datasets was manually annotated by specialists using two additional grayscale images with the same dimensions, containing one-pixel-wide lines to indicate the presence of rows or gaps. An example can be seen in the first row of Figure 3.5. To generate segmentation maps similar to those used in datasets such as Cityscapes and GTA, we applied multidimensional binary dilation to the annotated images.

Our approximated ground truth is a segmentation map containing specific RGB values and a color palette to represent background, rows, and gap annotations. In the second row of Figure 3.5, examples of ground truth using 3, 5, and 8 iterations of dilation are shown. In this work, we used 5 iterations as the default, as it provides a close approximation to the width of the crop rows for most images in the datasets. Henceforth, we will simply refer to this as dilation 5. However, it should be noted that this value can vary significantly within the same dataset if the images are captured at different flight heights.

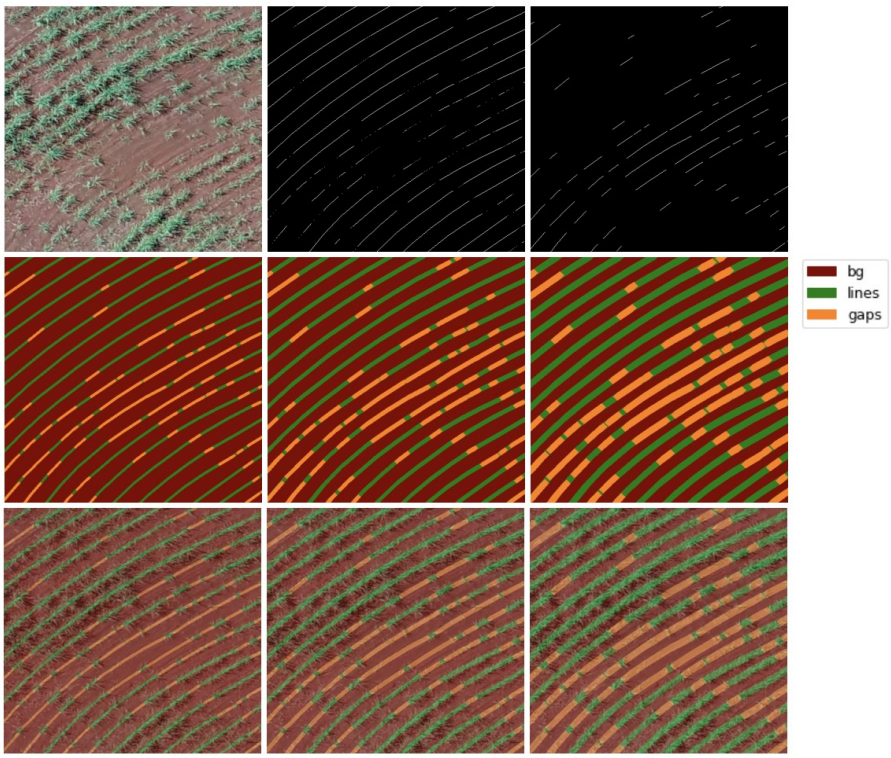

Figure 3.5: From left to right, in the first row, the original image and one-pixel-wide line annotations of rows and gaps. In the second row, the ground truth generated from the three above images using 3, 5 and 8 iterations of dilation. In the third row, the transparent ground truths merged into original image for better visualization.

3.2.3 Evaluation Metrics

To assess and compare the networks evaluated in the experiments, we utilized metrics commonly applied in the literature: precision, recall, F1-score, and intersection over union (IoU) at the pixel level. Precision indicates how many pixels predicted for a given class actually belong to that class. Recall indicates the ability of the predictions to recover correct information, i.e., how many pixels belonging to a specific class were correctly identified. F1-score represents the balance between precision and recall, serving as a harmonic average of these two metrics.

These four metrics can be calculated using the following equations:

$$Precision = \frac{TP}{TP+FP} \tag{3.1}$$

$$Recall = \frac{TP}{TP+FN} \tag{3.2}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{3.3}$$

$$IoU = \frac{P \cap GT}{P \cup GT} \tag{3.4}$$

where TP corresponds to true positives, FP corresponds to false positives, FN corresponds to false negatives, P corresponds to prediction, and GT corresponds to Ground Truth. We evaluated all assessments using F1-score, since F1 is a robust metric that calculates the trade-off between recall and precision, and used IoU as an additional metric in our supervised semantic segmentation results.

3.2.4 Experimental Setup

We ran our experiments on a Intel(R) Core(TM) i7-5820K CPU @ 3.30GHz, with 32 GB of RAM and GPU Nvidia GeForce GTX TITAN X with 12 GB GDDR5 memory and 3072 CUDA Cores. We performed the tests using four different datasets, Farm 1 (F1), Farm 2 (F2), Farm 3 (F3), and Farm 4 (F4).

For our supervised semantic segmentation tests, we used the available architectures in the *MMSegmentation* (Contributors, 2020), which that can be accessed at https://github.com/open-mmlab/mmsegmentation. We used the following supported methods: PSPNet, DeepLabV3+ and SegFormer.

To carry out our training using these methods we used the base config files available through *MMSegmentation*. For PSPNet, we used the Cityscapes config for backbone ResNet-50, crop size 512×1024 and learning rate schedule 40,000. For the DeepLabV3+ method, we utilized the Cityscapes config with ResNet-101 as the backbone, the same crop size of 512×1024, and a learning rate schedule of 40,000. Lastly, for SegFormer, we used the Cityscapes config with MiT-B5 as the backbone, a crop size of 1024×1024, and a learning rate schedule of 160,000.

We selected these base config files from the available options in *MMSegmentation* after evaluating the trade-off between mIoU performance and time / memory efficiency during training on the Cityscapes dataset, prioritizing mIoU performance. However, we adjusted the image scale to 512×512, the number of classes in the decode/auxiliary head to 3, and the crop size to 256×256 to suit our dataset, as base config files with a crop size of 256×256 were not available for these models in *MMSegmentation*.

For all architectures, we initialized the network weights using the models

Table 3.3: Hyperparameter values used in training for each network.

Parameter	**PSPNet**	**DeepLabV3+**	**SegFormer**	**DAFormer**
backbone	ResNet-50	ResNet-101	MiT-B5	MiT-B5
optimizer	SGD	SGD	AdamW	AdamW
learning_rate	0.01	0.01	6e-05	6e-05
momentum	0.9	0.9	-	-
betas	-	-	(0.9, 0.999)	(0.9, 0.999)
weight_decay	0.0005	0.0005	0.01	0.01
lr_config	poly	poly	poly	poly
power	0.9	0.9	1.0	1.0
min_lr	0.0001	0.0001	0.0	0.0
img_scale	(512, 512)	(512, 512)	(512, 512)	(512, 512)
crop_size	(256, 256)	(256, 256)	(256, 256)	(256, 256)
samples_per_gpu	8	8	8	4
workers_per_gpu	8	8	8	4
max_iters	16000	16000	16000	32000

pretrained on the Cityscapes dataset available in *MMSegmentation* for each config. We conducted our training using 16,000 iterations and a batch size of 8 source images. Apart from these adjustments, we retained all other hyperparameters such as optimizers and initial learning rate at their default values as specified in the base config files. For each dataset, we trained all three networks using the labeled ground truths generated with dilation 5 and 8, resulting in a total of $4 \times 3 \times 2$ different models for evaluation.

In our unsupervised domain adaptation experiments, we used the DAFormer code available at https://github.com/lhoyer/DAFormer. We used the config file available here as the base config for our training. Additionally, we initialized the MiT-B5 weights using a pre-trained file provided by the authors, found here. We performed our training using 32,000 iterations, using a batch of 4 images for source and 4 images for target. We doubled the number of iterations, compared to the supervised tests, as we needed to divide the batch size of the source images by 2, to share it with the target images due to GPU memory constraints.

Similar to the training of the semantic segmentation models, with the exception of these modifications, we kept all other hyperparameters with the default values defined in the base config files. We trained all the combinations of different source and targets farms using only dilation 5, enabling and disabling RCS, for a total of $4 \times 3 \times 2$ different models. More detailed information about the hyperparameters can be found in Table 3.3.

3.3 Supervised Semantic Segmentation

	F1 → F1	**F2 → F2**	**F3 → F3**	**F4 → F4**
PSPNet (ResNet-50)				
Background	93.82 (88.37)	93.93 (88.56)	93.21 (87.28)	94.15 (88.95)
Rows	84.48 (73.14)	85.80 (75.13)	81.90 (69.35)	81.10 (62.21)
Gaps	52.35 (35.45)	54.19 (37.17)	65.75 (48.97)	75.98 (61.26)
Average	**76.89 (65.65)**	**77.97 (66.95)**	80.28 (68.53)	83.75 (72.81)
DeepLabV3+ (ResNet-101)				
Background	92.91 (86.76)	93.64 (88.05)	91.26 (83.93)	93.96 (88.60)
Rows	80.67 (67.60)	84.53 (73.20)	75.15 (60.20)	80.28 (67.05)
Gaps	44.08 (28.27)	49.66 (33.03)	55.70 (38.60)	76.06 (61.37)
Average	72.55 (60.88)	75.94 (64.76)	74.04 (60.91)	83.43 (72.34)
SegFormer (MiT-B5)				
Background	93.96 (88.60)	93.88 (88.47)	93.54 (87.86)	94.17 (88.98)
Rows	84.82 (73.64)	85.38 (74.49)	82.57 (70.32)	81.03 (68.11)
Gaps	50.82 (34.06)	52.79 (35.86)	68.27 (51.82)	76.58 (62.05)
Average	76.53 (65.43)	77.35 (66.27)	**81.46 (70.00)**	**83.93 (73.05)**

Table 3.4: F1-score and IoU, in parentheses, of supervised training for each dataset. The models were evaluated using ground truth generated with dilation 5. In bold, the best average result for each dataset.

In our first experiment, we trained semantic segmentation architectures using segmentation maps obtained using dilation 5 and compared the performance of ConvNets with the recent transformer-based model, SegFormer. All models were trained with labeled data and evaluated using images from the same dataset, $F_S \rightarrow F_S$, with $1 \leq S \leq 4$. In Table 3.4, we show the F1-score, at

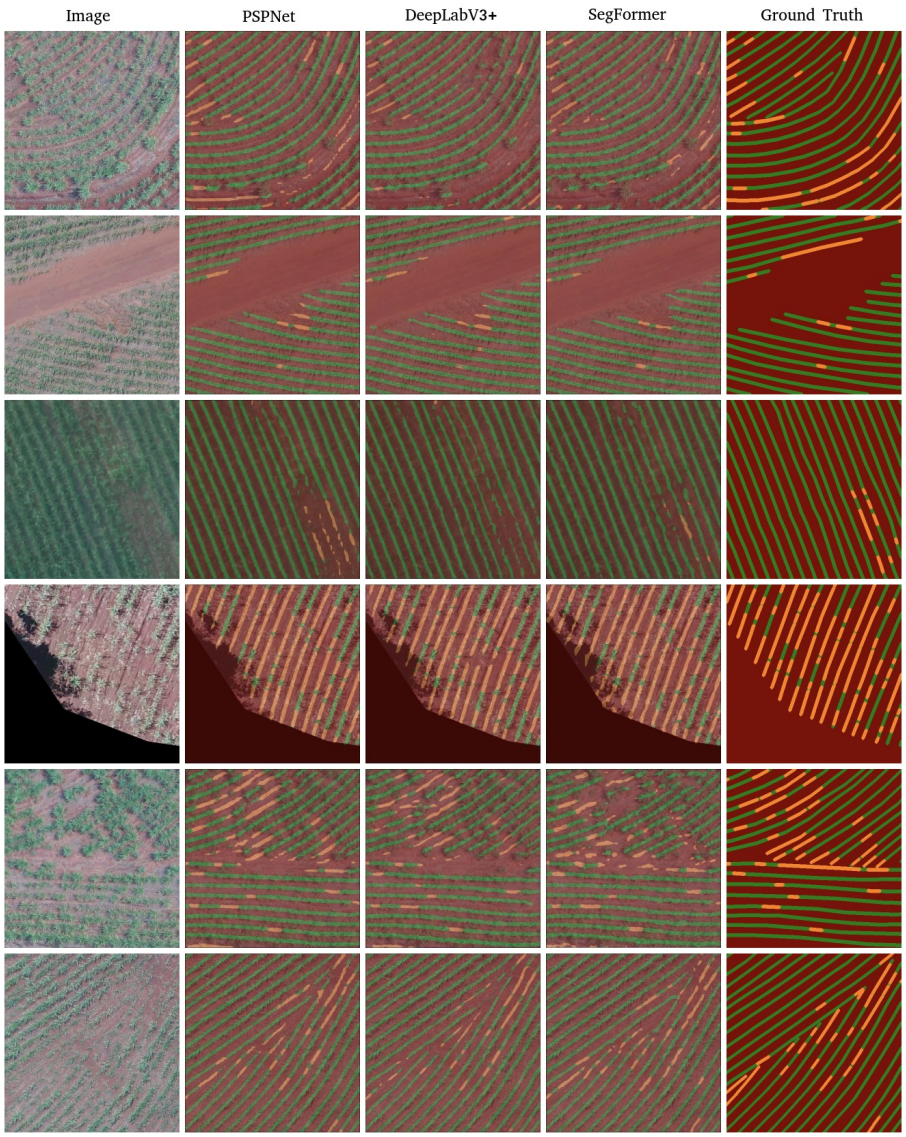

Figure 3.6: From left to right, the original image, supervised predictions made by PSPNet, DeepLabV3+ and Segformer, and the ground truth. It's possible to notice that the networks achieved impressive results even when dealing with challenging conditions such as curves, shadows and non-parallel lines.

the pixel level, for the three architectures. Generally, PSPNet and SegFormer showed very close results, with DeepLabV3+ performing slightly lower despite using ResNet-101 as the backbone compared to ResNet-50 used by PSPNet. However, all architectures achieved an average F1-score above 70 for all tested farms. It is also noteworthy that there is a direct correspondence between the F1-score and IoU values.

Analyzing the results of supervised segmentation by categories, the gaps class exhibited poorer performance compared to the other classes. This phenomenon can be attributed to the fact that pixels belonging to the gaps class share identical visual characteristics, such as color and texture, with most pixels in the background class. Therefore, feature learning for gaps pixels

must be performed using spatial context information. Nevertheless, due to the imbalance in pixel distribution across classes in the images, the network tends to exhibit a bias towards the background class, which contains significantly more pixels than the gaps class. Consequently, we achieved better gap detection results on farms that had more images with a higher number of pixels annotated as gaps, as evidenced in Table 3.2, where farm $F4$ achieved the best results. We analyzed this problem more effectively when performing domain adaptation.

In addition to the quantitative results shown in the table, we also analyzed the visual qualitative results of the segmentation. We selected images containing several challenges typical in current approaches to this problem, such as curves, complex variations in line directions, shadows, and poor-quality images, to assess the robustness of our segmentation. As depicted in Figure 3.6, all networks, particularly PSPNet, successfully detected rows and most gaps, with predictions very similar to the approximated ground truths used in the training.

3.3.1 Dilation Impact

Since we calculated recall and precision at the pixel-level rather than the line-level, as in most similar works, the number of dilation iterations chosen can impact the F1-score obtained. To analyze this impact, we also evaluated the same datasets using segmentation maps generated using dilation 8, and the results can be seen in Table 3.5.

Using dilation 8, the average F1-score and IoU showed a significant increase in almost all evaluations conducted. This improvement was particularly notable for gaps, as the larger dilation helped mitigate the class imbalance between the background and gaps. However, it cannot be assumed that further increasing dilation indefinitely will consistently yield better results. The recommended approach is to set the dilation close to the actual width of the rows and gaps for optimal performance.

We would also like to point out that we can skeletonize the inferred images obtained in Figure 3.6 to represent rows and gaps as one-pixel-wide lines, similar to the original labels. Skeletonization is a process which represents a pattern by a collection of thin (or nearly thin) arcs and curves and was used similarly by Bah et al. (2019). With these skeletons, we could apply metrics defined in (Mnih and Hinton, 2012; Wei et al., 2021), using a buffer of ρ pixels, to calculate the F1-score at the line level and potentially achieve higher values than those obtained at the pixel level. As we dealt with multiples scenarios in this research, we decided not to pursue this approach to avoid increasing the complexity of the study.

	F1 → F1	**F2 → F2**	**F3 → F3**	**F4 → F4**
PSPNet (ResNet-50)				
Background	91.84 (84.91)	91.43 (84.22)	91.44 (84.23)	91.80 (84.84)
Rows	89.24 (80.56)	89.53 (81.04)	87.22 (77.33)	84.65 (73.38)
Gaps	60.85 (43.73)	62.14 (45.07)	71.78 (55.98)	81.16 (68.29)
Average	**80.64 (69.74)**	**81.03 (70.11)**	83.48 (72.52)	**85.87 (75.50)**
DeepLabV3+ (ResNet-101)				
Background	91.64 (84.58)	90.80 (83.15)	90.08 (81.96)	91.39 (84.14)
Rows	89.14 (80.41)	88.60 (79.54)	85.91 (75.30)	84.15 (72.64)
Gaps	59.45 (42.30)	57.70 (40.54)	66.14 (49.41)	80.66 (67.59)
Average	80.08 (69.10)	79.03 (67.74)	80.71 (68.89)	85.40 (74.79)
SegFormer (MiT-B5)				
Background	91.83 (84.90)	91.35 (84.08)	91.37 (84.11)	91.65 (84.58)
Rows	89.23 (80.55)	89.42 (80.87)	87.19 (77.29)	84.59 (73.30)
Gaps	60.51 (43.38)	61.18 (44.07)	73.00 (57.48)	80.89 (67.91)
Average	80.52 (69.61)	80.65 (69.67)	**83.85 (72.96)**	85.71 (75.26)

Table 3.5: F1-score and IoU, in parentheses, of supervised training for each dataset. The models were evaluated using ground truth generated with dilation 8. In bold, the best average result for each dataset.

3.3.2 Source Model Only (Src-Only) Performance (without UDA)

Despite the robust performance achieved by fully supervised segmentation, it still relies on annotated data, posing a significant obstacle to the practical application of these techniques in real-world agricultural challenges. Manual image annotation for this specific problem is highly time-consuming, as it requires not only the ability to draw continuous lines per crop row or gap but also to repeat this process for each image (Bah et al., 2019). In an optimistic scenario, knowledge learned from specific farms with labeled data could be generalized to other farms, overcoming real-world complexities such as geographic domain shifts and data noise (Beery et al., 2022).

