Multi-task Learning Applied to Computer Vision Problems

Diogo Nunes Gonçalves

SERVIÇO DE PÓS-GRADUAÇÃO DO FACOM-UFMS

Data de Depósito:

Assinatura:_____

Multi-task Learning Applied to Computer Vision Problems[1]

Diogo Nunes Gonçalves

Orientador: *Professor Doutor Hemerson Pistori*

Tese apresentada à Faculdade de Computação - FACOM-UFMS - como parte dos requisitos necessários à obtenção do título de Doutor em Ciências da Computação.

UFMS - Campo Grande
março/2022

[1]Trabalho Realizado com Auxílio da CAPES

Aos meus pais,
Osmar e Marlene,

Aos meus irmãos,
Pâmela e Wesley,

À Isabella Vitorete.

Agradecimentos

Primeiro agradeço a Deus por sempre estar ao meu lado, me concedendo o privilégio de sentir a sua presença todos os dias da minha vida, pela sua misericórdia e por colocar pessoas tão boas e especiais em meu caminho.

Agradeço aos meus pais Osmar e Marlene por me ensinarem sobre todas as coisas da vida, me apoiando e ensinando o caminho correto, tanto na parte espiritual quanto na material. Agradeço aos meus irmãos e cunhados Wesley e Jociane, Pâmela e Anderson por serem os melhores irmãos e cunhados que alguém pode ter, me apoiando em todos os sentidos da minha vida. Agradeço também a minha sobrinha Júlia.

Agradeço a minha namorada, Isabella Vitorete, por todo o apoio e amor que teve comigo durante todos esses anos, me aconselhando e ajudando nos momentos difíceis. Agradeço também o apoio e ensinamentos dos meus sogros Vanderlei e Vera.

Agradeço ao meu orientador Hemerson Pistori, que esteve comigo durante toda a minha pós-graduação sendo o meu orientador no mestrado e doutorado, por tudo que tem me ensinado e por ser exemplo de dedicação e ensino na formação de pessoas e pesquisadores. Agradeço ao nosso grupo de pesquisa INOVISÃO pelos encontros que tivemos durante esse período.

Finalmente gostaria de agradecer à CAPES pela minha bolsa de doutorado e a FACOM-UFMS, pelo suporte e estrutura disponibilizados para o desenvolvimento de minha formação.

Abstract

Deep learning has been widely studied, mainly to solve problems considered complex. In general, these problems can be described and divided into a set of tasks. These tasks are intrinsic to the general problem, that is, they are naturally defined because they are part of the essence of the problem. In addition, they can be learned in isolation but are related to the solution of the general problem. Another important factor is that for a larger computer vision problem, performing the distinct tasks individually becomes expensive in memory and inference time. To solve these problems, several approaches as Multi-task Learning (MTL) was proposed. The idea is to simulate human learning, in which people can learn new tasks through experiences gained in learning similar tasks. This approach allows the learning of the tasks simultaneously, building a relationship between them. From these directions, this work in the form of a collection of articles presents MTL approaches for solving computer vision problems. Initially, two problems were addressed: detection of plantation lines in the first article and detection of fingerlings in the second. In the detection of plantation lines the idea is to divide the problem into identifying the plants individually and detecting the plantation lines. In fingerling detection, the tasks are divided into detecting the fingerling and identifying the fingerling direction in subsequent frames. For both problems, a method was proposed with a backbone that extracts the initial features for all tasks. Taking the initial features as input, independent branches learn the solution of each task. The exchange of information between tasks occurs through the concatenation of features extracted at specific points in each branch. The results showed that sharing between tasks is important for the solution, achieving results superior to the state-of-the-art. In addition to the two proposals, a new semantic segmentation method using MTL and attention mechanism was proposed. The main advance was the use of weights learned by Transformers to indicate the importance of a task to others. Thus, only image regions considered relevant influence other tasks. Results on two problems, plantation

line and gaps, and leaf segmentation and defoliation, showed the effectiveness of the approach compared to the state-of-the-art.

Resumo

O aprendizado profundo tem sido amplamente estudado, principalmente para resolver problemas considerados complexos. De forma geral, esses problemas podem ser descritos e divididos em um conjunto de tarefas. Essas tarefas são intrínsecas ao problema geral, ou seja, são definidas de forma natural por fazer parte da essência do problema. Além disso, elas podem ser aprendidas de forma isolada porém estão relacionadas para solução do problema geral. Outro fator importante é que para um problema maior de visão computacional, realizar as tarefas distintas individualmente se torna muito custoso em memória e tempo de inferência. Para solucionar esses problemas uma abordagem chamada Aprendizado Multitarefa (MTL) foi proposta. A ideia é simular o aprendizado humano, em que pessoas podem aprender novas tarefas através de experiências obtidas no aprendizado de tarefas similares. Essa abordagem permite o aprendizado das tarefas do problema de maneira simultânea, construindo uma relação entre elas. A partir desses direcionamentos, este trabalho na forma de coleção de artigos apresenta abordagens MTL para a resolução de problemas de visão computacional. Inicialmente, dois problemas foram abordados: detecção de linhas de plantação no primeiro artigo e detecção de alevinos no segundo. Na detecção de linhas de plantação a ideia é dividir o problema em identificar as plantas individualmente e detectar as linhas de plantação. Na detecção de alevinos as tarefas são divididas em detectar o alevino e identificar a direção desse alevino nos quadros subsequentes. Para ambos os problemas, um método foi proposto com um backbone que extrai as características iniciais para todas as tarefas. Tendo como entrada as características iniciais, ramos independentes aprendem a solução de cada tarefa. A troca de informações entre as tarefas ocorre por meio da concatenação das características extraídas em pontos específicos de cada ramo. Os resultados mostraram que o compartilhamento entre as tarefas é importante para a solução, alcançando resultados superiores ao estado-da-arte. Além das duas propostas, um novo método de segmentação semântica

usando MTL e mecanismo de atenção foi proposto. O principal avanço foi o uso de pesos aprendidos por Transformers para indicar a importância de uma tarefa nas demais. Assim, apenas regiões da imagem consideradas relevantes influenciam em outras tarefas. Os resultados em dois problemas, segmentação de linhas e falhas de plantação, e segmentação da folha e desfolha, mostraram a eficácia da abordagem frente ao estado-da-arte.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

Deep Learning, Convolutional neural networks (CNNs) and, more recently, transformers have been widely studied in the literature due to the great advances obtained in computer vision problems, such as classification (Dosovitskiy et al., 2021; Pan et al., 2020) and image segmentation (Gonçalves et al., 2020a; Osco et al., 2021a; Lobo Torres et al., 2020). These computer vision problems can be described as a set of tasks that are defined intrinsically to the problem. In general, CNNs perform one task at a time, which for larger computer vision problems becomes expensive in memory and inference time. This is because it is usually necessary for these problems to be divided into smaller parts to be solved by these networks. However, studies have discussed whether this division leads to a loss of generalization (Crawshaw, 2020; Zhang and Yang, 2021).