In our source model only (Src-Only) experiments, we evaluated the ability of our supervised models, trained using data only from source farm F_S, to segment images of other target farms F_T ($F_S \rightarrow F_T$). The results can be seen at Table 3.6. We can see in the results that although PSPNet achieved very similar re-

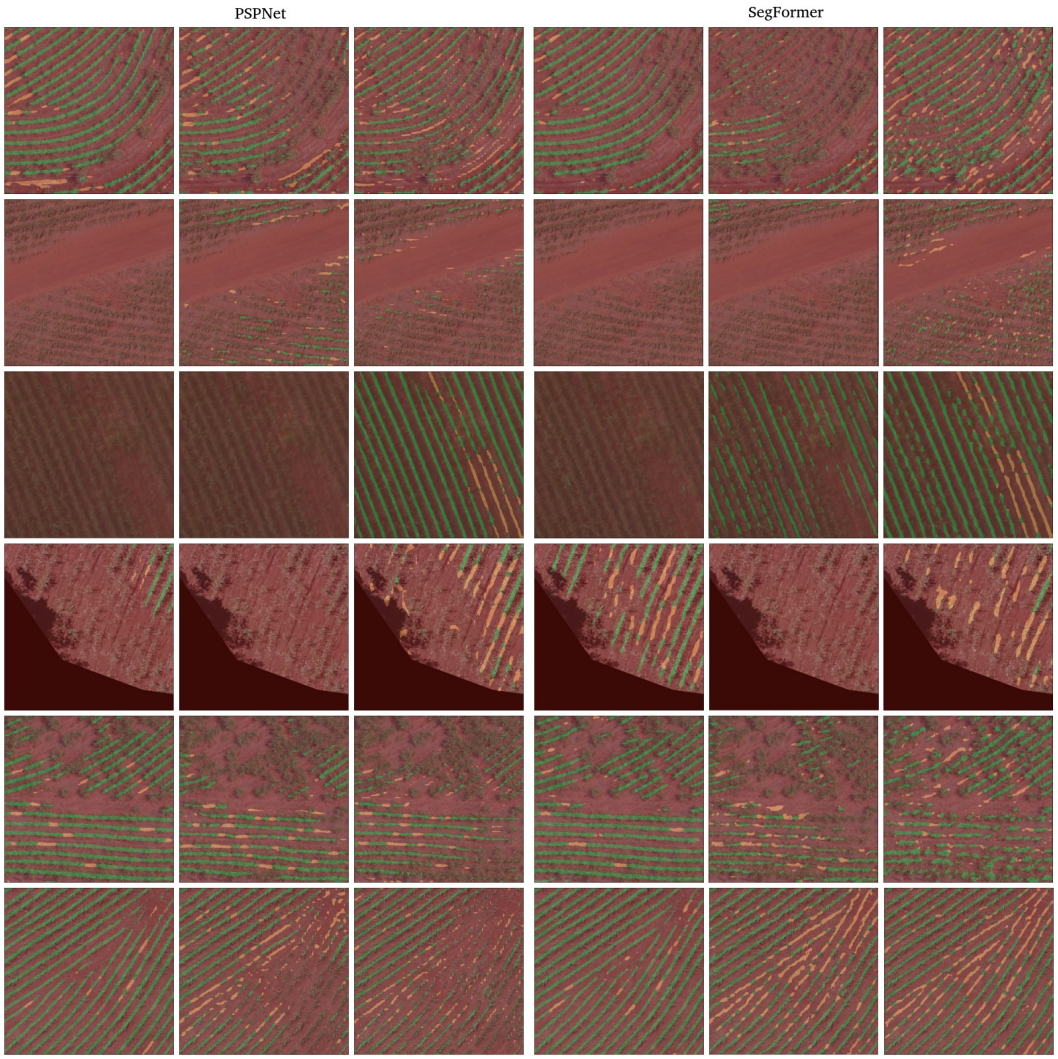

Figure 3.7: Src-only visual results using the same images analysed in Figure 3.6. In these inferences, each target image was evaluated using models trained only with images from another source farm, with each column corresponding to one of the source farms F_S, with $1 \leq S \leq 4$ and $S \neq T$. On the left, the results achieved by PSPNet and by SegFormer on the right

sults to SegFormer when evaluating images from farms of same domain used in training ($F_S \rightarrow F_S$), when evaluating images from different domains ($F_S \rightarrow F_T$) SegFormer presents a noticeable advantage over the ConvNet models.

The robustness of SegFormer to common corruptions and perturbations was reported by Xie et al. (2021) in their original work and represents an important resource for dealing with domain shift. The transformer-based model performs even better when considering the results of class gaps. Our work also corroborates the results achieved by SegFormer when evaluating Src-Only in Cityscapes dataset by Hoyer et al. (2021).

In Figure 3.7, we can observe the visual results of the same images previously shown, inferred with PSPNet and SegFormer using src-only training. While both architectures fail to recognize rows and gaps in some images, Seg-

	F2→F1	F3→F1	F4→F1	F1→F2	F3→F2	F4→F2	F1→F3	F2→F3	F4→F3	F1→F4	F2→F4	F3→F4
PSPNet (ResNet-50)												
Background	92.4	88.1	87.6	88.1	83.0	82.6	85.2	85.4	89.4	91.6	87.8	87.6
Rows	80.3	62.0	54.6	58.9	13.4	8.9	41.8	45.2	69.8	72.2	44.3	58.5
Gaps	35.9	17.2	23.3	20.9	6.6	7.5	1.1	6.8	36.7	28.7	32.9	75.9
Average	69.5	55.8	55.2	56.0	34.3	33.0	42.7	45.8	65.3	64.2	55.0	60.0
DeepLabV3+ (ResNet-101)												
Background	92.1	86.9	87.4	87.8	83.9	82.7	84.3	86.3	89.7	90.3	89.0	88.7
Rows	79.2	51.8	51.7	57.1	24.2	5.2	30.5	49.1	68.1	66.8	55.7	60.1
Gaps	31.7	17.2	23.4	15.9	5.2	5.0	1.4	4.8	37.9	22.1	37.1	37.5
Average	67.7	52.0	54.1	53.6	37.8	31.0	38.7	46.7	65.2	59.7	**60.6**	62.1
SegFormer (MiT-B5)												
Background	92.5	90.3	90.5	88.7	86.0	84.1	87.5	88.2	90.5	92.3	87.8	90.0
Rows	80.8	70.8	72.1	62.7	41.7	41.0	58.1	66.6	74.5	73.0	44.1	67.7
Gaps	36.5	21.0	30.4	25.1	17.2	19.8	7.4	23.1	44.0	40.5	31.0	52.4
Average	**70.0**	**60.7**	**64.4**	**58.8**	**48.3**	**48.3**	**51.0**	**59.3**	**69.7**	**68.6**	54.3	**70.1**

Table 3.6: F1-score of the src-only evaluation for each pair of datasets $F_S \rightarrow F_T$. The models were evaluated using ground truth generated with dilation 5. In bold, the best average result for each pair of datasets.

Former shows a significant improvement in recognition compared to PSPNet, with 6 out of 15 images exhibiting better performance.

While improving results in the source domain caused the ConvNet models to specialize in specific source information, thereby decreasing their performance on different but related domains, SegFormer improved its performance in the source domain with less impact on the results of target domains. This enhancement in knowledge generalization underscores transformer-based models as a highly promising choice for future research, not only in agriculture but also in computer vision.

3.3.3 Generalization by Epochs in Supervised Learning

In our source model only experiments in Section 3.3.2, it was observed that although the ConvNet models achieved similar performance when segmenting images from the same domain, SegFormer achieved noticeably better perfor-

mance when segmenting images from a similar but different domain. To better analyze this behavior, we conducted experiments comparing the performance of PSPNet and SegFormer in the same and different domains, trained over different epochs.

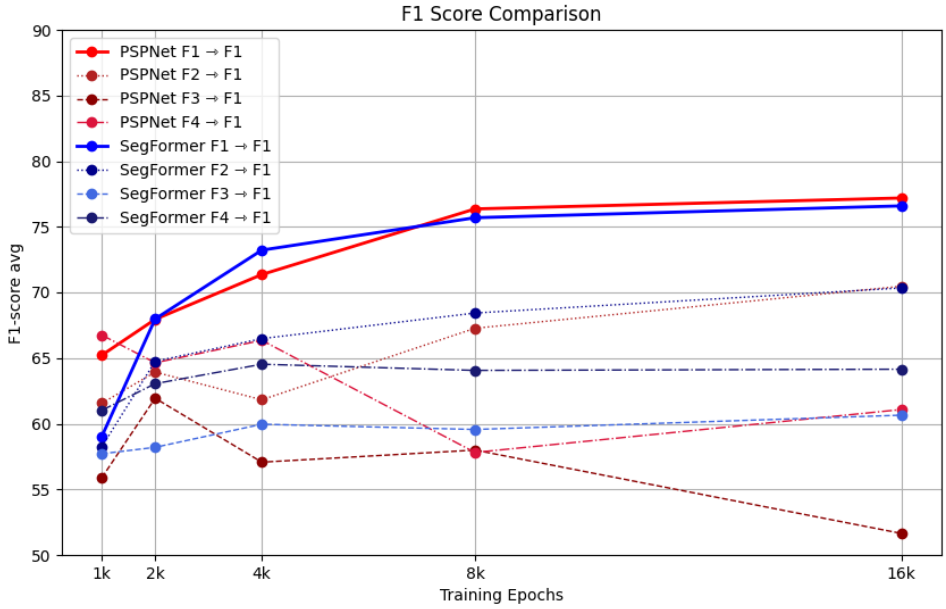

Figure 3.8: F1-score comparison of PSPNet and SegFormer models trained for different numbers of epochs evaluated on the target farm $F1$.

In Figures 3.8, 3.9, 3.10, and 3.11, we present the results of our evaluation for each target farm, considering the average F1-score across all classes. It is noticeable that, in general, both PSPNet and SegFormer show a significant performance improvement with an increased number of epochs when segmenting images from the same domain. All farms achieved their best supervised segmentation performance with 16k epochs, as indicated by the large blue and red lines in the graphs. However, the scenario differs when performing segmentation on images from different farms.

In this scenario, both networks generally exhibited performance degradation when trained with more epochs. The red dashed lines, representing PSPNet, show a recurring significant drop in performance between epochs 8k and 16k, as seen in $F3 \rightarrow F1$(3.8), $F3 \rightarrow F2$(3.9), $F1 \rightarrow F3$(3.10), and $F1 \rightarrow F4$(3.11). It is also noticeable, from analyzing the blue dashed lines, that SegFormer achieved better results when classifying different domains because it can better control specialization after 2k epochs.

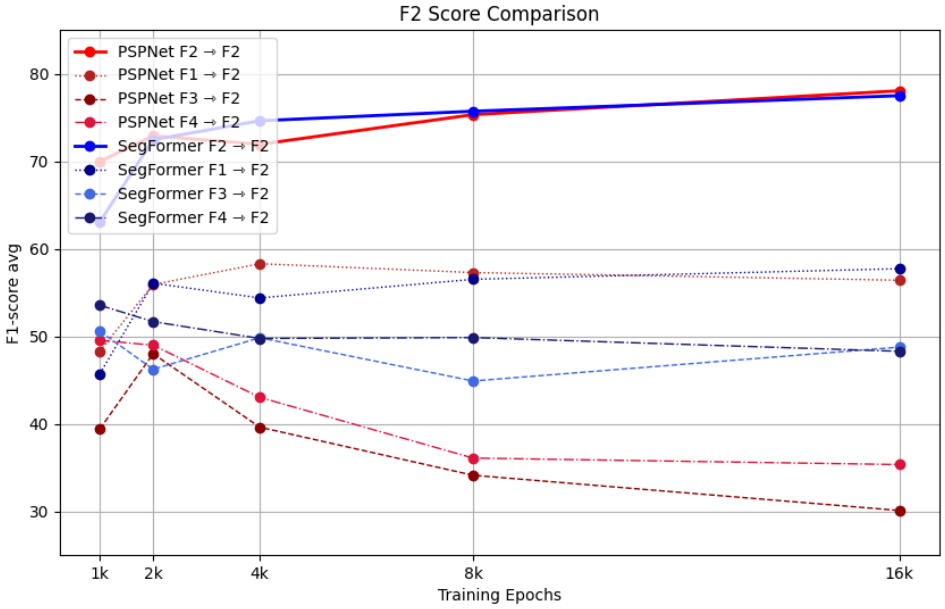

Figure 3.9: F1-score comparison of PSPNet and SegFormer models trained for different numbers of epochs evaluated on the target farm $F2$.

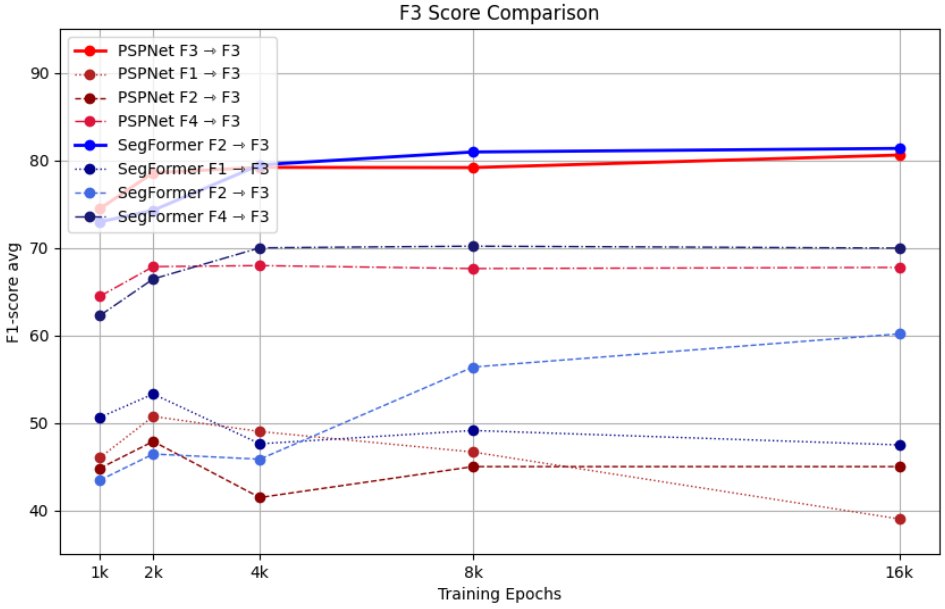

Figure 3.10: F1-score comparison of PSPNet and SegFormer models trained for different numbers of epochs evaluated on the target farm $F3$.

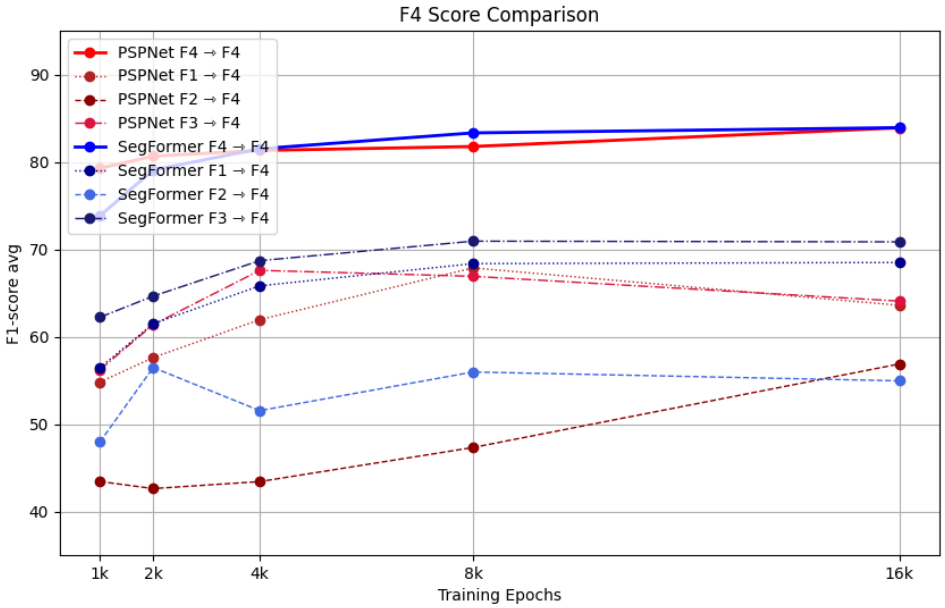

Figure 3.11: F1-score comparison of PSPNet and SegFormer models trained for different numbers of epochs evaluated on the target farm $F4$.

As we can see in this experiment, even though several approaches have been developed to mitigate overfitting, we still have to consider a trade-off between specialization and generalization when deciding the number of epochs to train a network. However, since our general focus is to achieve the best possible results on known data, this will inevitably lead to an increase in specialization at the expense of generalization. Despite the fact that SegFormer presents better robustness to this specialization, these experiments demonstrate the importance of unsupervised domain adaptation to improve generalization.

3.4 Unsupervised Domain Adaptation

3.4.1 DAFormer

Although SegFormer outperformed CNN architectures in the src-only evaluation, we expect to achieve better results by using UDA approaches in conjunction with SegFormer compared to the naive training used in src-only. For the unsupervised domain adaptation experiments, we tested the same pairs of datasets ($F_S \rightarrow F_T$) using the SegFormer-based UDA architecture, DAFormer, and analyzed the impact of rare class sampling (RCS) on model performance. The results of these experiments are shown in Table 3.7.

In general, there was a significant increase in the F1-score for rows and gaps when comparing with the naive training used in src-only. DAFormer

	F2→F1	F3→F1	F4→F1	F1→F2	F3→F2	F4→F2	F1→F3	F2→F3	F4→F3	F1→F4	F2→F4	F3→F4
DAFormer (SegFormer MiT-B5)												
Background	91.1	86.2	85.3	89.8	87.3	81.2	87.7	89.9	90.5	91.4	92.0	90.0
Rows	79.5	71.5	72.1	76.2	74.7	67.5	71.0	75.8	75.8	68.9	74.0	72.6
Gaps	46.0	28.1	25.6	29.6	20.8	13.2	12.6	47.5	45.2	43.2	62.6	66.4
Average	72.2	61.9	61.3	**65.2**	60.9	54.0	**57.1**	**71.1**	**70.5**	67.9	**76.2**	**76.3**
DAFormer (SegFormer MiT-B5) w/o RCS												
Background	92.0	89.6	89.1	89.4	89.8	85.0	81.9	86.7	89.4	90.9	86.1	90.7
Rows	80.7	75.0	75.7	78.0	76.2	70.8	65.0	69.9	75.2	66.7	59.1	71.9
Gaps	45.1	32.6	32.3	27.5	30.5	15.1	7.7	9.4	41.8	31.8	27.0	63.9
Average	**72.6**	**65.7**	**65.7**	65.0	**65.5**	**57.0**	51.5	55.3	68.8	63.1	57.4	75.5
SegFormer (MiT-B5)												
Background	92.5	90.3	90.5	88.7	86.0	84.1	87.5	88.2	90.5	92.3	87.8	90.0
Rows	80.8	70.8	72.1	62.7	41.7	41.0	58.1	66.6	74.5	73.0	44.1	67.7
Gaps	36.5	21.0	30.4	25.1	17.2	19.8	7.4	23.1	44.0	40.5	31.0	52.4
Average	70.0	60.7	64.4	58.8	48.3	48.3	51.0	59.3	69.7	**68.6**	54.3	70.1

Table 3.7: F1-score of UDA evaluation for each pair of datasets $F_S \rightarrow F_T$, compared to src-only evaluation of SegFormer. The models were evaluated using ground truth generated with dilation 5. In bold, the best average result for each pair of datasets.

without RCS performed better when adapting knowledge to target farms $F1$ and $F2$, while the original DAFormer performed better for target farms $F3$ and $F4$. In Figure 3.12, the benefits of using unsupervised domain adaptation compared to the source only model are visible. With the exception of one image, all other images showed rows and gaps detected by DAFormer.