To solve these problems, an approach called Multi-task Learning (MTL) (Caruana, 1998) has been proposed. This idea was inspired by human learning, where people can learn new tasks through experiences gained in learning similar or correlated tasks. Unlike transfer learning (Zhuang et al., 2021), these tasks are learned simultaneously. Therefore, this approach allows simultaneous learning of multiple tasks that may have some relationship with each other. Several challenges are faced in the implementation of this approach, the two main ones being: building the CNN architecture and updating the weights considering simultaneous tasks.

- **Building the network architecture**: the challenge is to establish how

CNN is built to allow the exchange of learning between the multiple tasks addressed by the problem.

- **Weights Update**: the challenge is to find weights for the layers that indicate the relationship between the multiple tasks.

This work aims to propose methods using MTL and deep learning. Given a problem, the proposal is to learn two or more tasks, that is, to define tasks that help in solving the general problem. This work initially addresses two applications: plantation line detection and fingerling detection. In the detection of plantation lines, the idea is to divide the problem into two tasks: identify the plants individually and detect the plantation lines. We can observe that the identification of individual plants can help in the detection of the plantation line, because the lines are composed of plants. Likewise, detecting a line assists in detecting individual plants as, in general, plants tend to be located only in the plantation line. In fingerling detection, the tasks are divided into detecting the fingerling and identifying the direction of movement in subsequent frames. Identifying a fingerling (first task) facilitates the identification of its movement direction (second task), as well as identifying its movement direction facilitates the detection of individual fingerlings in the next frame. The architecture proposed in these works allows the information of tasks extracted by different branches of CNN to be shared with each other through concatenation at specific points. This exchange of information between the tasks of the two problems and the results are further explored in Chapters 2 and 3.

Considering recent advances in attention mechanisms, we also propose a semantic segmentation method for MTL based on Transformers in Chapter 4. The problems consist of: i) segmenting plantation lines and gaps (sequence of the line without plants) and ii) segmenting a main leaf and its defoliation (area absent due to pests). The main challenge of the first problem is to segment the gaps, as their visual characteristics are similar to the background and, in fact, a gap can only be identified if the plantation lines are identified properly. The second problem consists of segmenting a leaf in the foreground and its absent area caused by pests, called defoliation. Defoliation segmentation can only be performed properly with knowledge of the leaf shape, especially when defoliation occurs at the leaf edges. This shows that the problems are composed of correlated tasks that benefit from information sharing. For sharing, an attention mechanism was proposed in which each region of the image of a specific task can directly influence the features of other tasks through a weight. This sharing proved to be effective mainly in tasks that depend on others, as in the aforementioned problems.

1.2 Related Works

Several works on MTL have been proposed in the literature (Caruana, 1998; Doersch and Zisserman, 2017). The first works studied methods of sharing features between tasks using different classification models (Kumar and Daumé, 2012; Xue et al., 2007; Passos et al., 2012). Ruder (2017) presented an overview of MTL in deep learning. This work showed that most proposals in the literature have two MTL approaches: hard and soft parameter sharing. Figure 1.1 presents the hard parameter sharing approach. In this approach, all tasks have a shared backbone followed by a branch for each task, and there is no more sharing between these branches. Figure 1.2 presents the soft parameter sharing approach. In this case, each task has independent layers, but information is shared between the branches of the tasks.

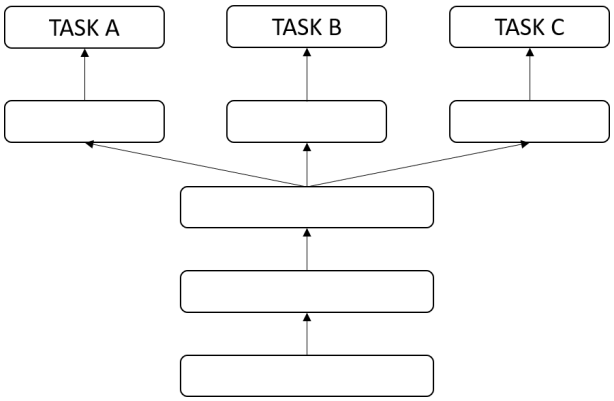


Figure 1.1: Illustration of the hard parameter sharing approach.

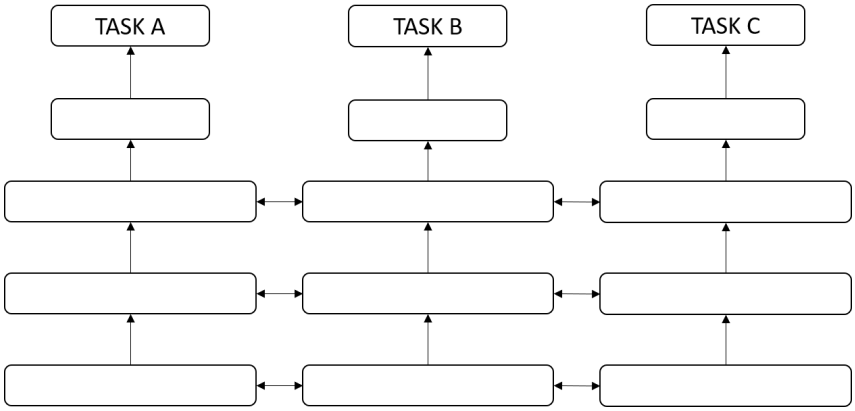


Figure 1.2: Illustration of the soft parameter sharing approach.

Hard Parameter Sharing

In this approach, the sharing of parameters takes place in the hidden layers of the CNN between the tasks of the problem. Several works have been proposed in the literature using this approach. Ranjan et al. (2019) presented

a work for simultaneous face detection, landmark location, pose estimation and gender recognition called HyperFace. The main idea is to present a CNN to learn the common features of these tasks and explore the similarities between them. Finally, they built a new CNN that shares some layers to learn the features of the tasks through their similarities.

Jou and Chang (2016) proposed a residual learning extension that allows the learning of several tasks at the same time. For this, connections between these networks called cross-residual learning (CRL) were proposed, which can be seen as a form of regularization of the network and allow a greater generalization. The authors indicated that CRL can be integrated into multi-task CNNs. Dvornik et al. (2017) presented a CNN called BlitzNet to perform object detection and semantic segmentation at the same time. They showed that, when trained simultaneously, each task benefits from the other, in addition to the computational gain of having a single network to perform both tasks.

Other MTL proposals have addressed multi-branch architecture learning, using greedy optimization to measure task affinity, such as Lu et al. (2017) and Vandenhende et al. (2020). Lu et al. (2017) proposed an MTL approach to design multi-task deep learning architectures. The idea is to start with a small network and dynamically scale up greedily during training, using a cluster of similar tasks as a criterion. Using the same multi-branch approach, Vandenhende et al. (2020) presented an approach to automatically build multi-task networks using the affinities of the tasks. This approach generates architectures, in which the first layers learn more general features of the tasks, while the deeper ones learn more specific features.

Finally, methods used convolutional filter clustering to create multi-task networks, such as the work of Bragman et al. (2019). They presented a probabilistic approach to learning the shared and specific features of tasks in CNNs. The idea was to propose stochastic filter groups (SFG) that assign convolution kernels in each layer to specialist or generalist groups, that is, that are specific or shared by the tasks of the problem. These modules determine the connectivity between the layers and the network structure between the tasks, that is, how much information is exchanged between the tasks. All these works described above use the hard parameter sharing approach as they share the learning of the hidden layers of the CNNs among the different tasks of the problem.

Soft Parameter Sharing

This approach consists of dividing each task into columns, where each column contains its own parameters and model. Unlike the hard parameter sharing approach, there is no exchange of information between the deep hid-

den layers of the network, so it is necessary to define a way of sharing between these columns, which in general, consists of a backbone that allows the initial exchange of information. Several works have been proposed in the literature using this approach such as Cross-stitch (Misra et al., 2016)], Sluice (Ruder et al., 2019) and NDDR (Gao et al., 2019).

Misra et al. (2016) presented Cross-stitch which is an approach to learning ways to share between tasks. They proposed a new sharing unit called cross-stitch unit that combines the activations of several CNNs. The idea is that a network with these units can learn an optimal sharing combination between tasks. Ruder et al. (2019) also developed an approach that learns a multi-task architecture. They showed that multi-task learning involves searching a huge space of possibilities for sharing architectures, which becomes complex. Therefore, they proposed a new architecture that learns: which layers to share between CNNs, which parts of the layers are best to share, and how much of those layers are shared. Gao et al. (2019) proposed a new CNN framework for multi-task that allows automatic merging of features in each layer of different tasks. This automatic fusion occurs in the layers and occurs through a combination of the existing CNN components, but in a new way, aiming at a reduction of discriminative dimensionality. This reduction was called Neural Discriminative Dimensionality Reduction (NDDR).

Other approaches that are being proposed in the literature are attention-based methods (Attention-based methods). Liu et al. (2019) developed a new multi-task learning architecture that allows learning the level of attention for each feature of a given task. The idea is to build a Multi-task Attention Network (MTAN) that consists of a single shared network containing a set of global features, along with what they called a soft attention module for each task in the problem. These modules allow the learning of the specific features of each task from the global features and also allows the sharing of these features between the tasks.

1.3 Objectives

From these directions, the general objective of this work is the development of methods using MTL in deep learning and its application in relevant problems using images.

To achieve the general objective, the following specific objectives were defined:

- Build image datasets of the problems to be applied, such as plantation lines and fingerlings;

- Propose and implement methods for sharing features of tasks involving images;
- Extend the sharing proposal to temporal tasks composed of a sequence of images.
- Propose and implement a method combining transformers and Multi-task learning;
- Validate the proposals comparing with methods presented in the literature.

1.4 Structure of the Thesis

The rest of this work is described below. The presentation is a collection of articles, in which the general objective is achieved through the specific objectives of each article. The papers present methods based on multi-task learning. The proposal combines the two approaches: soft and hard parameter sharing. Soft parameter sharing occurs because there are independent branches for tasks. In addition, the methods also perform hard parameter sharing because an exchange of information between these branches takes place in deep layers. Thus, the methods proposed in this work benefit from the advantages of both approaches to multi-task learning. Chapter 2 presents the first article with the proposal for the detection of plantation lines. Chapter 3 describes the work for fingerling detection where temporal information is considered. Chapter 4 presents a new approach that combines Transformers and MTL for semantic segmentation. Finally, Chapter 5 presents final considerations and future work.

CHAPTER

2

A Deep Learning Approach Based on Graphs to Detect Plantation-Lines

Preface

Identifying plantation lines in aerial images of agricultural landscapes is required for many automatic farming processes. Deep learning-based networks are among the most prominent methods to learn such patterns and extract this type of information from diverse imagery conditions. However, even state-of-the-art methods may stumble in complex plantation patterns. Here, we propose a deep learning approach based on graphs to detect plantation lines in UAV-based RGB imagery, presenting a challenging scenario containing spaced plants. The first module of our method extracts a feature map throughout the backbone, which consists of the initial layers of the VGG16. This feature map is used as an input to the Knowledge Estimation Module (KEM), organized in three concatenated branches for detecting 1) the plant positions, 2) the plantation lines, and 3) for the displacement vectors between the plants. A graph modeling is applied considering each plant position on the image as vertices, and edges are formed between two vertices (i.e. plants). Finally, the edge is classified as pertaining to a certain plantation line based on three probabilities (higher than 0.5): i) in visual features obtained from the backbone; ii) a chance that the edge pixels belong to a line, from the KEM step; and iii) an alignment of the displacement vectors with the edge, also from the KEM step. Experiments were conducted initially in corn plantations with different growth stages and patterns with aerial RGB imagery to present the

advantages of adopting each module. We assessed the generalization capability in the other two cultures (orange and eucalyptus) datasets. The proposed method was compared against state-of-the-art deep learning methods, and achieved superior performance with a significant margin considering all three datasets. This approach is useful in extracting lines with spaced plantation patterns and could be implemented in scenarios where plantation gaps occur, generating lines with few-to-none interruptions.