In Table 3.8, we present the evolution of the average F1-score using different methods, compared to the oracle (Yang et al., 2021), which represents the fully supervised model PSPNet trained on the target domain ($F_T \rightarrow F_T$). There are two particular cases to analyze in this data.

When the source and target domains are very similar, the results achieved without UDA are already very close to the oracle, as seen in $F_2 \rightarrow F_1$. In this case, although the UDA improvements were minor, the relative performance with respect to the oracle exceeded 94%. In the second case, where the source and target domains are less related, such as $F_1 \rightarrow F_3$, the original DAFormer achieved a lower F1-score initially. However, it increased the average F1-score

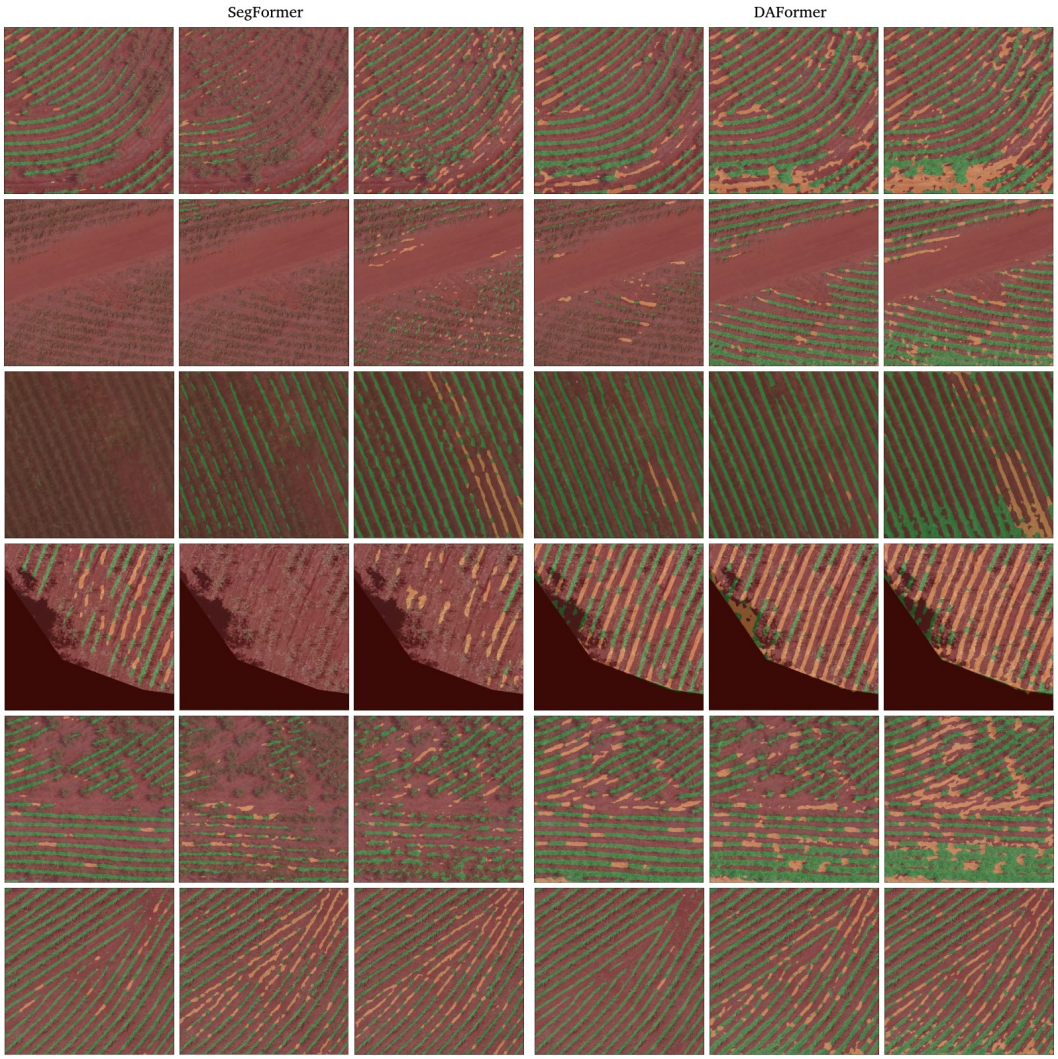

Figure 3.12: The results after applying UDA to the same images analyzed in Figure 3.6, on the right, and compared to src-only results obtained by SegFormer, on the left. Each column corresponds to the model trained using only labeled images from one of the source farms F_S, with $1 \leq S \leq 4$ and $S \neq T$.

by 33% compared to PSPNet Src-Only and by 12% compared to SegFormer Src-Only.

However, in addition to considering the similarity between the domains, the complexity of each dataset when performing UDA should also be taken into account. While a model trained with more data, whether in quantity or quality, can generalize better to smaller datasets, it is more challenging to adapt a simpler dataset to more complex ones. This could explain, for example, the variation in the F1-score between $F_1 \rightarrow F_2$ and $F_2 \rightarrow F_1$.

3.4.2 Rare Class Sampling (RCS)

Rare class sampling is a key strategy employed in DAFormer aimed at improving the accuracy of the least represented classes in the dataset. In our

	PSPNet Src-Only	SegFormer Src-Only	DAFormer UDA	DAFormer w/o RCS	Oracle (PSPNet)
F2 → F1	69.5 (90.4%)	70.0 (91.4%)	72.2 (94.0%)	**72.6 (94.5%)**	**76.8**
F3 → F1	55.8 (72.6%)	60.7 (79.0%)	61.9 (80.5%)	**65.7 (85.5%)**	**76.8**
F4 → F1	55.2 (71.8%)	64.4 (83.8%)	61.3 (79.8%)	**65.7 (85.5%)**	**76.8**
F1 → F2	56.0 (71.8%)	58.8 (75.4%)	**65.2 (83.6%)**	65.0 (83.4%)	**77.9**
F3 → F2	34.3 (44.3%)	48.3 (62.0%)	60.9 (78.1%)	**65.5 (84.0%)**	**77.9**
F4 → F2	33.0 (41.0%)	48.3 (62.0%)	54.0 (69.3%)	**57.0 (73.1%)**	**77.9**
F1 → F3	42.7 (53.2%)	51.0 (63.5%)	**57.1 (71.1%)**	51.5 (64.2%)	**80.2**
F2 → F3	45.8 (57.1%)	59.3 (73.9%)	**71.1 (88.6%)**	55.3 (68.9%)	**80.2**
F4 → F3	65.3 (81.4%)	69.7 (86.9%)	**70.5 (87.9%)**	68.8 (85.7%)	**80.2**
F1 → F4	64.2 (76.7%)	**68.6 (81.9%)**	67.9 (81.1%)	63.1 (75.3%)	**83.7**
F2 → F4	55.0 (65.7%)	54.3 (64.8%)	**76.2 (91.0%)**	57.4 (68.5%)	**83.7**
F3 → F4	60.0 (71.6%)	70.1 (83.7%)	**76.3 (91.1%)**	75.5 (90.2%)	**83.7**

Table 3.8: F1-score average comparison of src-only and UDA methods to oracle (fully supervised PSPNet). In parentheses, the relative performance of each method with reference to oracle. In bold, the best results for each pair of datasets $F_S \rightarrow F_T$.

experiments, DAFormer without RCS exhibited better performance on targets $F1$ and $F2$, when trained using images from $F3$ and $F4$, datasets with a higher prevalence of annotated gaps, as indicated in Table 3.2. Conversely, there was significant improvement when employing DAFormer with RCS to adapt $F2 \rightarrow F4$, underscoring the importance of RCS when the source dataset has significantly fewer representations of a class (in this case, gaps) compared to the target dataset.

In Figure 3.13, we can see that when using DAFormer with RCS, the model is able to increase gaps detection. However, this behavior can also lead to false positives, classifying background pixels as gaps, thereby decreasing the F1-score. In addition, as mentioned in the DAFormer paper, it is possible to adjust the RCS temperature T for optimal results, although this step adds an additional layer of complexity to the experiments.

An issue encountered in our experiments with DAFormer was that both DAFormer models trained using source images and labels from farm $F4$ exhibited some corruption at the bottom of predictions for the target farms, as

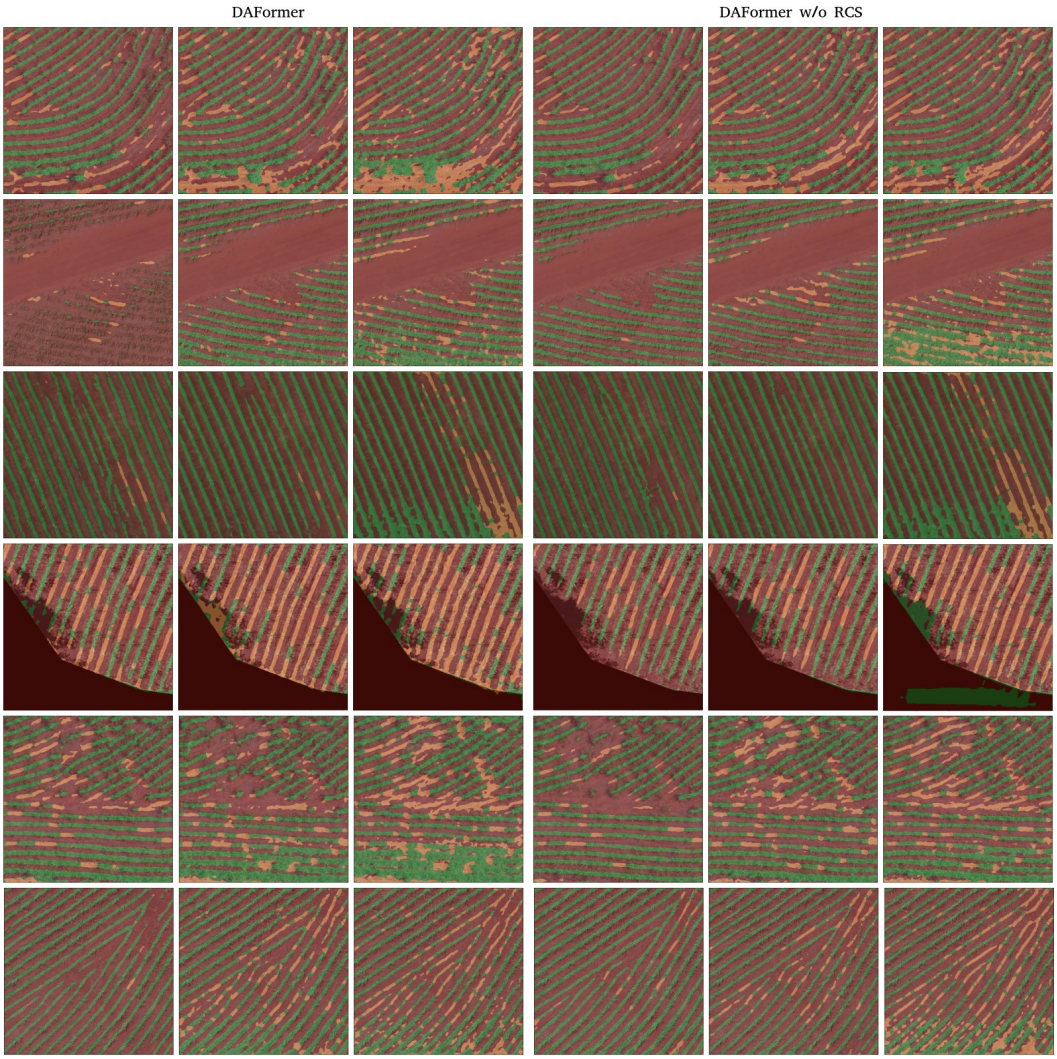

Figure 3.13: Visual results applying UDA to the same images analyzed in Figure 3.6. Each column corresponds to the model trained using only labeled images from one of the source farms F_S, with $1 \leq S \leq 4$ and $S \neq T$. On the left, all target images were classified using the DAFormer model trained using RCS. On the right, the results achieved when using models trained without RCS.

depicted in Figure 3.13. In the third and sixth columns from left to right, unexpected distortions are evident at the bottom of the images. These distortions likely impacted the results obtained in the $F4 \rightarrow F_T$ experiments, where $1 \leq T \leq 3$.

Finally, Figure 3.14 illustrates the main findings and contributions of these experiments. It shows the visual performance evolution of the techniques used to classify images from target farms $F2$, $F3$, and $F4$, training our models using only labeled images from the source farm $F1$, and comparing them with the oracle and ground truth. Subsequent Figures 3.15, 3.16, and 3.17 demonstrate the visual performance evolution when training using images from source farms $F2$, $F3$, and $F4$, respectively.

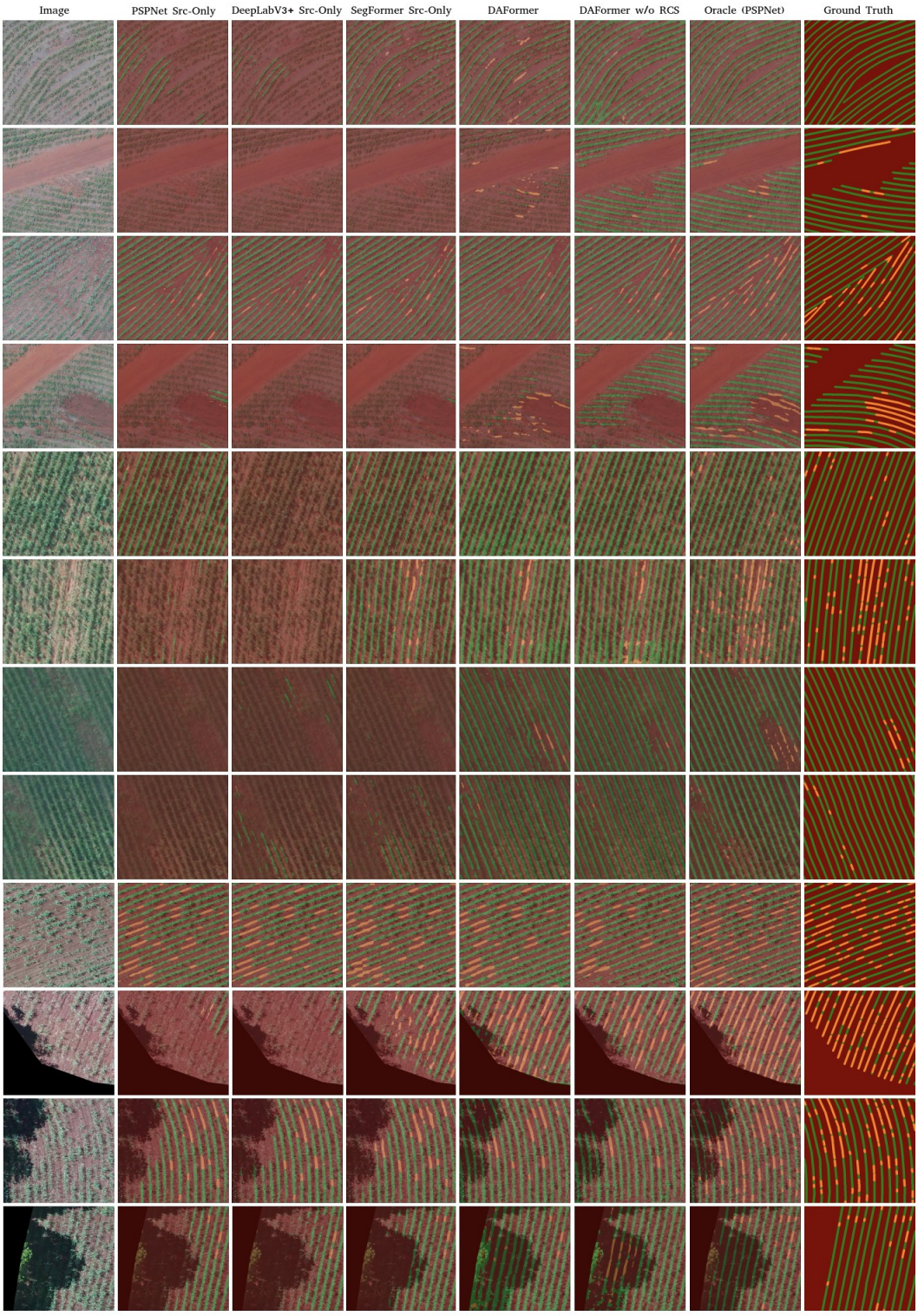

Figure 3.14: From left to right, visual performance evolution of the methods used to classify images from target farms $F2$, $F3$ and $F4$ training our models using only labeled images from source farm $F1$, and comparing with oracle and ground truth.