Relation of the paper with MTL. The method proposed in this first paper has a shared backbone between the tasks to extract a feature map. Given the backbone feature map, each task is learned in an independent branch. As described in the literature review, this architecture belongs to the category of hard parameter sharing. However, there is an exchange of information between tasks through the concatenation of feature maps in specific layers of branches, which characterizes the category of soft parameter sharing. Therefore, the architecture of the proposed method can be considered as hybrid parameter sharing, reducing the computational cost with the shared backbone but keeping the information sharing between tasks in deep layers.

2.1 Introduction

Linear objects, also denominated linear features in the photogrammetric context, are common in images, especially in anthropic scenes. Consequently, they are used in several photogrammetric tasks, such as orientation or triangulation (Kubik, 1991; Schenk, 2004; Tommaselli and Junior, 2012; Marcato Junior and Tommaselli, 2013; Sun et al., 2019; Yavari et al., 2018), rectification (Li and Shi, 2014; Long et al., 2015b), matching (Wei et al., 2021), restitution (Lee and Bethel, 2004), and camera calibration (Ravi et al., 2018; Babapour et al., 2017). The registration of images and LiDAR (Light Detection And Ranging) data is also a topic that benefits from this type of object (Habib et al., 2005) (Yang and Chen, 2015). Previous works proposed several approaches to automatically detect lines in images based on traditional digital image processing techniques. But, these approaches usually require a significant number of parameters and are not always robust when dealing with challenging situations, including shadows, pixel-pattern, geometry, among others.

In recent years, artificial intelligence methods, especially those based on deep learning, have been adapted to process remote sensing images from several spatial-spectral-resolution traits, aiming to attend distinct application areas, including agriculture and farming (Hasan et al., 2021; Osco et al., 2020b; Kamilaris and Prenafeta-Boldú, 2018; Ramos et al., 2020). Deep learning-

based methods are state-of-the-art, well-known for their ability to deal with challenging and varied tasks, involving scene-wise recognition, object detection, and semantic segmentation problems (Osco et al., 2021b). For each of these problems-domain specifics, several attempts have been made, and great results were found. As such, deep neural networks (DNN) are quickly becoming one of the most prominent paths to learn and extract information from remote sensing data. This is mainly because it is difficult for the same method to evaluate different domains with the same performance, while deep learning developments aim to produce intelligent and robust mechanisms to deal with multiple learning patterns.

Based on recent literature analysis in the remote sensing field, few studies were developed focusing on deep learning to detect linear objects. Some deep networks based on segmentation approaches were proposed for line pattern detection, being mostly of them in road and watercourse extraction of aerial or orbital imagery. Yang et al. (2019), for example, developed a multitask learning method to segment roads simultaneously and to detect their respective centerlines. Their framework was based on recurrent neural networks and the U-Net method (Ronneberger et al., 2015a). A more recent publication (Wei et al., 2020) proposed an innovative solution to segment and detect road centerlines. Similarly, semantic segmentation approaches were developed in environmental applications with linear patterns, like river margin extraction in remote sensing imagery. An investigation (Xia et al., 2019) proposed a deep network adopting the ResNet (He et al., 2016) as the backbone of their framework for river segmentation in orbital images of medium-resolution. Another study (Weng et al., 2020) presented a separable residual SegNet (Badrinarayanan et al., 2017) method to segment rivers in remote sensing images, showing significant improvements over other deep learning-based approaches, including FCN (Long et al., 2015a) and DeconvNet (Noh et al., 2015). Wei et al. (2020) developed a semantic distance-based segmentation approach to extract rivers in images obtaining an F1-score superior to 93%, which outperformed several state-of-the-art algorithms.

In agricultural applications, a previous related-work (Osco et al., 2021a) proposed a method to simultaneously detect plants and plantation lines in the agriculture field using UAV (Unmanned Aerial Vehicle) imagery datasets through deep learning algorithms. However, for this task, only visual features of the plants and plantation lines were considered by the DNN algorithm. Consequently, the plants' locations (i.e., points) from different plantation lines were considered, in some situations, as belonging to the same line due to their proximity. This, however, indicated a limited potential of this approach mainly when gaps or adverse patterns in the plantations occurred. In other

agricultural-related remote sensing tasks, Rosa et al. (2020) proposed an approach based on semantic segmentation associated with geometric features to detect citrus plantation lines. Still, segmentation-based methods are not adequate to deal with spaced plants (non-continuous objects), which is the case for most crops in the initial stage. Moreover, another problem is when plantation gaps occur in later stages, wherein, for instance, plants are removed due to diseases or environmental hazards (e.g., strong winds). Additionally, line extraction, when associated with gap detection, is essential to conduct the replanting process, minimizing the losses in the cultivars, but that still remains an unsolved question inside both remote sensing and agricultural contexts supported by deep learning approaches.

A potential alternative that may support the aforementioned issues regarding differences in patterns and space between the objects (e.g., plants, for instance) is the adoption of graph theory at the learning and extraction processes. Graphs are a type of structure that considers that some pairs of objects are related to a given feature or real-world space scene. Therefore, they can be useful for representing the relationship between objects in multiple domains and can even inherit complicated structures containing rich underlying values (Zhang et al., 2022). As such, recent deep learning-based approaches have been proposed to evaluate or implement graph patterns for distinct problems-domain. Some of these approaches include strategies related to graph convolutional and/or recurrent neural networks, graph autoencoders, graph reinforcement learning, graph adversarial methods, and others (Zhang et al., 2022). Since graphs work by representing both the domain concept and their relationships, it makes them an innovative approach for improving the inference ability of objects in remote sensing imagery. Therefore, the combination of graph reasoning with the deep learning capability may work as complementary advantages of both techniques.

It is worth mentioning that few recent investigations integrating graphs in deep network models were conducted in the remote sensing and agriculture domains. One of which demonstrated the potential of implementing a semantic segmentation network with a graph convolutional neural network (CNN) to perform the segmentation of urban aerial imagery, identifying features like vegetation, pavement, buildings, water, vehicles, as others (Ouyang and Li, 2021). Another study (Ma et al., 2019) used an attention graph convolution network to segment land covers from SAR imagery, which demonstrated its high potential. A graph convolutional network was also used in a scene-wise classification task (Gao et al., 2021), discriminating between varied scenes from publicly available repositories containing images from several examples of land cover. In the hyperspectral domain, one approach (Hong et al., 2021)

was capable of successfully presenting a graph convolutional network-based method to pixel-wise classify differential land cover in urban environments. Furthermore, in urban areas, a graph convolutional neural network was investigated to classify building patterns using spatial vector data (Yan et al., 2019). In the agricultural context, a cross-attention mechanism was adopted with a graph convolution network (Cai and Wei, 2022) to separate (scene-wise classification) different crops, such as soybeans, corn, wheat, wood, hay, and others. The results were compared against state-of-the-art deep learning networks, outperforming them.

Regardless, up to the time of writing, no single-step approach to a DNN architecture was proposed with the integration of graphs to solve issues related to the identification and refinement of a plantation-line position in remote sensing imagery. In this paper, we propose a novel deep learning method based on graphs that estimate the displacement vectors linking one plant to another on the same plantation line. Three information branches were considered, being the first used for extracting the plants' positions, the second for extracting the plantation lines, and the third for the displacement vectors. To demonstrate this approach effectiveness, experiments were conducted within a corn plantation field at different growth stages, where some plantation gaps were identified due to problems that occurred during the planting process. Moreover, to verify the robustness of our method with the addition of graphs, we compared it against both a baseline and other state-of-the-art deep neural networks, like (Lin et al., 2020) and (Zhang et al., 2019). Our study brings an innovative contribution related to extracting plantation lines at challenging conditions, which may support several precision agriculture-related practices, since identifying plantation-lines in remote sensing images is necessary for automatic farming processes.

The rest of this paper is organized as follows. In section 2, we detail the structure of our neural network and demonstrate how each step in its architecture is used in favor of extracting the plantation lines. In section 3, we present the results of the experiment, highlighting the performance of our network in relation to its baselines, as well as comparing it against state-of-the-art deep learning-based methods. For section 4, we discuss in a broader tone the implications of implementing graph information into our model, as well as indicating future perspectives in our approaches. Lastly, section 5 concludes the research presented here.

Initially, the proposed method estimates the necessary information from the input image using a backbone and a knowledge estimation module, as shown in Figure 2.1. The first information consists of a confidence map that corresponds to the probability of occurrence of plants in the image (Figure

2.1 (b)). Through this confidence map, it is possible to estimate the position of each plant, which is useful in estimating the plantation lines. The second information corresponds to the probability that a pixel belongs to a crop line (Figure 2.1 (c)). Finally, the third information is related to the estimate displacement of vectors linking one plant to another on the same plantation line (Figure 2.1 (d)). These three information steps are relevant and help in detecting the plantation lines and estimating the number of plants in the image.

After these estimates, the problem of detecting plantation lines is modeled using a graph similar to Zhang et al. (2019). Each plant identified in the confidence map is considered a vertex in the graph. The vertices/plants are connected forming a complete graph (Figure 2.1 (e)). Each edge between two vertices is represented by a set of features extracted from the line that connects the two vertices in the image. These features and information from the knowledge estimation module are used in the edge classification module (Figure 2.1 (f)) that classifies the edges as a planting line. The sections below describe these modules in detail. In Figure 2.1, the features are extracted from the image by means of a backbone and used to extract knowledge related to the position of each plant and line, in addition to displacement vectors between the plants. The position of each plant is modeled on a complete graph and each edge is classified based on the extracted knowledge.

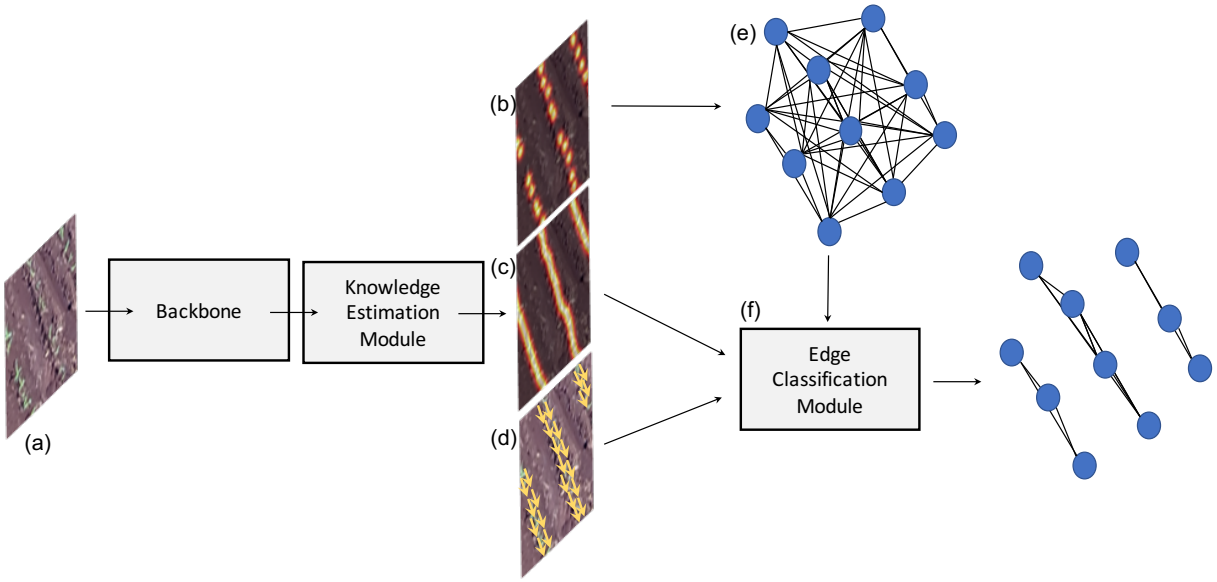


Figure 2.1: The proposed approach composed of different modules.