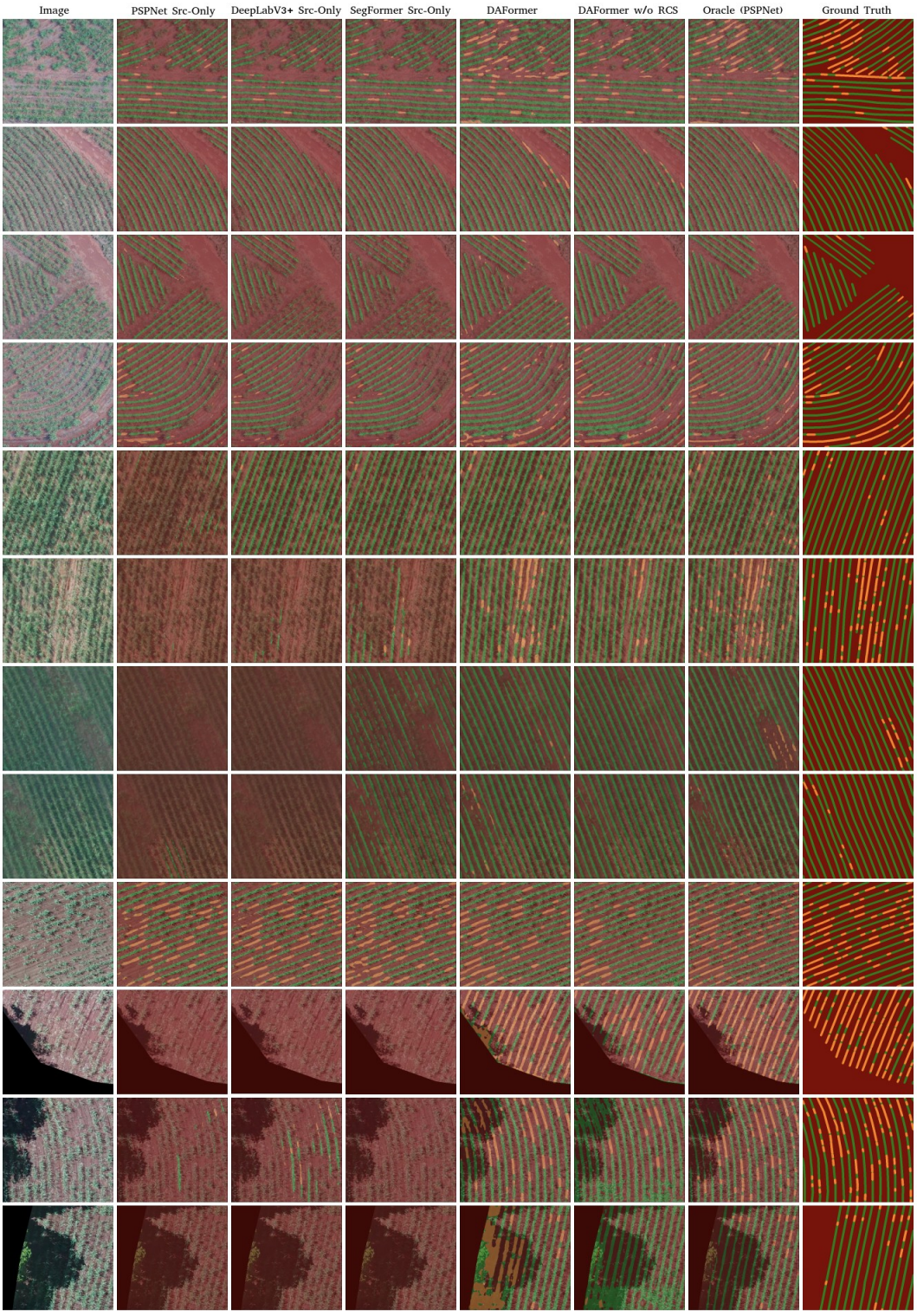

Figure 3.15: From left to right, visual performance evolution of the methods used to classify images from target farms $F1$, $F3$ and $F4$ training our models using only labeled images from source farm $F2$, and comparing with oracle and ground truth.

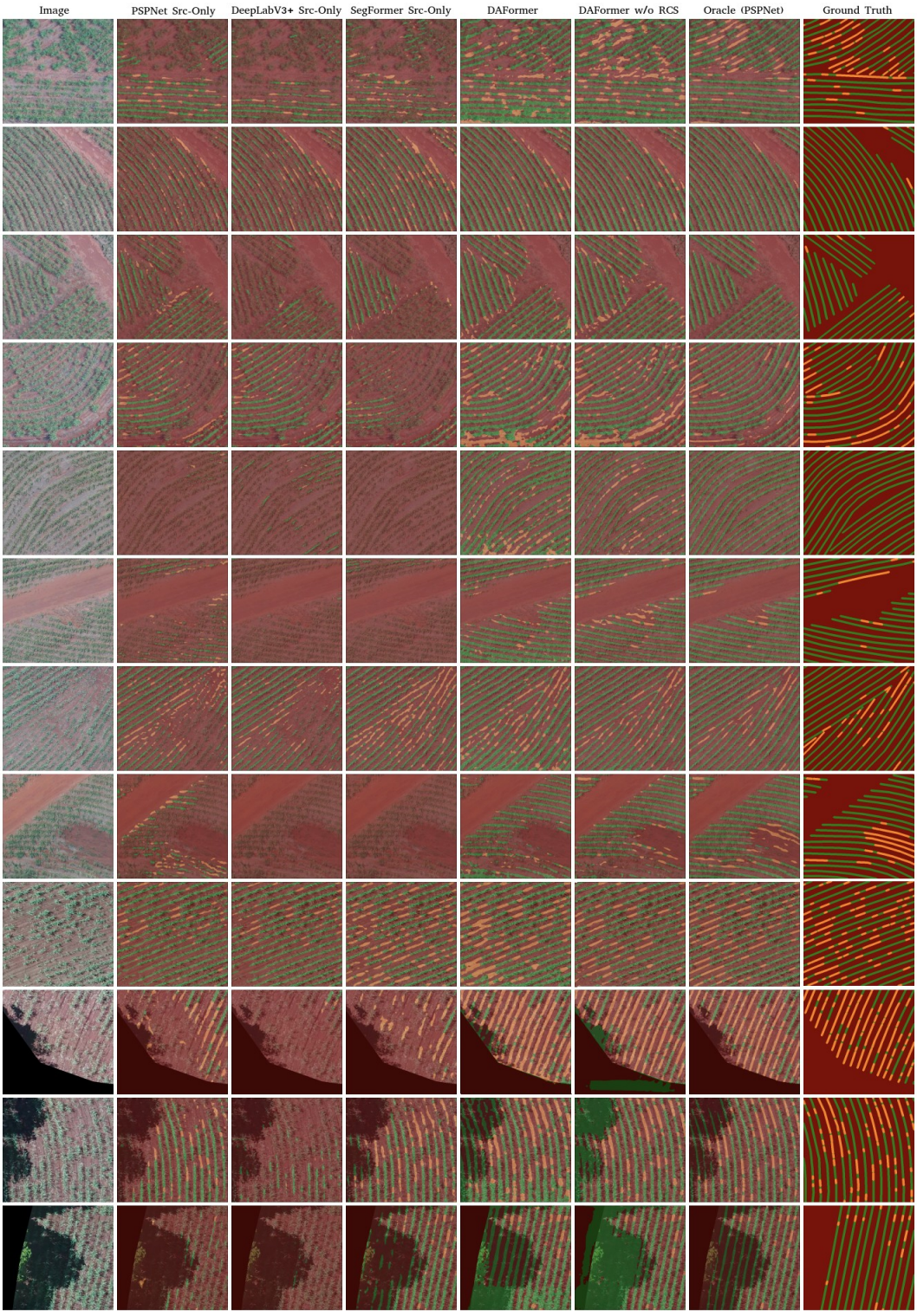

Figure 3.16: From left to right, visual performance evolution of the methods used to classify images from target farms $F1$, $F2$ and $F4$ training our models using only labeled images from source farm $F3$, and comparing with oracle and ground truth.

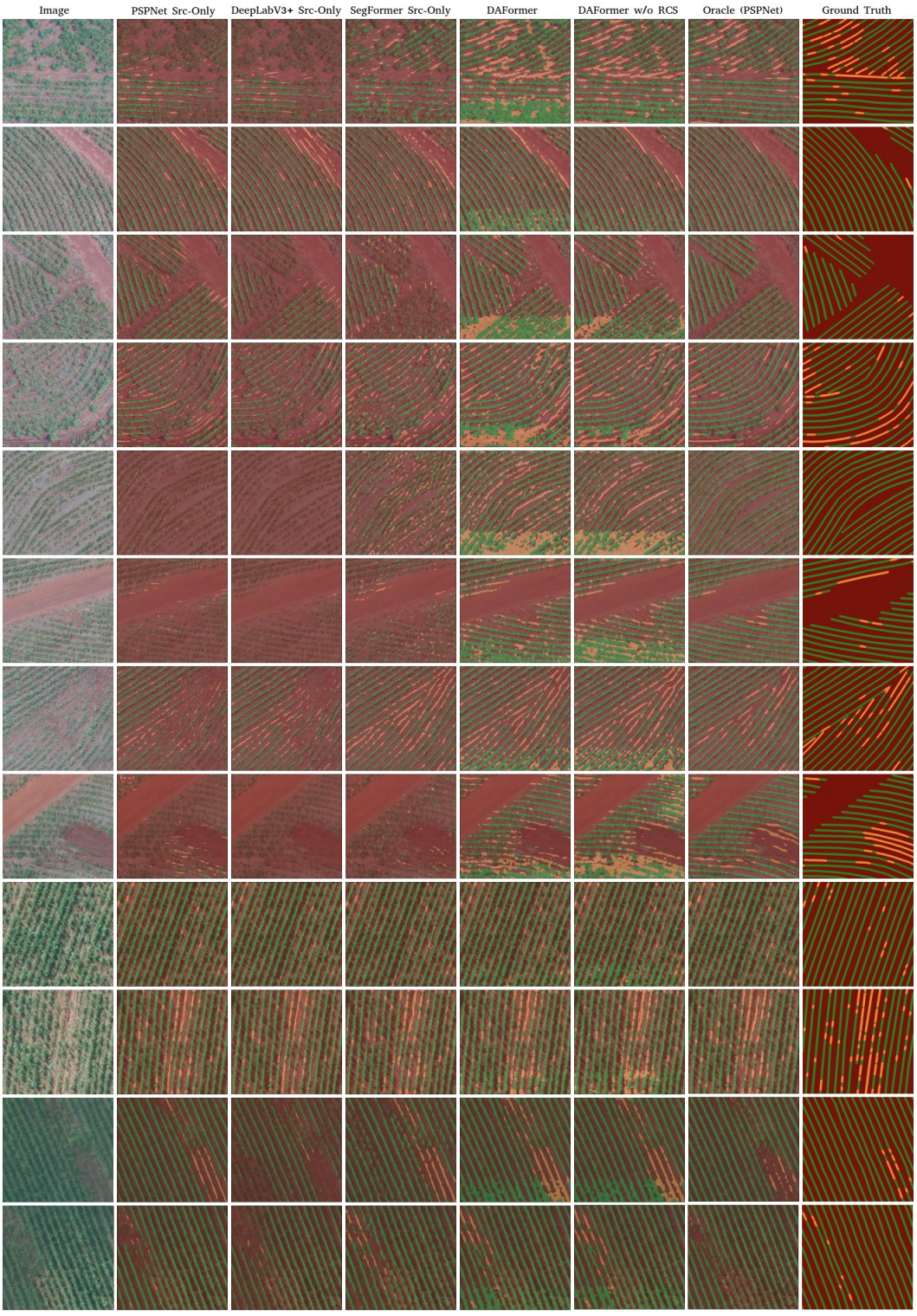

Figure 3.17: From left to right, visual performance evolution of the methods used to classify images from target farms $F1$, $F2$ and $F3$ training our models using only labeled images from source farm $F4$, and comparing with oracle and ground truth.

3.5 Domain Generalization

Despite having images from different farms, in our previous experiments we trained the models using only images from the same farm at a time. One alternative approach to try to increase performance is to focus on domain generalization. In this case, we grouped images from different farms, expecting that this could benefit the generalization of the network's learning. To achieve this, we excluded each farm, one at a time, and performed the training using only images from the other three farms for each experiment.

In Table 3.9, we present the results. In general, our results using the images from grouped farms were significantly superior or, at worst, similar to those obtained when each farm was trained individually. Based on these experiments, we can conclude that, although we cannot provide strong proof that domain generalization helps the network uncover hidden properties that it could not find when trained using only individual farms, the results achieved using domain generalization are at least as good as those achieved when training with images from more similar farms, i.e., $F1 \leftrightarrow F2$ and $F3 \leftrightarrow F4$.

	F2→F1	F234→F1	F1→F2	F134→F2	F4→F3	F124→F3	F3→F4	F3→F124
DAFormer (SegFormer MiT-B5)								
Background	91.1	92.6	89.8	89.6	90.5	89.6	90.0	91.5
Rows	79.5	81.3	76.2	76.4	75.8	74.4	72.6	73.2
Gaps	46.0	48.1	29.6	28.7	45.2	45.9	66.4	66.2
Average	72.2	**74.0**	**65.2**	64.9	**70.5**	70.0	76.3	**77.0**
SegFormer (MiT-B5)								
Background	92.5	92.9	88.7	88.3	90.5	89.3	90.0	92.4
Rows	80.8	81.8	62.7	60.4	74.5	67.6	67.7	72.9
Gaps	36.5	42.0	25.1	22.3	44.0	36.7	52.4	59.1
Average	70.0	72.3	58.8	57.0	69.7	64.5	70.1	74.8

Table 3.9: F1-score of UDA evaluation with Domain Generalization, compared to src-only evaluation of SegFormer, using Domain Generalization in both experiments. The models were evaluated using ground truth generated with dilation 5. In bold, the best average result for each target farm.

Therefore, even if no hidden properties were found, using images from all available farms makes the training process more practical and did not demonstrate side effects. When training using images only from a specific domain, we would need to test different models to determine which one presents the

best performance due to the similarity between the domains. However, when using domain generalization, these brute-force evaluations are not required.

3.6 Conclusion

In this section, we presented an approach to detect sugarcane rows and gaps, reducing the problem to a segmentation task. Our method overcomes common challenges found in traditional techniques, generally based on the Hough Transform, which rely on line detection. The approach has demonstrated its robustness in handling challenges like curve detection and non-parallel lines. Furthermore, the proposed dilation method for generating semi-supervised segmentation maps helps mitigate the costly manual annotation process.

We also employed the SegFormer-based DAFormer model, an unsupervised domain adaptation network, to enhance the performance of row and gap detection across various farms different from those where the training data was collected. Additionally, we compared the ability of ConvNets and transformer-based methods to generalize knowledge to unseen data without domain adaptation. Our experiments highlighted the superior robustness of the SegFormer network to overfitting during training across several epochs, a feature that contributed to DAFormer's improved domain adaptation.

Lastly, the theoretical findings of this research can be applied to real-world scenarios in diverse ways. For example, they can be integrated with geolocated imagery to provide detailed information on the occurrence, length, and exact coordinates of gaps in sugarcane plantations. Mapping pixels classified as gaps to geographic coordinates can significantly accelerate the process of addressing these gaps, helping farmers reduce crop losses and, consequently, increase both productivity and revenue.

CHAPTER

4

Domain Adaptation using GANs and Diffusion Models for Tree Detection in Aerial Images

4.1 Introduction

Urban forests are increasingly recognized for their significant benefits to human well-being. They contribute to energy savings, reduce stormwater runoff and improve water quality (Velasquez-Camacho et al., 2023; Ventura et al., 2024). Additionally, these forests provide essential ecosystem services that combat climate change, such as carbon sequestration, oxygen generation, water cycling, soil conservation, and mitigation of the urban heat island effect. Automated tree mapping is essential for effective management of both native and invasive vegetation (Lv et al., 2023; Beloiu et al., 2023).

In this context, techniques such as semantic segmentation, which offer pixel-based classification, are increasingly employed across a range of applications. Recent advancements in tree detection, classification, and segmentation predominantly utilize deep learning networks, such as ConvNets (Ferreira et al., 2020; Iqbal et al., 2021; Jintasuttisak et al., 2022), applied to aerial RGB and multispectral imagery (Beloiu et al., 2023; Velasquez-Camacho et al., 2023; Ventura et al., 2024). More recently, transformers have also been utilized for tree counting in aerial images (Chen and Shang, 2022).

Accurately detecting individual tree from remote sensing data presents a significant challenge for traditional deep learning-based methods due to the

variability encountered in cross-regional scenarios (Wang et al., 2022; Kapil et al., 2024; Zheng et al., 2020). This variability can arise from various factors, including deformations or shifts caused by biased sampling in the spatial domain, changes in acquisition conditions (such as variations in illumination or acquisition angle), or seasonal changes (Tuia et al., 2021).

Despite substantial advancements with deep neural networks, their performance improvement largely depends on the availability of extensive labeled training data, which involves costly and labor-intensive data curation (dos Santos Ferreira et al., 2019; Amirkolaee et al., 2024). The challenge is further compounded when a deep neural network must handle multiple distinct domains. For instance, in tree detecting, each domain might include different scenes (e.g., urban, countryside, farmland), imagery types (e.g., aerial or satellite), and varying levels of tree density, shadows, or overlap among individual trees.

To overcome these challenges, recent works have focused on applying unsupervised domain adaptation in satellite and aerial images. Zheng et al. (2020) proposed a domain-adaptive method to detect and count cross-regional oil palm trees using an adversarial learning-based multi-level attention mechanism. Wang et al. (2022) also employed an adversarial domain-adaptive model with a transferable attention mechanism for tree crown detection using high-resolution remote sensing images. More recently, AdaTreeFormer was introduced by Amirkolaee et al. (2024), demonstrating the ongoing trend of combining adversarial learning with attention mechanisms to perform domain adaptation for tree detection in high resolution images.

In this work, we propose a novel approach that differs from these previous studies. While we also utilize attention mechanisms for tree segmentation, instead of employing adversarial learning on high-resolution images, we perform domain adaptation with image-to-image translation models and super-resolution networks to enhance the quality of low-resolution aerial images.

Our method also addresses the challenge of limited labeled data by providing novel data augmentation techniques to generate additional training samples from the existing labeled data. This approach not only improves the model's performance in generalizing learning for images captured at different heights but also reduces the need for expensive labeling processes.