2.1.1 Backbone - Feature Map Extraction

The first module of the proposed method consists of extracting a feature map F through a backbone as shown in Figure 3.1. In this work, the backbone consists of the initial layers of the VGG16 network (Simonyan and Zisserman,

2015). The first and second convolutional layers have 64 filters of size 3×3 and are followed by a max-pooling layer with window 2×2. Similarly, six convolutional layers (two with 128 filters and four with 256 filters of size 3×3) and a max-pooling layer are then applied. To obtain a resolution large enough, a bilinear upsampling layer is applied to double the resolution of the feature map. Finally, two convolutional layers with 256 and 128 3×3 filters are used to obtain a feature map that describes the image content. All convolutional layers have the ReLU activation function (Rectified Linear Units). Given an input image I with resolution $w \times h$, a feature map F with resolution $\frac{w}{2} \times \frac{h}{2}$ is obtained.

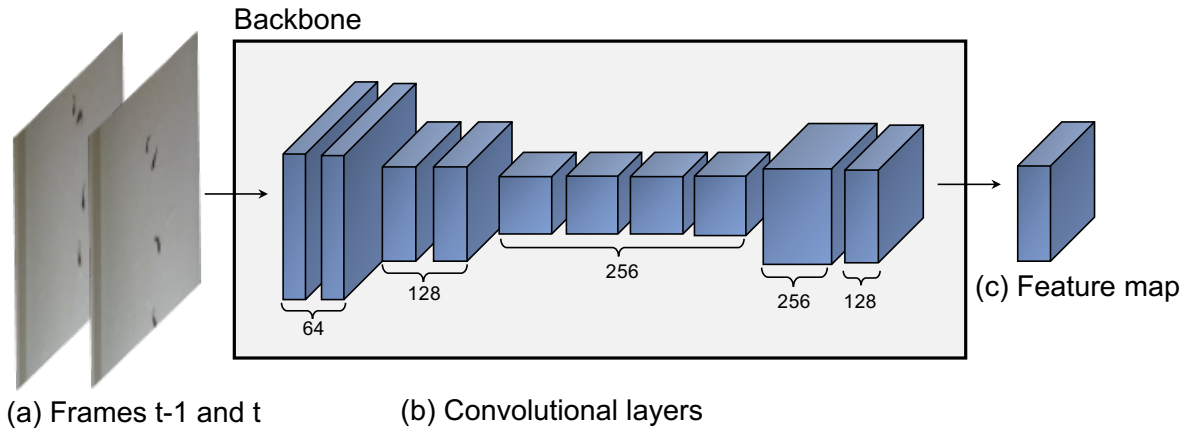


Figure 2.2: The backbone used in the proposed structure is composed of the initial layers of VGG16 and a bilinear upsampling layer.

2.1.2 Knowledge Estimation Module (KEM)

The feature map F is used as an input to the Knowledge Estimation Module - KEM (Figure 2.3). The information is estimated through three branches, each branch consisting of T stages. The first stage of each branch receives the feature map F and estimates a confidence map for the plant positions C_1^p (first branch), a confidence map for plantation lines C_1^r (second branch), and the displacement vectors C_1^v that connect a plant to another on the same plantation line (third branch). The estimation in the first stage is performed by five convolutional layers: three layers with 128 filters of size 3×3 and one layer with 512 filters of size 1×1. The 1×1 filter can perform a channel-wise information fusion and dimensionality reduction to save computational cost. Finally, the last layer has a single filter for estimating plants C_1^p and plantation lines C_1^r, and two filters (i.e., displacement in x, y) for the displacement vectors C_1^v.

At a later stage t, the estimates from the previous stage $C_{t-1}^p, C_{t-1}^r, C_{t-1}^v$ and the feature map F are concatenated and used to refine the estimates C_t^p, C_t^r, C_t^v.

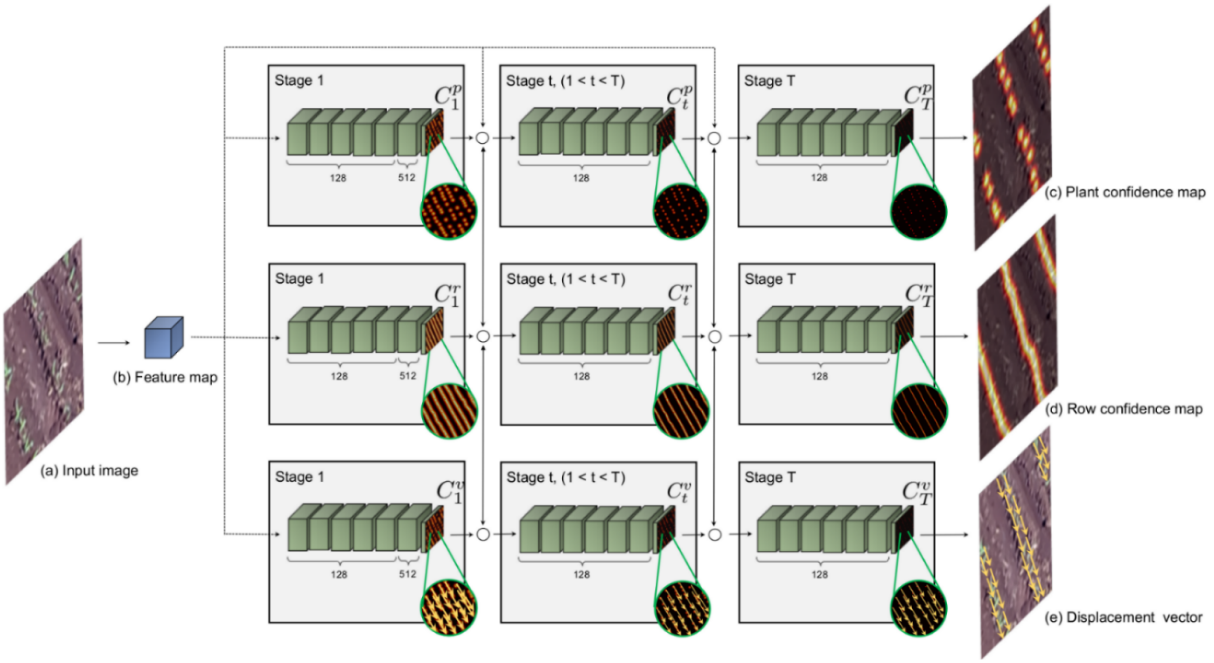


Figure 2.3: Knowledge estimation module related to a confidence map of plants and plantation lines, in addition to displacement vectors between plants on the same line.