4.2 Methodology

4.2.1 Dataset

The images used in the experiments are separated into the datasets $P20$ and $P50$ based on the ground sample distance (GSD) utilized in the capture of

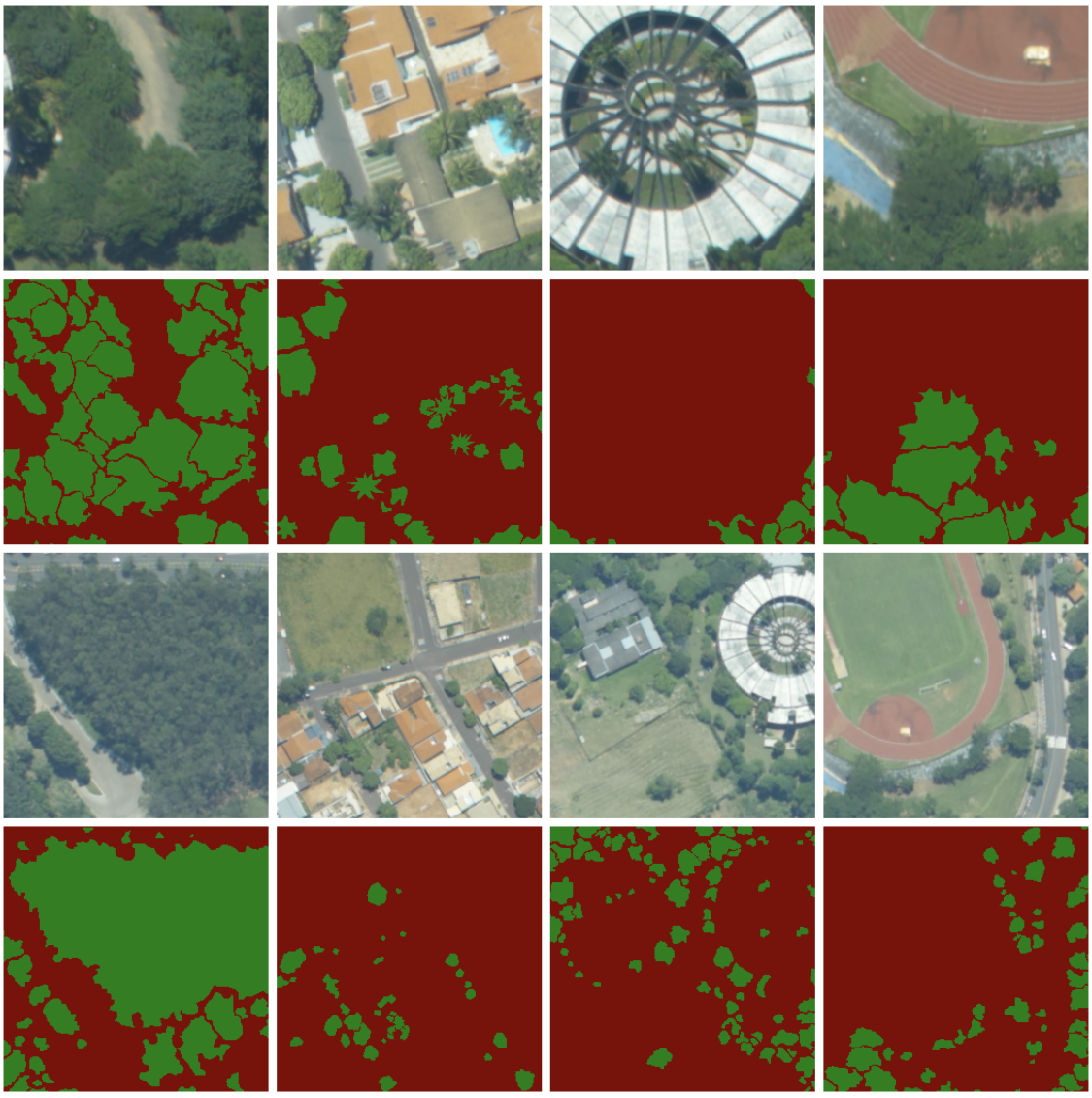

Figure 4.1: At the top are sample images from dataset $P20$ with their respective pixel annotations. At the bottom are sample images from dataset $P50$ with their respective pixel annotations.

the images. The $P20$ dataset consists of 363 images sized 256×256 pixels with a 20-centimeter GSD, i.e., each pixel corresponds to approximately 20 cm in the real world. The $P50$ dataset consists of 224 images sized 256×256 pixels with a 50-centimeter GSD. Thus, the resolution of the images in the $P20$ dataset is 2.5 times greater than that of the images in the $P50$ dataset.

The images consist of aerial views of urban environments and have been manually annotated by specialists as either background or tree classes. Sample images from both datasets, along with their respective annotations, can be seen in Figure 4.1, and the distribution of images in these datasets is shown in Table 4.1.

Dataset	GSD	Train	Validation	Test	Total
P20	20cm	218	36	109	363
P50	50cm	134	23	67	224

Table 4.1: Total of images of train (60%), validation (10%) and test (30%) sets for datasets *P*20 and *P*50 and their respective GSD.

4.2.2 Method

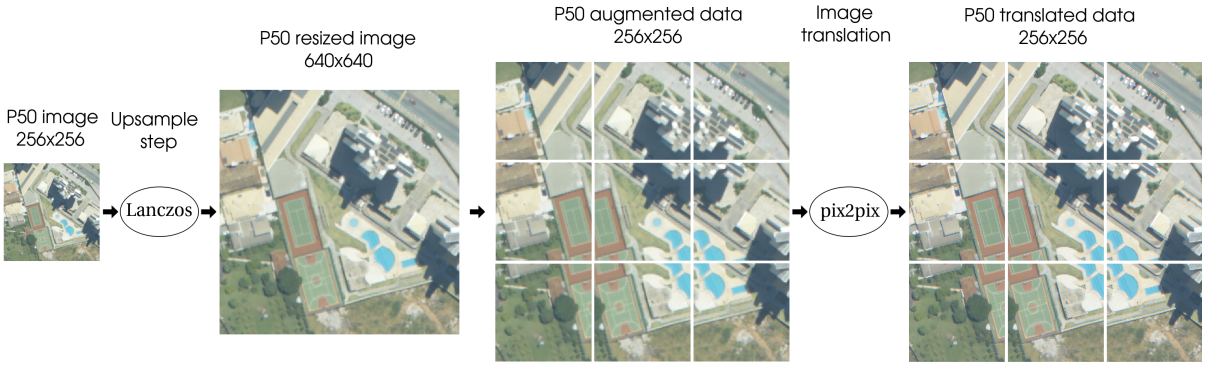

Figure 4.2: The images of the *P*50 dataset are resized to 640×640 using Lanczos resampling. For each resized image, we generated 9 patches of size 256×256 and translated them using pix2pix-trained models.

The difference in ground sample distance between our datasets affects the area in pixels used to represent different elements of the image, such as trees and roads, as seen in Figure 4.1. While the size of these elements tends to be similar within the same dataset, it consistently differs between datasets, posing an obstacle to teaching-student techniques such as those used in DAFormer. The strategy we propose to address this problem is to make the size of elements in both datasets similar, i.e., adjust the GSD of the *P*50 dataset to match the value used in the *P*20 dataset using upsampling techniques.

We developed two different methods to implement this strategy. In our first method, we upsample the *P*50 dataset, which has a $2.5\times$ difference in centimeters per pixel compared to the *P*20 dataset, by resizing the images from 256×256 to 640×640 using the default *ImageMagick* filter, Lanczos resampling (Duchon, 1979; Still, 2006), to make the size of objects similar to those in the *P*20 dataset. After this step, we generated 9 patches of size 256×256.

This process also augments the data in the *P*50 dataset by a factor of 9, increasing it from 224 images to 1,206 images. However, this procedure significantly decreases the resolution of these images, which could hamper the performance of network training and increase the data shift compared to the other

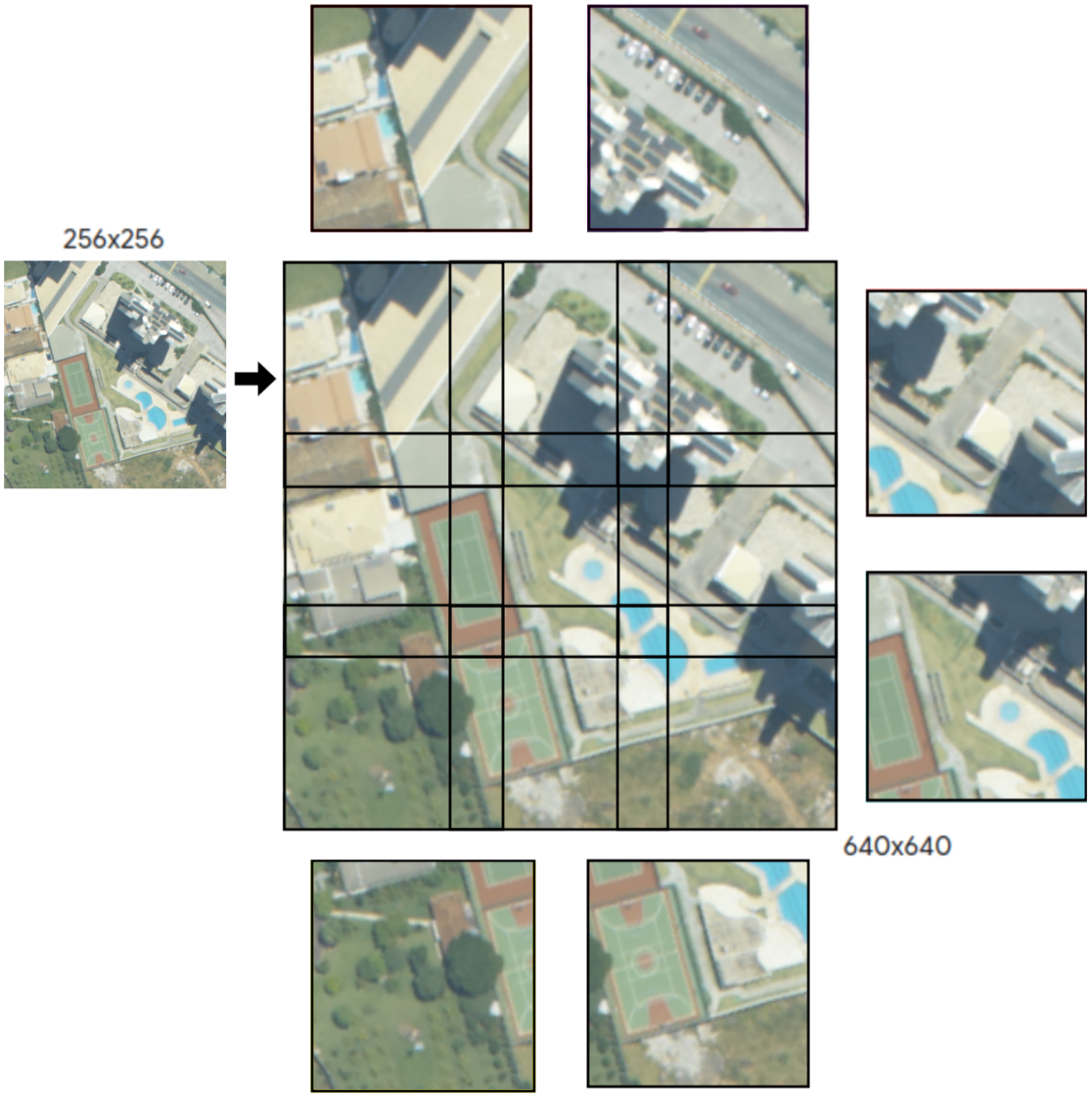

Figure 4.3: The images of *P*50 dataset are resized from 256×256 to 640×640 using Lanczos resampling method. After this step, we augmented the data, generating 9 patches of size 256×256.

dataset. To overcome this drawback, we trained pix2pix models to perform image-to-image translation and address the loss of resolution. The pipeline of this method can be seen in Figure 4.2. A more detailed visualization of the process for generating patches is illustrated in Figure 4.3.

In our second approach, we used recent super-resolution GANs and Diffusion models to upsample the images directly without loss of quality. The pipeline for this method is illustrated in Figure 4.4. The advantage of this approach is that we can leverage publicly available models trained on millions of images, unlike the pix2pix model, which needed to be trained from scratch with image pairs generated from our training sets. However, these models do not achieve direct image-to-image translation between the two domains; they primarily enhance resolution to compensate for quality loss during upsam-

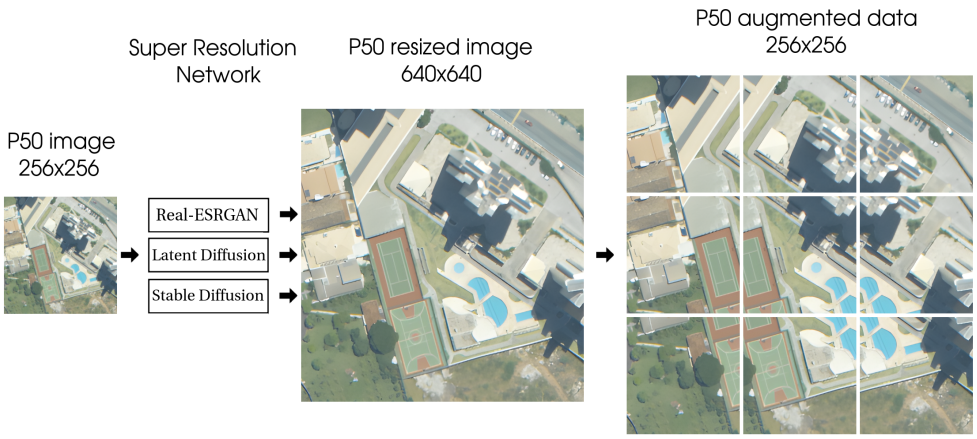

Figure 4.4: The images of the *P*50 dataset are resized to 640×640 using Real-ESRGAN, Latent and Stable Diffusion. For each resized image, we augmented the data, generating 9 patches of size 256×256.

pling.

Additionally, it is important to highlight that both approaches used here produce nine times more data from the original images, with these new images having a 2.5 times superior ground sample distance. Since we also updated all annotations for the new GSD automatically, this process helps address the cost of pixel-annotated data and mitigates the drawbacks of low-resolution aerial images. It produces significantly more high-quality annotated data, which is required to train deep learning models efficiently, in a fully automatic way. In the following sections, we provide more details about the methods used.

4.2.2.1 pix2pix

Dataset	**Generation Method**	**GSD**	**Train**	**Validation**	**Test**	**Total**
P50-20p	pix2pix trained with *P*20 pairs	20cm	1206	207	603	2016
P50-50p	pix2pix trained with *P*50 pairs	20cm	1206	207	603	2016

Table 4.2: Total of images of train (60%), validation (10%) and test (30%) sets for the datasets generated using pix2pix translation. Image pairs used in the training of *P*50-20p can be seen in Figure 4.5, and those used in the training of *P*50-50p can be seen in Figure 4.6.

Pix2pix is an image-to-image translation GAN and has shown promising results in datasets with a paired image relationship between the source and target domains, such as the Facade and Cityscapes datasets (Tyleček and Šára, 2013; Cordts et al., 2016). The image-to-image translation used here

could alleviate distortions in the generated images that might otherwise decrease the segmentation performance in subsequent steps. However, since we lack a direct relationship between the images of the two datasets, *P*20 and *P*50, to perform a true paired translation, we proposed two approximate mapping approaches.

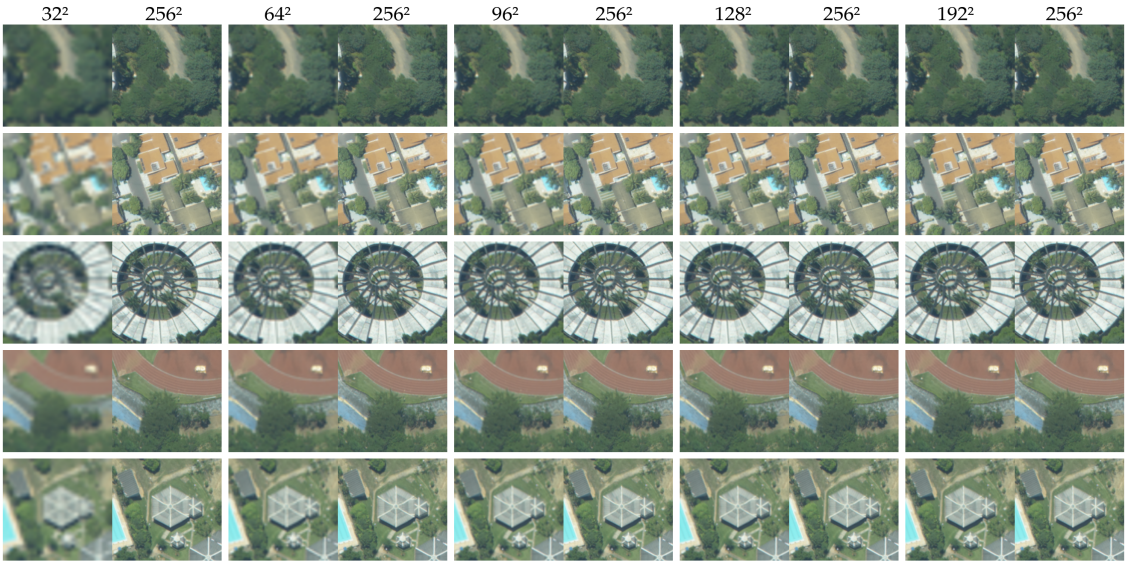

Figure 4.5: Pix2pix training pairs with images of the *P*20 dataset at resolutions of 32×32, 64×64, 96×96, 128×128, and 192×192.

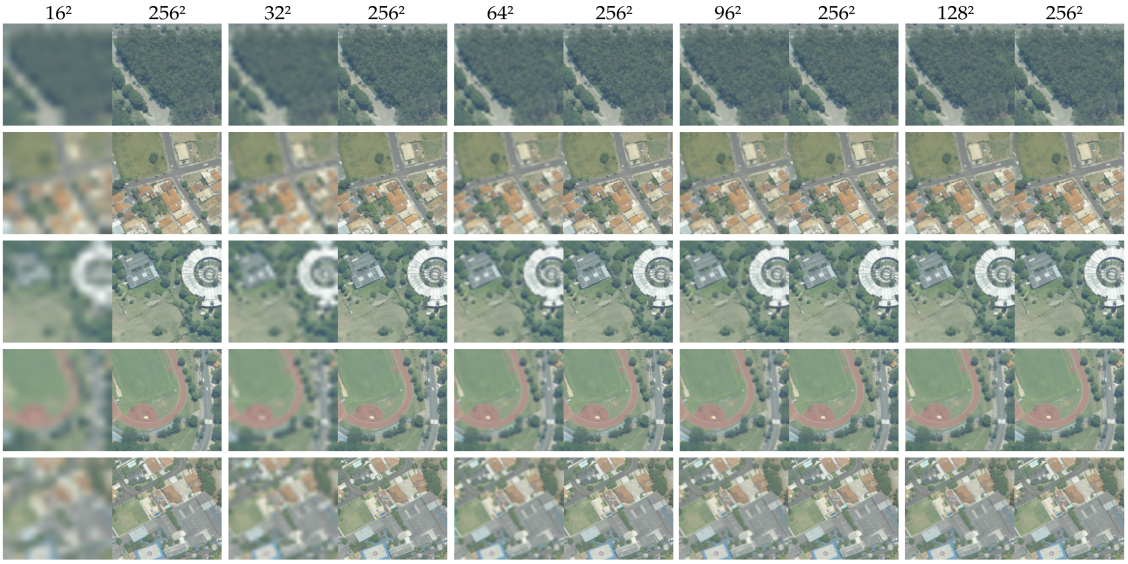

Figure 4.6: Pix2pix training pairs with images of the *P*50 dataset at resolutions of 16×16, 32×32, 64×64, 96×96, and 128×128.

To perform the mapping required for paired image-to-image translation used in pix2pix, we reduced the resolution of the images in datasets *P*20 and *P*50. For dataset *P*20, we used resolutions of 32×32, 64×64, 96×96, 128×128, and 192×192. For dataset *P*50, we used resolutions of 16×16, 32×32, 64×64,

96×96, and 128×128. After resizing to these smaller resolutions, we upscaled the images back to 256×256 without any preprocessing steps and generated the paired images illustrated in Figures 4.5 and 4.6. We trained two different pix2pix models using these pairs.

Figure 4.7: Sample images generated from datasets $P20$ and $P50$ using pix2pix ($P50-20p$ and $P50-50p$), Real-ESRGAN ($P20G$ and $P50G$), Latent Diffusion ($P20D$ and $P50D$), and Stable Diffusion ($P20S$ and $P50S$).