The $T-1$ final stages consist of seven convolutional layers, five layers with 128 filters of size 7×7, one layer with 128 filters of size 1×1 and the final layer for estimation according to the first stage.

The multiple stages assist in hierarchical and collaborative learning in estimating the occurrence of plants, lines and displacement vectors (Osco et al., 2020a, 2021a). The first stage performs the rough prediction of the information that is refined in the later stages.

2.1.3 Graph Modeling

The problem of detecting plantation lines is modeled by a graph $G = (V, E)$ composed of a set of vertices $V = \{v_i\}$ and edges $E = \{e_{ij}\}$. Each detected plant is represented by a vertex $v_i = (x_i, y_i)$ with the spatial position of the plant in the image. The vertices are connected to each other forming a complete graph.

The plants are obtained from the confidence map of the last stage, C_T^p. For this, the peaks (local maximum) are estimated from C_T^p by analyzing a 4-pixel neighborhood. Thus, a pixel is a local maximum if $C_T^p(x,y) > C_T^p(x+l, y+m)$ for all neighbors given by $(l \pm 1, m)$ or $(l, m \pm 1)$. To avoid detecting plants with a low probability of occurrence, a plant is detected only if $C_T^p(x,y) > \tau$. In addition, we consider a minimum distance δ so that the detection of very close plants does not occur. After preliminary experiments, we set $\tau = 0.15$ and $\delta = 1$ pixels.

2.1.4 Edge Classification Module (ECM)

Given the complete graph, the detection of plantation lines consists of classifying each edge (Figure 2.4). Here, the feature vectors of the backbone are sampled from the line connecting the vertices i and j and from the estimates made by the knowledge estimation module. This information is used to classify an edge as a plantation line. Each edge e_{ij} is equal to one (existing) only if the vertices v_i and v_j (i.e., plants i and j) belong to the same plantation line. For this, this module estimates three probabilities of a given edge belonging to a plantation line, being related to: i) visual features obtained from the backbone, ii) chance that the edge pixels belong to a line, and iii) alignment of the displacement vectors with the edge, the last two obtained by the knowledge estimation module. Therefore, an edge is classified as a plantation line if the three probabilities are greater than 0.5. The use of different characteristics for the edge classification makes it more robust. The subsections below describe the calculation of the three probabilities.

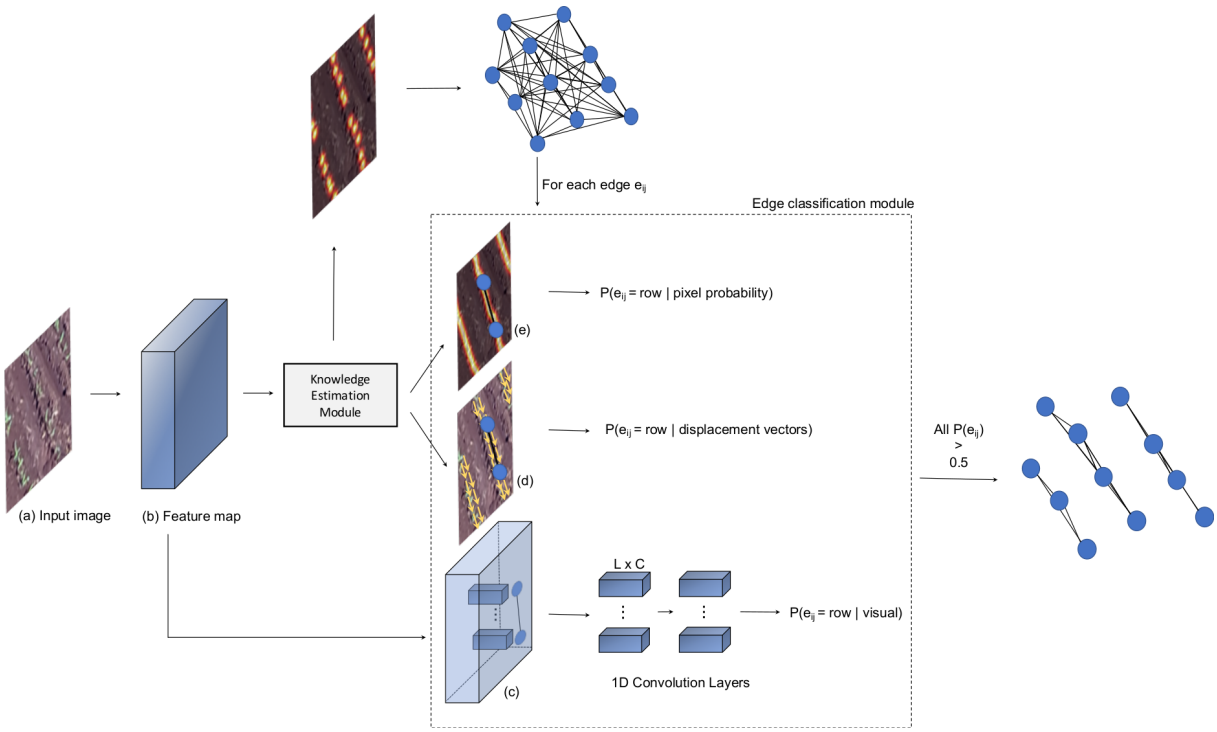


Figure 2.4: Module for extracting features and classifying an edge e_{ij}.

Visual Features Probability

Given an edge e_{ij}, L equidistant points are sampled between $v_i = (x_i, y_i)$ and $v_j = (x_j, y_j)$. For each sampled point, a feature vector is obtained from the backbone activation map. In this way, each edge $e_{i,j}$ is represented by a set of features $F^{e_{i,j}} = \{f_1^{e_{i,j}}, \ldots, f_l^{e_{i,j}}, \ldots, f_L^{e_{i,j}}\} \mid f_l^{e_{i,j}} \in \Re^C$, where C is the number of channels in the activation map ($C = 128$ in this work). To classify an edge using

visual features, $F^{e_{ij}}$ is given as input for three 1D convolutional layers with $128, 256, 512$ filters. At the end, a fully connected layer with sigmoid activation corresponds to the probability of the edge belonging to a plantation line. Figure 2.4 (c) illustrates the process and the features that represent an edge.