We used the images obtained after applying the Lanczos method to the $P50$ dataset as input for the pix2pix models, generating two new datasets: $P50-20p$ and $P50-50p$. The distribution of images in these datasets is described in Table 4.2. Sample images from these datasets are shown in Figure 4.7.

4.2.2.2 Real-ESRGAN, Latent and Stable Diffusion

We used the Real-ESRGAN and Diffusion public models, without any fine-tuning, to generate our 640×640 images from dataset $P50$. Using the resulting

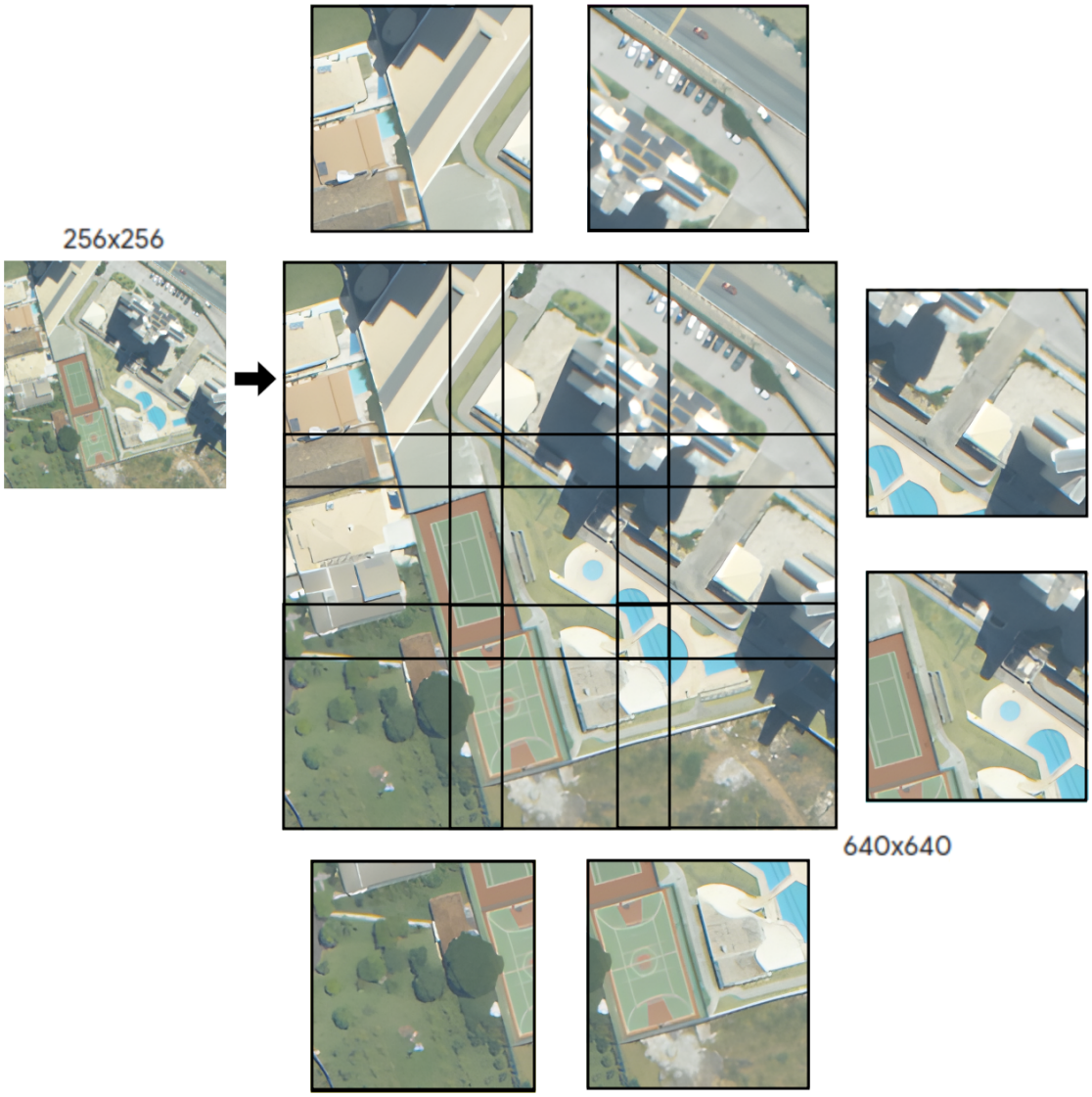

Figure 4.8: The images of *P*50 dataset are upscaled from 256×256 to 640×640 using Real-ESRGAN. After this step, we generated 9 patches of size 256×256. The visual quality is significantly better compared to the resized images shown in Figure 4.3.

super-resolution images, we generated 9 patches of size 256×256, as described in Figure 4.8. For dataset *P*20, we upscaled the original images to 640×640 using the models and then resized them back to the original size of 256×256 to maintain similarity with the images generated by the previous pipeline. The distribution of images in each generated dataset can be seen in Table 4.3, where the suffix G represents Real-ESRGAN, the suffix D represents Latent Diffusion, and the suffix S represents Stable Diffusion.

Using Stable Diffusion, we have the option to provide a prompt that guides the image generation. While this could be an advantage over Latent Diffusion, for this work, this feature poses a challenge in choosing a prompt that optimizes our segmentation results. Since evaluating the best prompt for the segmentation task is somewhat beyond the scope of this work, we only con-

Dataset	Generation Method	GSD	Train	Validation	Test	Total
P20G	Real-ESRGAN	20cm	218	36	109	363
P50G	Real-ESRGAN	20cm	1206	207	603	2016
P20D	Latent Diffusion	20cm	218	36	109	363
P50D	Latent Diffusion	20cm	1206	207	603	2016
P20S	Stable Diffusion	20cm	218	36	109	363
P50S	Stable Diffusion	20cm	1206	207	603	2016

Table 4.3: Total of images of train (60%), validation (10%) and test (30%) sets for each super-resolution dataset.

sidered a few prompts and selected them based on the best qualitative visual results.

In Figure 4.9, we can observe visual results based on these simple prompts. It is noticeable that when specifying that the images are aerial images, the network generates textures specific to elements such as trees or roofs. We selected this prompt for generating our images, despite the fact that these generated textures may not necessarily improve performance in the segmentation experiments.

4.2.3 Evaluation Metrics

To assess and compare the networks evaluated in the experiments, we used the metric commonly applied in the literature: intersection over union (IoU) at the pixel level, described in Equation 4.1.

$$IoU = \frac{P \cap GT}{P \cup GT} \tag{4.1}$$

where P corresponds to model prediction and GT corresponds to Ground Truth.

In all experimental results presented here, the notation $P_S \rightarrow P_T$ indicates that the model was trained on images from dataset P_S and evaluated on test images from dataset P_T. Thus, $S = T$ signifies supervised segmentation, where both training and test images come from the same dataset, while $S \neq T$ denotes a scenario where the model is trained on one dataset and evaluated on a different dataset.

Prompt: Enhance the resolution of this image

Prompt: Enhance the resolution of this aerial city image
without applying any filter

Figure 4.9: Images upscaled with specific prompts using Stable Diffusion. When we specify that the images are aerial images, the network generates textures specific to elements such as trees or roofs.

4.2.4 Experimental Setup

We ran our experiments with SegFormer, pix2pix, and Real-ESRGAN using the free version of Google Colab with a T4 GPU. For experiments with DAFormer, Latent Diffusion, and Stable Diffusion, we utilized an Intel(R) Core (TM) i7-5820K CPU @ 3.30GHz with 32 GB of RAM, and an Nvidia GeForce GTX TITAN X GPU with 12 GB GDDR5 memory and 3072 CUDA Cores.

In our supervised segmentation tests with SegFormer, we utilized the available architectures in MMSegmentation, accessible at https://github.com/open-mmlab/mmsegmentation. For training, we used the base configuration files provided by MMSegmentation, specifically using the Cityscapes configuration with the MIT-B5 backbone, a crop size of 1024×1024, and a learning rate schedule set at 160000. Additionally, we adjusted the image scale to 256×256, modified the number of classes in the decode/auxiliary head to 2, and resized the crop size to 128×128 to better suit our dataset.

In our experiments with DAFormer, we employed the configuration described in Section 3.2.4, with adjustments made to the image scale set to 256×256 and the crop size to 128×128 to adapt to our dataset.

For pix2pix training, we utilized the original code provided by the authors, accessible at github.com/junyanz/pytorch-CycleGAN-and-pix2pix. Each model was trained for 200 epochs with decay initiated after 100 epochs. No additional training or fine-tuning was conducted for Real-ESRGAN. Inference was performed using the default configurations provided in the script available from the authors' repository at github.com/xinntao/Real-ESRGAN.git.

For Latent and Stable Diffusion, we utilized the implementation provided by the authors in python library format, accessible at github.com/CompVis/latent-diffusion and github.com/CompVis/stable-diffusion. The images resulting from inference by the GANs and Diffusion models were used to train the SegFormer model.

Unlike Real-ESRGAN, the outscale parameter of the pre-trained Diffusion models could not be adjusted to a value smaller than 4. Due to our machine's 12GB memory limitation, we were unable to resize images from 256×256 to 1024×1024 directly. Therefore, we divided our original images into 4 patches of 128×128, upscaled them using the Diffusion models, and then used the 4 upscaled patches to reconstruct the image with size 1024×1024. We acknowledge that this step could have impacted our results and consider this aspect a limitation of the Diffusion pre-trained models.

4.3 Supervised Semantic Segmentation

4.3.1 Baseline

We evaluated the performance of supervised segmentation using SegFormer on two original datasets, *P*20 and *P*50, without upsampling the original images. The results are presented on the left side of Table 4.4. While both datasets achieved considerable performance in terms of IoU metric, dataset *P*20 exhibited a higher IoU than dataset *P*50. This outcome was anticipated, given that dataset *P*20 comprises higher-resolution images and a larger training set.

	P20 → P20	**P50 → P50**	**P50 → P20**	**P20 → P50**
SegFormer (MiT-B5)				
Background	94.87	95.56	91.05	94.22
Trees	**77.44**	70.18	57.43	63.27
Average	86.15	82.87	74.25	78.75

Table 4.4: IoU of supervised training using the original datasets. On the right side, the source model results are shown. In bold, the best result for the Trees class.

We also evaluated the models on a different dataset than those used for training (i.e., source model only). The results are presented on the right side of Table 4.4. When segmenting target images with models trained on images from a different domain, a noticeable decrease in IoU is observed due to data shift. This performance drop is particularly pronounced when using the model trained on dataset *P*50 to segment images from dataset *P*20, where the IoU decreases from 77.44 to 57.43 for the Trees class, approximately a 25.8% drop.

In Figure 4.10 we can see the visual predictions using the SegFormer model trained with images from datasets *P*20 and *P*50. The models performed well even when segmenting images from a different domain. However, the *P*50 model failed to detect some large trees and occasionally misidentified grass as trees in the *P*20 images. The *P*20 model failed to detect smaller trees in the *P*50 images, but the reduced size of the trees generated a smaller impact on the average IoU.

Although we can consider the performance of the source model only reasonable in these experiments, given the similarity of the images in both datasets, the next sections analyze techniques aimed at improving these results, as well as enhancing the performance of supervised segmentation.

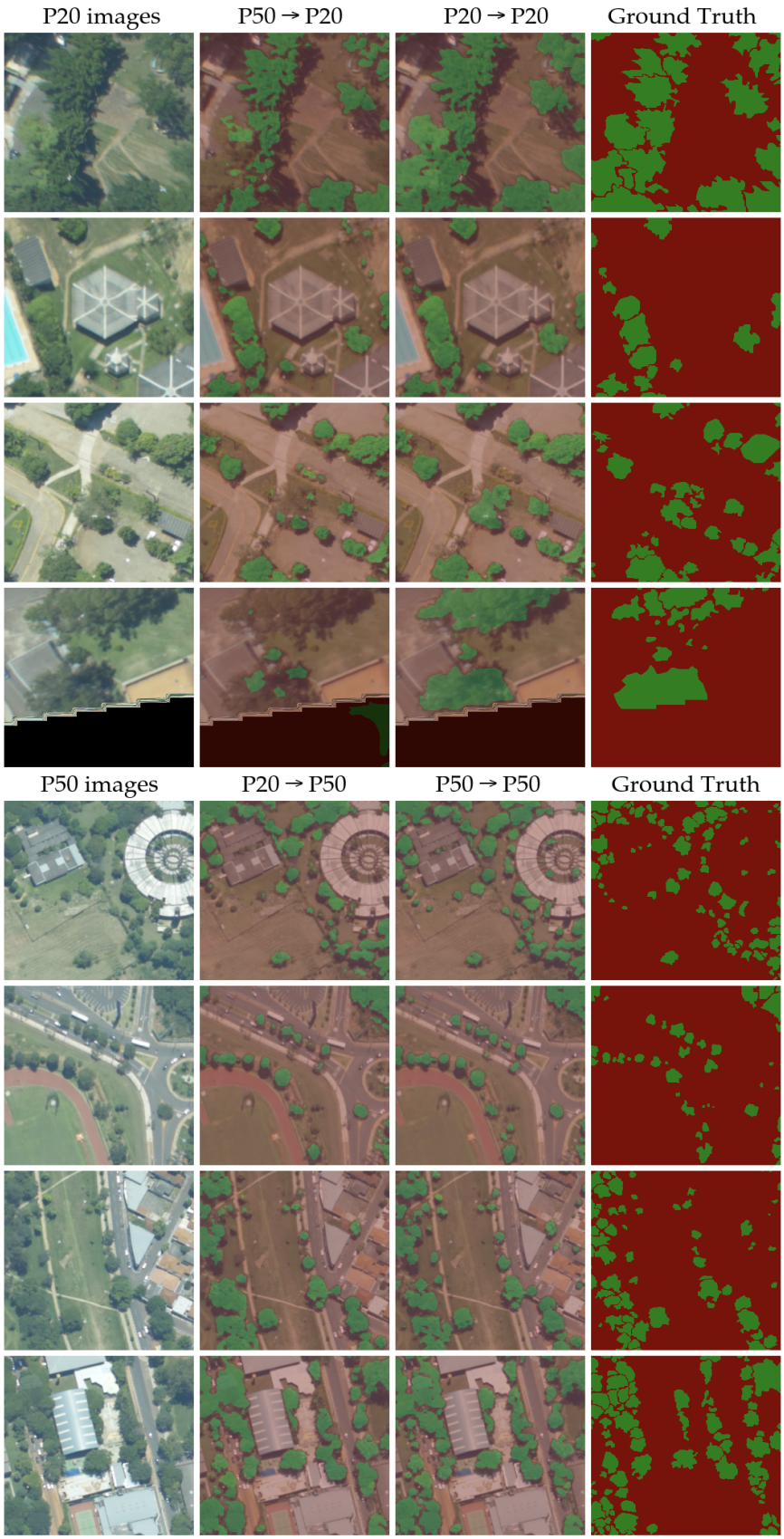

Figure 4.10: Predictions using the SegFormer model trained with images from datasets *P*20 and *P*50. The models performed well even when segmenting images from a different domain.

4.4 Unsupervised Domain Adaptation

4.4.1 DAFormer

To analyze the performance of Unsupervised Domain Adaptation, we used the SegFormer-based model DAFormer with the original images from datasets $P20$ and $P50$. The results can be seen in Table 4.5.

	P20 → P50	**P50 → P20**
DAFormer (SegFormer MiT-B5)		
Background	94.07	88.39
Trees	**63.27**	43.93
Average	78.67	66.16
SegFormer (MiT-B5)		
Background	94.22	91.05
Trees	**63.27**	57.43
Average	78.75	74.25

Table 4.5: IoU of the DAFormer evaluation compared to the SegFormer src-only. In bold, the best result for the Trees class.

When using dataset $P20$ as the source, our results were very similar to the src-only approach, using SegFormer without applying UDA. However, when using dataset $P50$ as the source, our results were significantly inferior to the performance of the source model only. A possible explanation for this discrepancy is that the trees in the source dataset $P50$ consistently appear at a much smaller size compared to those in the target dataset $P20$. This consistent difference may have influenced the DAFormer's self-training, which employs a student and teacher models approach, to not recognize larger-sized trees in the target dataset as trees.

4.4.2 Paired Image-to-Image Translation

4.4.2.1 pix2pix

We trained two pix2pix models using the pairs described in Section 4.2.2. These models were used to generate two new datasets, $P50-20p$ and $P50-50p$, which consist of translated images from dataset $P50$ after applying the upsampling process. The results of the SegFormer supervised segmentation trained

with these models can be seen in Table 4.6. In both cases, we observe an improvement in IoU compared to supervised segmentation using the original images.

	P50-20p→P50-20p	**P50-50p→P50-50p**	**P50→P50**
SegFormer (MiT-B5)			
Background	96.05	95.99	95.56
Trees	**73.25**	72.77	70.18
Average	84.65	84.37	82.87

Table 4.6: IoU of supervised training with images generated by the pix2pix models. compared to the original datasets. In bold, the best result for the Trees class.

However, it is important to highlight that we are not evaluating the translated images from dataset *P*50 directly but rather the corresponding augmented data generated through the upsampling process; thus, this improvement could also be attributed to the data augmentation process.

Nevertheless, it is an interesting finding that, in these experiments, we were able to enhance our segmentation results using the same network, SegFormer, without the need for more labeled images for training. Instead, we achieved this increase by generating more images at the same size but with lower resolution and then improving the quality using paired image-to-image translation, showing the potential of our data augmentation method.

	P20→P50-20p	**P20→P50-50p**	**P20→P50**	**P50-20p→P20**	**P50-50p→P20**	**P50→P20**
SegFormer (MiT-B5)						
Background	94.65	94.48	94.22	93.07	92.94	91.05
Trees	**67.20**	66.29	63.27	**68.05**	67.43	57.43
Average	80.92	80.38	78.75	80.56	80.19	74.25

Table 4.7: IoU of the src-only evaluation with images generated by the pix2pix models compared to the original datasets. In bold, the best results for the Trees class.

We also evaluated these models as source model only on the test images from dataset *P*20 and evaluated the model trained with images from dataset *P*20 on the images generated by pix2pix. The results can be seen at Table 4.7. In all tests, we achieved significant improvements compared to the results on the original images of dataset *P*50 without using image-to-image translation. The best model trained with pix2pix images improved the IoU for the Trees

class from 57.43 to 68.05, reducing the gap with the supervised results of $P20 \rightarrow P20$, 77.43, by approximately 60%.

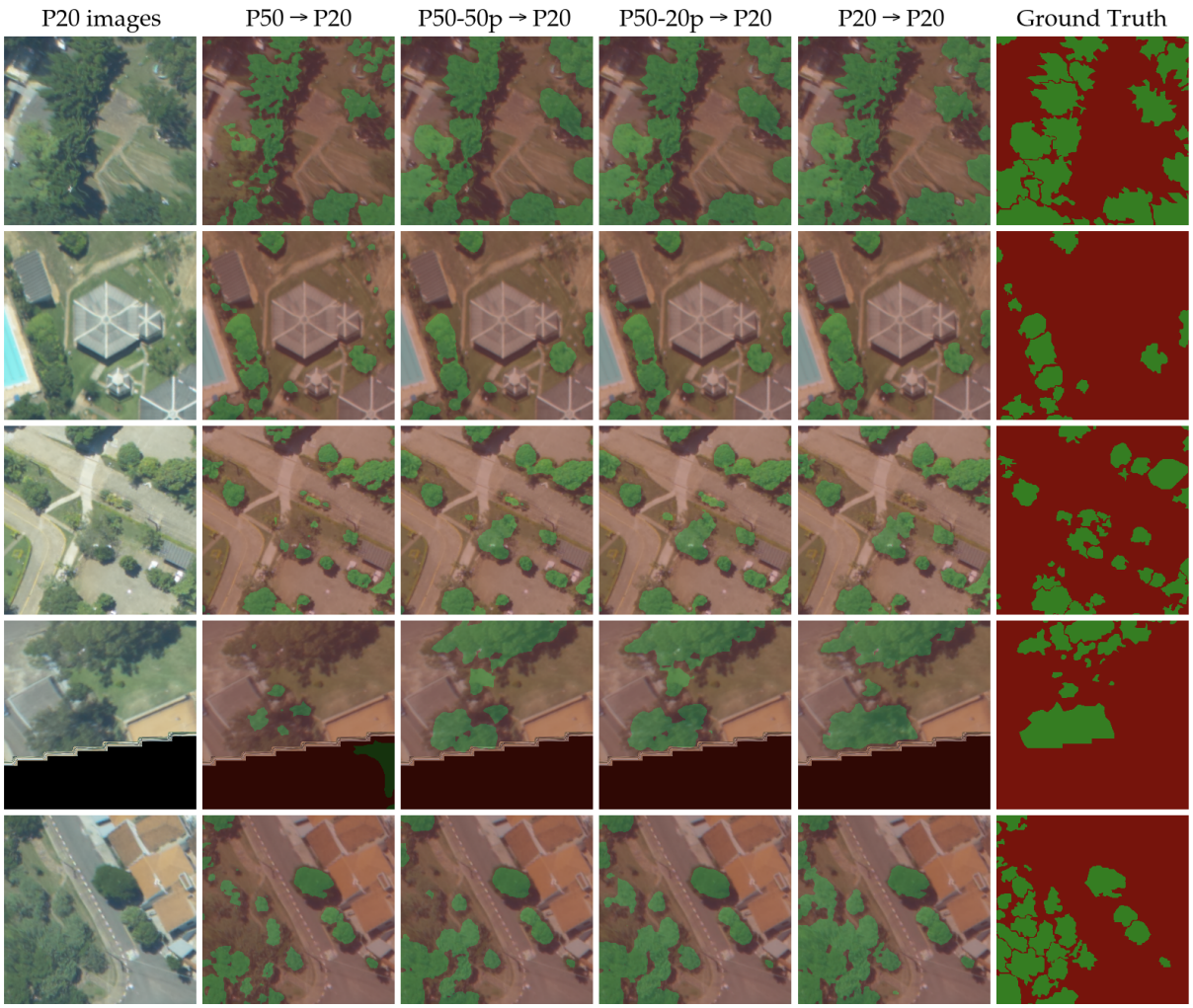

Figure 4.11: Predictions using the SegFormer model trained with images from datasets $P50-20p$ and $P50-50p$ in $P20$ images. In the bottom left corner of the first and last images, we can see the improvement of the pix2pix models in detecting larger trees.

4.4.3 Super-Resolution Models

We used the super-resolution models to generate high-resolution images from the datasets $P20$ and $P50$, as described in Section 4.2.2. We evaluated the SegFormer model trained on these images and compared its performance to training using the original images. The results for each network evaluated are detailed in the following sections.

4.4.3.1 Real-ESRGAN

Although the images generated by Real-ESRGAN exhibit superior visual quality compared to those generated by pix2pix models, as depicted in Figure

4.7, the results of our experiments were slightly inferior to those achieved by SegFormer trained with images translated by pix2pix models, as shown in Table 4.8. This difference can be attributed to the fact that while we trained the pix2pix models using images from our specific datasets, Real-ESRGAN uses a super-resolution model trained on general images.

	P20G → P50G	**P20 → P50**	**P50G → P20G**	**P50 → P20**
SegFormer (MiT-B5)				
Background	94.86	94.22	92.45	91.05
Trees	**66.57**	63.27	**63.92**	57.43
Average	80.71	78.75	78.19	74.25

Table 4.8: IoU of the src-only evaluation with images upscaled using Real-ESRGAN, compared to the original datasets. In bold, the best results for the Trees class.

This lack of training could have led the network to distort the semantic information of some pixels, resulting in a decrease in the segmentation results. However, it is worth highlighting that omitting the training step sped up our pipeline. Moreover, while semantic distortion of pixels can significantly impact segmentation tasks, in other tasks such as object detection, this effect is generally negligible.

4.4.3.2 Latent and Stable Diffusion

	P20D→P50D	**P20S→P50S**	**P20→P50**	**P50D→P20D**	**P50S→P20S**	**P50→P20**
SegFormer (MiT-B5)						
Background	94.42	94.63	94.22	92.59	91.63	91.05
Trees	65.58	**65.59**	63.27	**65.36**	62.73	57.43
Average	80.00	80.11	78.75	78.97	77.18	74.25

Table 4.9: IoU of the src-only evaluation with images upscaled using Latent and Stable Diffusion, compared to the original datasets. In bold, the best results for the Trees class.

With our Diffusion models, we obtained results similar to Real-ESRGAN, as shown in Table 4.9. We also experimented a combination of models trained using Latent and Stable Diffusion. One interesting finding was that our best results were achieved using a model trained with images from dataset *P*50*D* to

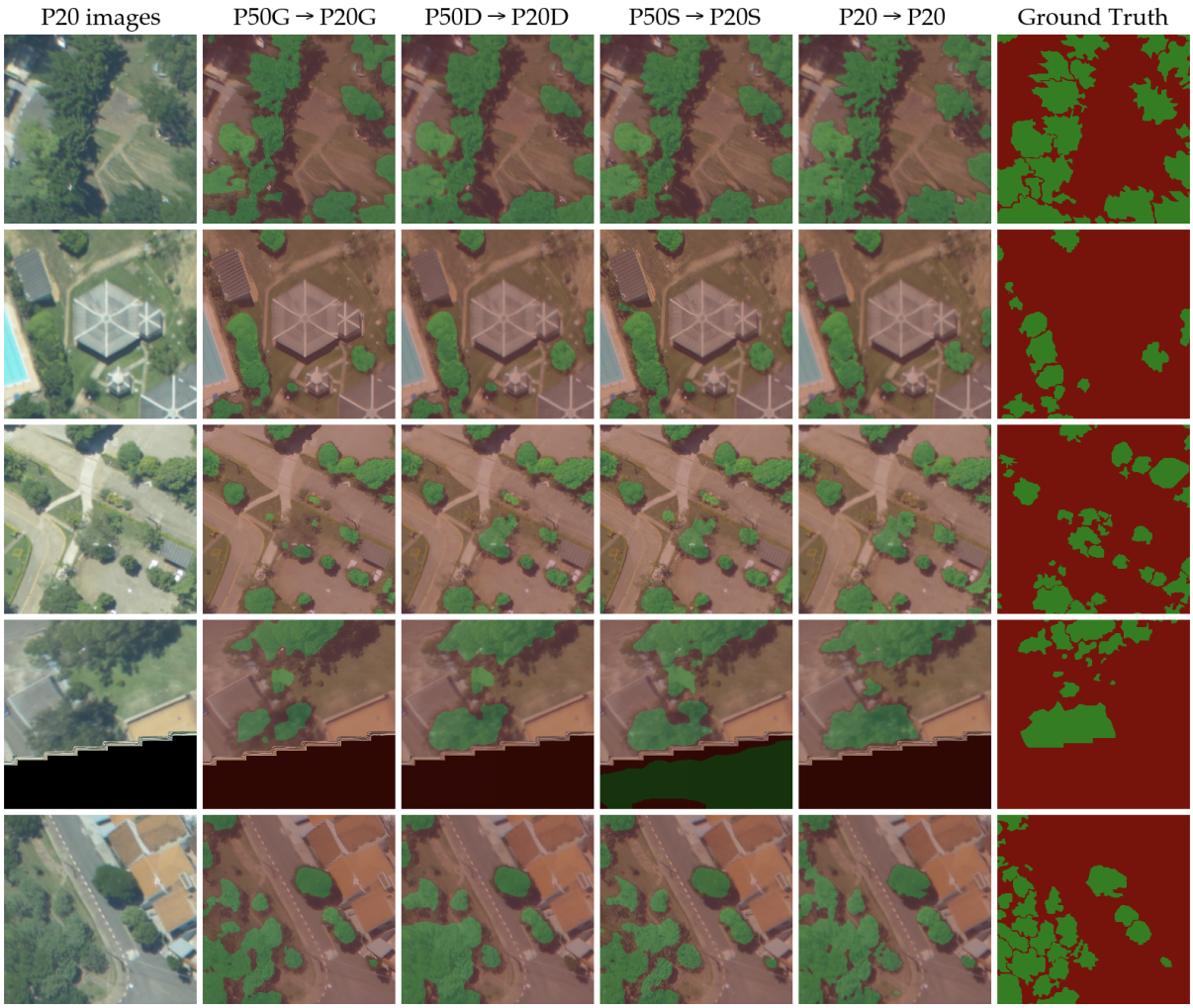

Figure 4.12: Latent diffusion produces better segmentation results than ESRGAN, despite the GAN model generating images with better visual quality. Ironically, Stable Diffusion suffers from instability in the fourth image, a behavior that may have been influenced by prompt usage.

segment the test images from dataset *P*20*S*, achieving an IoU of 67.79 for the Trees class, superior to our results shown in the Table.

However, it's difficult to establish a specific reason for this behavior, mainly due to the fact that the resulting images from Stable Diffusion are strongly influenced by the prompt used. Nevertheless, this aspect may highlight the possibilities that can be explored with the use of Stable Diffusion in similar tasks. In Figure 4.12, we can observe a visual comparison of the segmentation results of datasets generated by the super-resolution methods.

4.5 Low Resolution Images

Despite a 2.5-fold resolution difference between our original datasets *P*20 and *P*50, the visual quality in both cases was good, and the slight disparity in resolution between the datasets allowed us to achieve satisfactory results with

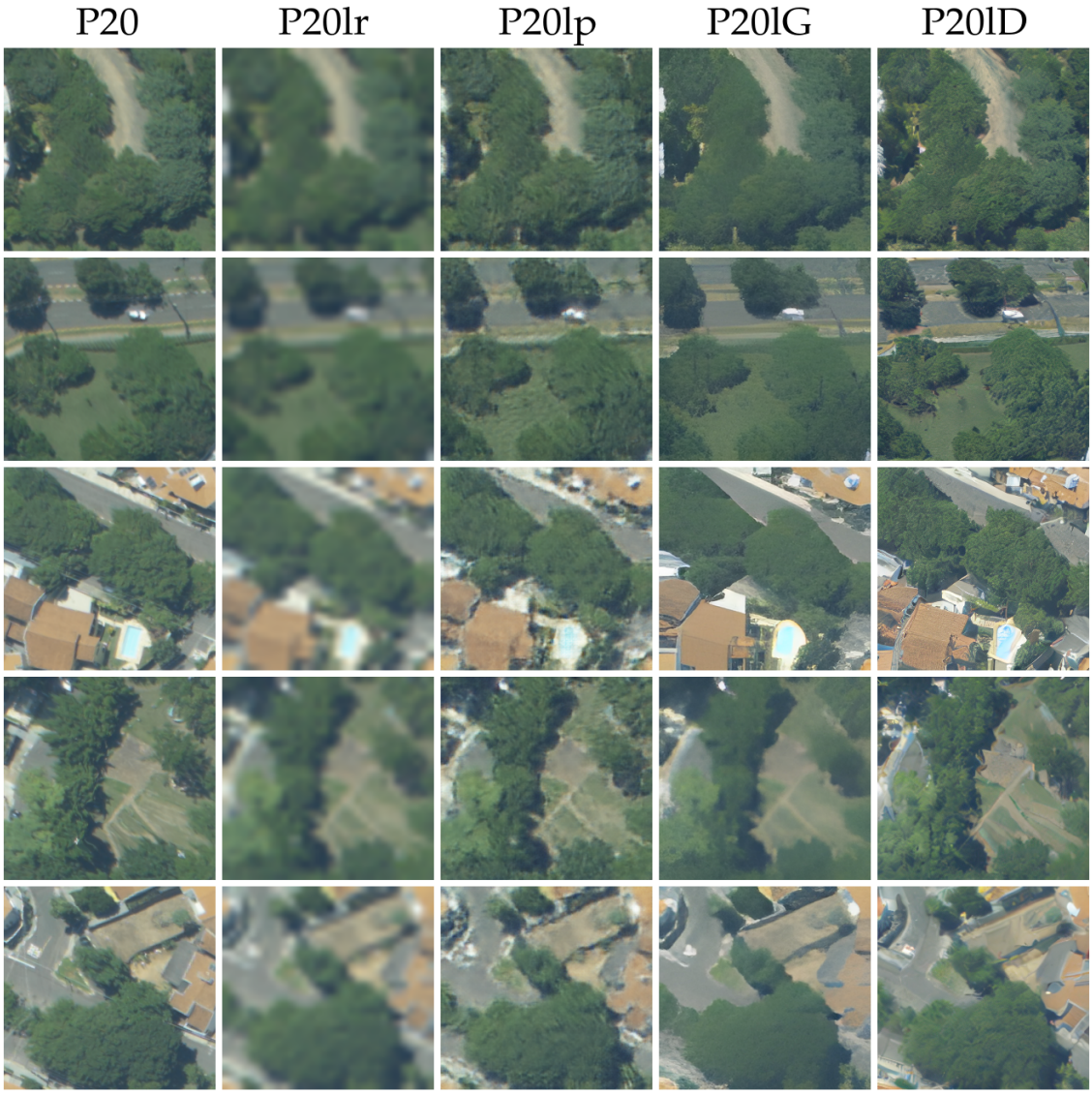

Figure 4.13: Sample images generated from low resolution dataset *P*20lr using pix2pix, Real-ESRGAN, and Latent Diffusion

the source model only approach, even without applying image translation or using super-resolution networks. One scenario not addressed in our experiments with these datasets is using our trained models with images of lower quality than those used in training.

We decided to simulate this scenario to evaluate the performance of the techniques presented here in enhancing the quality of low-resolution images. To simulate it, we resized the original 256×256 images from the *P*20 dataset to 32×32, decreasing their resolution by 8 times. This represents a difference significantly greater than the 2.5 times difference in our datasets.

Through this process, we created the dataset *P*20lr (*P20 low resolution*) and used it to test our GANs and Diffusion methods, creating new datasets with translated images. We generated the dataset *P*20lp after applying pix2pix translation, the dataset *P*20lG after increasing the resolution using Real-ESRGAN,

and the dataset *P*20lD after enhancing the resolution with Latent Diffusion. Examples of images from these datasets can be seen in Figure 4.13.

	P20 → P20lr	**P20 → P20lp**	**P20 → P20lG**	**P20 → P20lD**	**P20 → P20**
SegFormer (MiT-B5)					
Background	89.72	92.43	90.22	90.41	94.87
Trees	50.99	**67.80**	61.71	61.60	**77.44**
Average	70.36	80.11	75.97	76.00	86.15

Table 4.10: IoU of the src-only evaluation using the model trained with images from datasets *P*20 against low resolution and upscaled images using pix2pix, Real-ESRGAN, Latent Diffusion, and Stable Diffusion. In bold, the best result for the Trees class.

In Table 4.10, we present the IoU results of segmentation using our model trained with images from dataset *P*20. There is a noticeable decrease in performance when our model trained with original *P*20 images segments low-resolution images from database *P*20lr. However, when segmenting target images translated by the pix2pix model, this same model achieved significantly better results compared to those obtained using super-resolution models, despite the visually superior quality of images generated by Latent Diffusion, particularly evident in the depiction of roofs as shown in Figure 4.13.

This evaluation corroborates the idea that, for the approach used in this work, preserving the semantic information of original pixels is more crucial for segmentation results than achieving high visual quality in the generated images. However, it is important to acknowledge the capability of super-resolution models to generate coherent images from low-resolution inputs using a publicly available checkpoint without fine-tuning and the training process required by pix2pix models. The visual predictions, compared to the ground truth, can be seen in Figure 4.14.

4.6 Conclusion

In this chapter, we introduced an approach to enhance the resolution of aerial images to improve tree detection performance by utilizing image-to-image translation and super-resolution methods. Our method introduced a novel data augmentation technique, employing upsampling to generate high-quality annotated samples with varying ground sample distances (GSD). This approach also addresses the costly and labor-intensive process of manually labeling data.

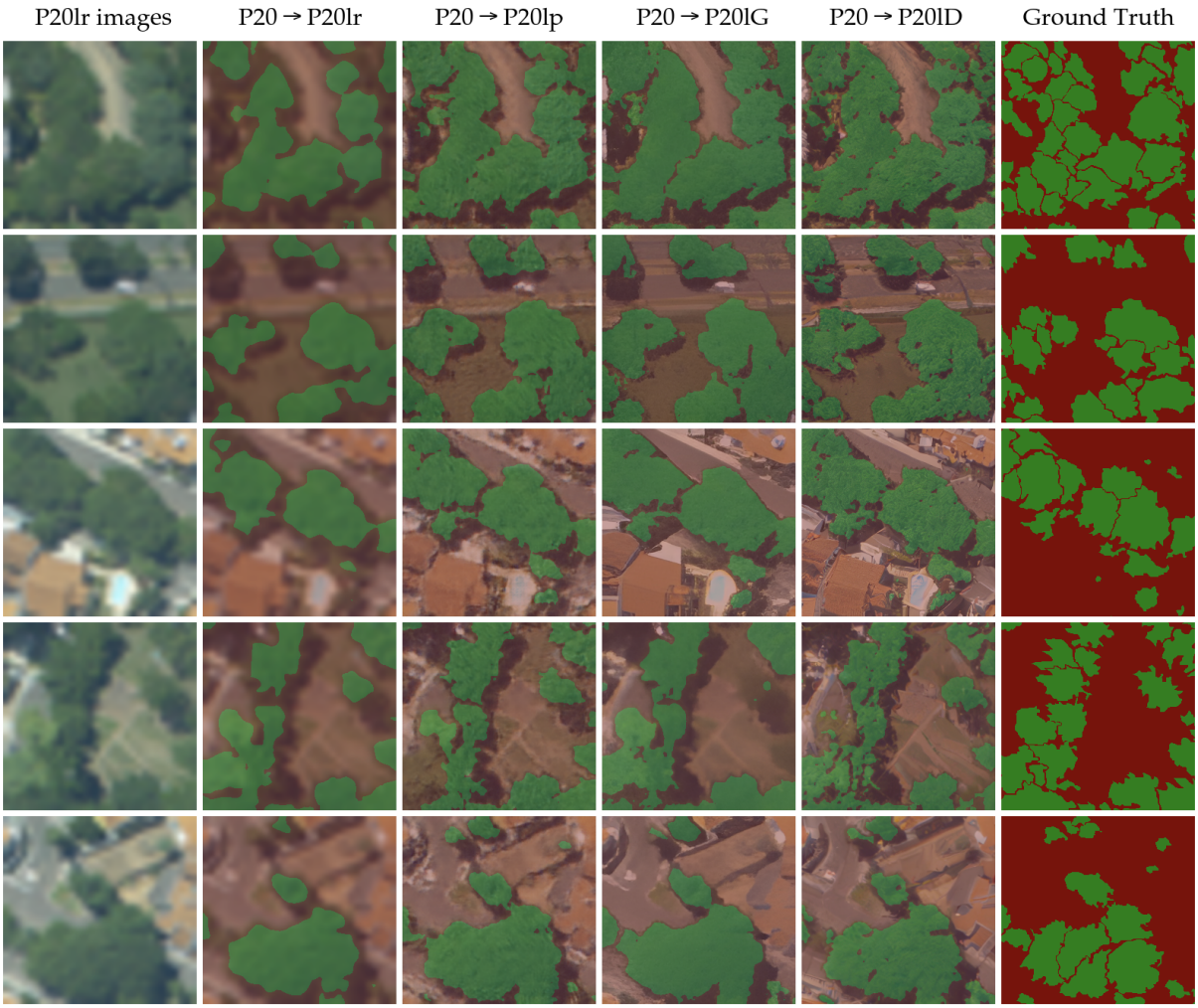

Figure 4.14: Predictions using the SegFormer model, trained with original images from dataset *P*20, in the low resolution images and their respective upscaled images using pix2pix, Real-ESRGAN, and Latent Diffusion.

Our data augmentation pipeline, which combines upsampling with translation and super-resolution steps, can be applied with different scaling factors to create new labeled images across a range of GSDs. This process enables the network to adapt to different image capture heights, thereby increasing the robustness of the supervised model when applied to new domains. Our evaluation revealed that lightweight models, such as pix2pix, can compete effectively with more recent and complex networks in translating images when trained appropriately.

In addition, we also conducted experiments reducing the resolution of our original dataset images, which were generally of high quality, by a factor of eight and evaluated the model's performance on both the original and enhanced images. The results demonstrated that our upsampling pipeline using pix2pix improved IoU tree detection performance by more than 50% when compared to the low-resolution images, validating the effectiveness of our up-

sampling strategy. The methods for enhancing resolution presented in this work can be applied in scenarios where remote sensing images lack the necessary quality for achieving high accuracy in computer vision tasks, such as detection, classification, and segmentation.

CHAPTER 5

Conclusion

5.1 Summary

In this work, we explored the use of domain adaptation to address challenges in agriculture and urban forests that require extensive annotated data. In this context, we investigated two problems: detecting sugarcane rows and gaps and segmenting trees in aerial images.

We proposed an approach to detect crop rows and gaps using semantic segmentation networks with semi-automatically generated ground truth. In our experiments, the transformer-based model, SegFormer, achieved performance equivalent to convolutional networks for detecting crop rows and gaps, but with better generalization to unseen data. The UDA model, DAFormer, performed better compared to SegFormer trained on source data only, proving to be an alternative in the absence of manually labeled data, a common scenario in agriculture.

Furthermore, Vision Transformers are proving to be a very promising method for computer vision. As a recent technique, it is expected that in the coming years, other semantic segmentation and unsupervised domain adaptation architectures will benefit from its robustness against source overfitting compared to convolutional networks, as analyzed in our discussion on generalization by epochs.

We also proposed a method that combines domain adaptation with image-to-image translation models and super-resolution networks for tree detection. Our approach evaluates recent super-resolution networks to enhance the quality of low-resolution aerial images. Additionally, our experiments us-

ing simulated low-resolution images demonstrated that the pix2pix model can significantly compete with these more powerful models when properly trained.

The data augmentation pipeline presented in this work offers an effective method for generating new annotated data for datasets with limited annotations. By adjusting the upsampling factor, we can simulate different ground sample distances, thereby creating images that mimic those captured at varying heights in aerial image datasets.

Finally, the findings presented here on unsupervised domain adaptation can be applied to similar agricultural and urban forest challenges in computer vision, such as weed detection or tree classification from UAV-captured images. These advancements can facilitate controlled experiments and address real-world issues more effectively.

5.2 Future Work

While we have made significant advancements in this research, there remains considerable potential for further exploration. Although our datasets addressed different factors that contribute to data shift, such as geographic location and capture height, several important variables have not been fully explored. Future studies could investigate the effects of illumination, acquisition angle, different sensors, and the phenological stages of vegetation, which were not covered in this research.

Testing our pipelines under these additional conditions could further validate the robustness of our methods and identify areas for improvement, which would help in developing a more resilient domain adaptation framework capable of handling greater data variability in real-world scenarios. Additionally, the methods we proposed could be integrated into existing remote sensing software as plugins or standalone applications. Such tools would assist specialists, reducing manual labor and improving the accuracy of their analyses.

Lastly, there are several promising areas for further exploration that were not fully addressed in this work. These include: (1) skeletonizing segmented images to represent rows and gaps as one-pixel-wide lines, (2) combining domain generalization with recently adopted masking techniques to capture shared domain characteristics, and (3) examining the influence of different prompts on the performance of the Stable Diffusion model. These extensions could provide valuable contributions to the field and broaden the applicability of our methods.

Bibliography

Ahonen, T., Hadid, A., e Pietikainen, M. (2006). Face description with local binary patterns: Application to face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 28(12):2037–2041. Citado na página 1.

Amirkolaee, H. A., Shi, M., He, L., e Mulligan, M. (2024). Adatreeformer: Few shot domain adaptation for tree counting from a single high-resolution image. *arXiv preprint arXiv:2402.02956*. Citado nas páginas 3 e 56.

Bah, M. D., Hafiane, A., e Canals, R. (2019). Crownet: Deep network for crop row detection in uav images. *IEEE Access*, 8:5189–5200. Citado nas páginas 27, 28, 38, e 39.

Beery, S., Wu, G., Edwards, T., Pavetic, F., Majewski, B., Mukherjee, S., Chan, S., Morgan, J., Rathod, V., e Huang, J. (2022). The auto arborist dataset: A large-scale benchmark for multiview urban forest monitoring under domain shift. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 21294–21307. Citado na página 39.

Beloiu, M., Heinzmann, L., Rehush, N., Gessler, A., e Griess, V. C. (2023). Individual tree-crown detection and species identification in heterogeneous forests using aerial rgb imagery and deep learning. *Remote Sensing*, 15(5):1463. Citado na página 55.

Chen, G. e Shang, Y. (2022). Transformer for tree counting in aerial images. *Remote Sensing*, 14(3):476. Citado na página 55.

Chen, P., Ma, X., Wang, F., e Li, J. (2021). A new method for crop row detection using unmanned aerial vehicle images. *Remote Sensing*, 13(17):3526. Citado na página 27.

Chudasama, D., Patel, T., Joshi, S., e Prajapati, G. I. (2015). Image segmentation using morphological operations. *International Journal of Computer Applications*, 117(18). Citado na página 28.

Contributors, M. (2020). Openmmlab semantic segmentation toolbox and benchmark. Citado na página 34.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., e Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 3213–3223. Citado nas páginas 18 e 60.

Cortes, C. e Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297. Citado na página 1.

Dalal, N. e Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, páginas 886–893. Citado na página 1.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., e Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, páginas 248–255. Ieee. Citado na página 16.

dos Santos Ferreira, A., Freitas, D. M., da Silva, G. G., Pistori, H., e Folhes, M. T. (2019). Unsupervised deep learning and semi-automatic data labeling in weed discrimination. *Computers and Electronics in Agriculture*, 165:104963. Citado nas páginas 2 e 56.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*. Citado nas páginas 16 e 28.

Duchon, C. E. (1979). Lanczos filtering in one and two dimensions. *Journal of Applied Meteorology and Climatology*, 18(8):1016–1022. Citado na página 58.

Ferreira, M. P., de Almeida, D. R. A., de Almeida Papa, D., Minervino, J. B. S., Veras, H. F. P., Formighieri, A., Santos, C. A. N., Ferreira, M. A. D., Figueiredo, E. O., e Ferreira, E. J. L. (2020). Individual tree detection and species classification of amazonian palms using uav images and deep learning. *Forest Ecology and Management*, 475:118397. Citado nas páginas 3 e 55.

Ganin, Y. e Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, páginas 1180–1189. PMLR. Citado nas páginas 2, 3, 7, 8, e 9.

García-Santillán, I. D., Montalvo, M., Guerrero, J. M., e Pajares, G. (2017). Automatic detection of curved and straight crop rows from images in maize fields. *Biosystems Engineering*, 156:61–79. Citado na página 27.

Giuffrida, M. V., Dobrescu, A., Doerner, P., e Tsaftaris, S. A. (2019). Leaf counting without annotations using adversarial unsupervised domain adaptation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, páginas 2590–2599. IEEE. Citado nas páginas 2 e 7.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., e Bengio, Y. (2014). Generative adversarial nets. In *2014 Advances in neural information processing systems*, páginas 2672–2680. Citado na página 9.

Gretton., A., Smola., A. J., Huang, J., Schmittfull, M., Borgwardt., K. M., e Scholkopf, B. (2009). *Covariate shift and local learning by distribution matching*, páginas 131–160. MIT Press. Citado na página 7.

He, K., Zhang, X., Ren, S., e Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 770–778. Citado na página 20.

Ho, J., Jain, A., e Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851. Citado na página 23.

Hoffman, J., Tzeng, E., Park, T., Zhu, J. Y., Isola, P., Saenko, K., e Darrell, B. (2018). Cycada: Cycle-consistent adversarial domain adaptation. In *2018 International conference on machine learning*, páginas 1989–1998. Citado na página 14.

Hough, P. V. (1962). Method and means for recognizing complex patterns. *US patent*, 3(6). Citado na página 27.

Hoyer, L., Dai, D., e Van Gool, L. (2021). Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentation. *arXiv preprint arXiv:2111.14887*. Citado nas páginas 18, 19, 29, e 40.

Huang, Z., Wang, X., Wang, J., Liu, W., e Wang, J. (2018). Weakly-supervised semantic segmentation network with deep seeded region growing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 7014–7023. Citado na página 29.

Iqbal, M. S., Ali, H., Tran, S. N., e Iqbal, T. (2021). Coconut trees detection and segmentation in aerial imagery using mask region-based convolution neural network. *IET Computer Vision*, 15(6):428–439. Citado na página 55.

Isola, P., Zhu, J.-Y., Zhou, T., e Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 1125–1134. Citado nas páginas 12, 14, e 15.

Jain, A. K., Mao, J., e Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, 29(3):31–44. Citado na página 1.

Jiang, G., Wang, Z., e Liu, H. (2015). Automatic detection of crop rows based on multi-rois. *Expert systems with applications*, 42(5):2429–2441. Citado na página 27.

Jintasuttisak, T., Edirisinghe, E., e Elbattay, A. (2022). Deep neural network based date palm tree detection in drone imagery. *Computers and Electronics in Agriculture*, 192:106560. Citado na página 55.

Kamilaris, A. e Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and electronics in agriculture*, 147:70–90. Citado nas páginas 2 e 3.

Kapil, R., Marvasti-Zadeh, S. M., Erbilgin, N., e Ray, N. (2024). Shadowsense: Unsupervised domain adaptation and feature fusion for shadow-agnostic tree crown detection from rgb-thermal drone imagery. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, páginas 8266–8276. Citado nas páginas 3 e 56.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., e Shah, M. (2021). Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*. Citado na página 16.

Kingma, D. P. e Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. Citado nas páginas 13 e 23.

LeCun, Y., Bengio, Y., e Hinton, G. (2015). Deep learning. *nature*, 521(7553):436. Citado na página 2.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 4681–4690. Citado na página 20.

Liang, J., Hu, D., e Feng, J. (2020). Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, páginas 6028–6039. PMLR. Citado na página 29.

Liaw, A. e Wiener, M. (2002). Classification and regression by randomforest. *R news*, 2(3):18–22. Citado na página 1.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110. Citado na página 1.

Lv, L., Li, X., Mao, F., Zhou, L., Xuan, J., Zhao, Y., Yu, J., Song, M., Huang, L., e Du, H. (2023). A deep learning network for individual tree segmentation in uav images with a coupled cspnet and attention mechanism. *Remote Sensing*, 15(18):4420. Citado nas páginas 3 e 55.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., e Frey, B. (2015). Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*. Citado na página 14.

Mnih, V. e Hinton, G. E. (2012). Learning to label aerial images from noisy data. In *Proceedings of the 29th International conference on machine learning (ICML-12)*, páginas 567–574. Citado na página 38.

Osco, L. P., de Arruda, M. d. S., Gonçalves, D. N., Dias, A., Batistoti, J., de Souza, M., Gomes, F. D. G., Ramos, A. P. M., de Castro Jorge, L. A., Liesenberg, V., et al. (2021). A cnn approach to simultaneously count plants and detect plantation-rows from uav imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 174:1–17. Citado na página 27.

Richter, S. R., Vineet, V., Roth, S., e Koltun, V. (2016). Playing for data: Ground truth from computer games. In *European conference on computer vision*, páginas 102–118. Springer. Citado na página 18.

Rocha, B. M., da Fonseca, A. U., Pedrini, H., e Soares, F. (2022). Automatic detection and evaluation of sugarcane planting rows in aerial images. *Information Processing in Agriculture*. Citado na página 27.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., e Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, páginas 10684–10695. Citado na página 24.

Ronneberger, O., Fischer, P., e Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference,*

Munich, Germany, October 5-9, 2015, proceedings, part III 18, páginas 234–241. Springer. Citado nas páginas 22 e 24.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. (2022). Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294. Citado na página 25.

Simonyan, K. e Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. Citado na página 20.

Soares, G. A., Abdala, D. D., e Escarpinati, M. C. (2018). Plantation rows identification by means of image tiling and hough transform. In *VISIGRAPP (4: VISAPP)*, páginas 453–459. Citado na página 27.

Soh, L. K. e Tsatsoulis, C. (1999). Texture analysis of sar sea ice imagery using gray level co-occurrence matrices. *IEEE Transactions on geoscience and remote sensing*, 37(2):780–795. Citado na página 1.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., e Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, páginas 2256–2265. PMLR. Citado na página 22.

Song, J., Meng, C., e Ermon, S. (2020). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*. Citado nas páginas 22 e 24.

Still, M. (2006). *The definitive guide to ImageMagick*. Apress. Citado na página 58.

Sun, C., Shrivastava, A., Singh, S., e Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, páginas 843–852. Citado na página 16.

Tuia, D., Persello, C., e Bruzzone, L. (2016). Domain adaptation for the classification of remote sensing data: An overview of recent advances. *IEEE geoscience and remote sensing magazine*, 4(2):41–57. Citado nas páginas 3 e 4.

Tuia, D., Persello, C., e Bruzzone, L. (2021). Recent advances in domain adaptation for the classification of remote sensing data. *arXiv preprint arXiv:2104.07778*. Citado na página 56.

Tyleček, R. e Šára, R. (2013). Spatial pattern templates for recognition of objects with regular structure. In *Pattern Recognition: 35th German Conference, GCPR 2013, Saarbrücken, Germany, September 3-6, 2013. Proceedings 35*, páginas 364–374. Springer. Citado na página 60.

Tzeng, E., Hoffman, J., Saenko, K., e Darrell, B. (2017). Adversarial discriminative domain adaptation. In *2017 Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 7167–7176. Citado na página 11.

Tzeng, E. e Tuzel, O. (2016). Coupled generative adversarial networks. In *2016 Advances in neural information processing systems*, páginas 469–477. Citado na página 10.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., e Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30. Citado nas páginas 16, 25, e 28.

Velasquez-Camacho, L., Etxegarai, M., e de Miguel, S. (2023). Implementing deep learning algorithms for urban tree detection and geolocation with high-resolution aerial, satellite, and ground-level images. *Computers, Environment and Urban Systems*, 105:102025. Citado nas páginas 3 e 55.

Ventura, J., Pawlak, C., Honsberger, M., Gonsalves, C., Rice, J., Love, N. L., Han, S., Nguyen, V., Sugano, K., Doremus, J., et al. (2024). Individual tree detection in large-scale urban environments using high-resolution multispectral imagery. *International Journal of Applied Earth Observation and Geoinformation*, 130:103848. Citado na página 55.

Vezhnevets, A., Ferrari, V., e Buhmann, J. M. (2011). Weakly supervised semantic segmentation with a multi-image model. In *2011 international conference on computer vision*, páginas 643–650. IEEE. Citado na página 29.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., e Shao, L. (2021a). Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, páginas 568–578. Citado na página 17.

Wang, X., Xie, L., Dong, C., e Shan, Y. (2021b). Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF international conference on computer vision*, páginas 1905–1914. Citado nas páginas 21 e 22.

Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., e Change Loy, C. (2018). Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, páginas 0–0. Citado nas páginas 20 e 21.

Wang, Y., Yang, G., e Lu, H. (2022). Domain adaptive tree crown detection using high-resolution remote sensing images. *Journal of Applied Remote Sensing*, 16(4):044505–044505. Citado na página 56.

Wei, X., Lv, X., e Zhang, K. (2021). Road extraction in sar images using ordinal regression and road-topology loss. *Remote Sensing*, 13(11):2080. Citado na página 38.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., e Luo, P. (2021). Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34. Citado nas páginas 17, 29, e 40.

Yang, J., Shi, S., Wang, Z., Li, H., e Qi, X. (2021). St3d: Self-training for unsupervised domain adaptation on 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 10368–10378. Citado na página 45.

Zheng, J., Fu, H., Li, W., Wu, W., Zhao, Y., Dong, R., e Yu, L. (2020). Cross-regional oil palm tree counting and detection via a multi-level attention domain adaptation network. *ISPRS Journal of Photogrammetry and Remote Sensing*, 167:154–177. Citado nas páginas 2 e 56.

Zhu, J. Y., Park, T., Isola, P., e Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 Proceedings of the IEEE international conference on computer vision*, páginas 2223–2232. Citado nas páginas 12, 13, e 14.