Displacement Vector Probability

For each sampled point l on edge e_{ij}, we measure the alignment between the line connecting v_i and v_j and the displacement vector at l. For the two vertices v_i and v_j of e_{ij}, we sample the displacement vectors predicted in C_T^v along the line to calculate an association weight (Cao et al., 2017):

$$\omega_l^{e_{ij}} = C_T^v(l) \cdot \frac{v_j - v_i}{\|v_j - v_i\|_2}, \tag{2.1}$$

where $C_T^v(l)$ corresponds to the displacement vector for the sampled point l between v_i and v_j. Finally, the edge probability based on the displacement vectors is given by the mean, $p(e_{ij} \mid \text{displacement vectors}) = \frac{1}{L}\sum_{l=1}^{L} \omega_l^{e_{ij}}$.

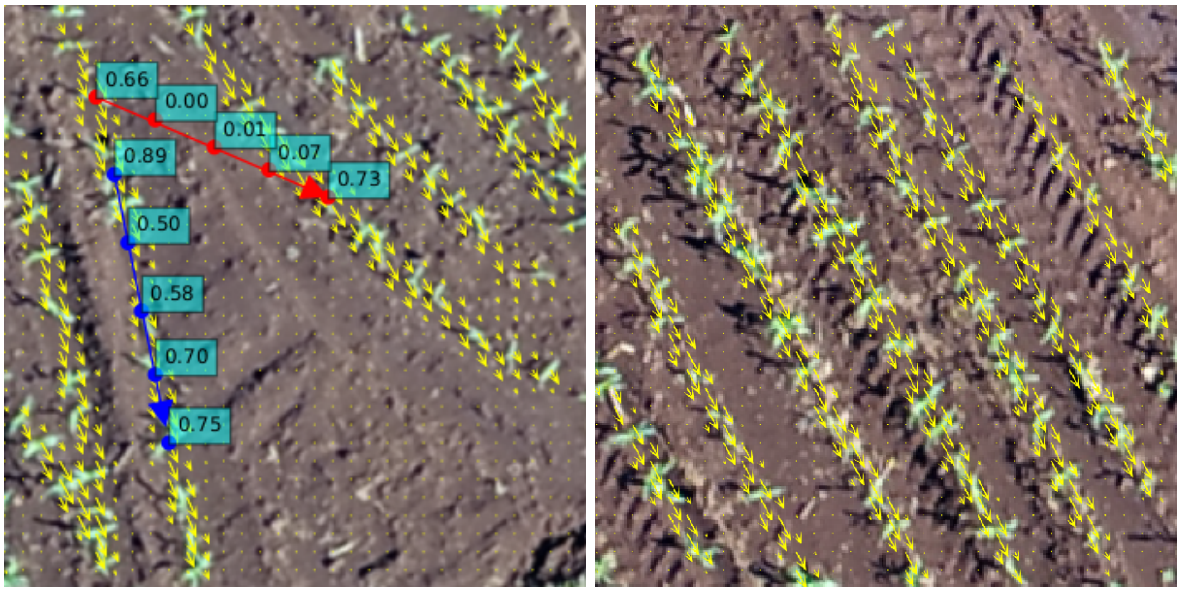
Figure 2.5(a) illustrates the process for estimating the probability of an edge based on the displacement vectors. The blue edge connects two vertices/plants of the same plantation line while the red edge connects two vertices of different lines. For each edge, points are sampled along the line and the weights of the predicted vector alignment and the line connecting the vertices are shown. We can observe that points sampled in a plantation line tend to have a greater weight than points sampled in the background regions. As an illustration, Figure 2.6 presents an example of the displacement vectors estimated by KEM for another test image.

2.1.5 Pixel Probability

This probability is calculated to estimate the edge importance based on the probability that the pixels are from a plantation line. Similarly to the previous section, we sample the points along the line v_i and v_j on the confidence map C_T^r obtained from the KEM. The probability is given by the average of each sampled pixel l:

$$p(e_{ij} \mid \text{pixel probability}) = \frac{1}{L}\sum_{l} C_T^r(l). \tag{2.2}$$

Figure 2.6 presents the calculation for two edges. We can see that the probability of a pixel belonging to a plantation line presents a good initial estimate, although it is not enough to obtain completely connected lines.



(a) (b)

Figure 2.5: (a) Example of the probability of two edges based on the displacement vectors and (b) example of the vectors estimated by the proposed method in a test image.

2.1.6 Proposed Method Training

Although the entire method can be trained end-to-end, we initially trained the knowledge estimation module (KEM). Next, we keep the KEM weights frozen and train the 1D convolutional layers of the edge classification module. This step-by-step training process was adopted to save computational resources. To train KEM, the loss function is applied at the end of each stage according to Equations 3.4, 3.5 and 3.6 for the estimate made for the confidence map of the plant positions, line and displacement vectors, respectively. The overall loss function is given by Equation 3.7.

$$f_t^p = \sum_i \| \hat{C}_t^p(i) - C_t^p(i) \|_2^2 \tag{2.3}$$

$$f_t^r = \sum_i \| \hat{C}_t^r(i) - C_t^r(i) \|_2^2 \tag{2.4}$$

$$f_t^v = \sum_i \| \hat{C}_t^v(i) - C_t^v(i) \|_2^2 \tag{2.5}$$

$$f = \sum_{t=1}^{T} f_t^p + f_t^r + f_t^v \tag{2.6}$$

where $\hat{C}_t^{\,p}$,$\hat{C}_t^{\,r}$ and $\hat{C}_t^{\,v}$ are the ground truths for plant position, lines and displacement vectors, respectively.

$\hat{C}_t^{\,p}$ is generated for each stage t by placing a Gaussian kernel in each center of the plants (Osco et al., 2021a). The Gaussian kernel of each stage t is different and has a standard deviation σ_t equally spaced between $[\sigma_{max}, \sigma_{min}]$.

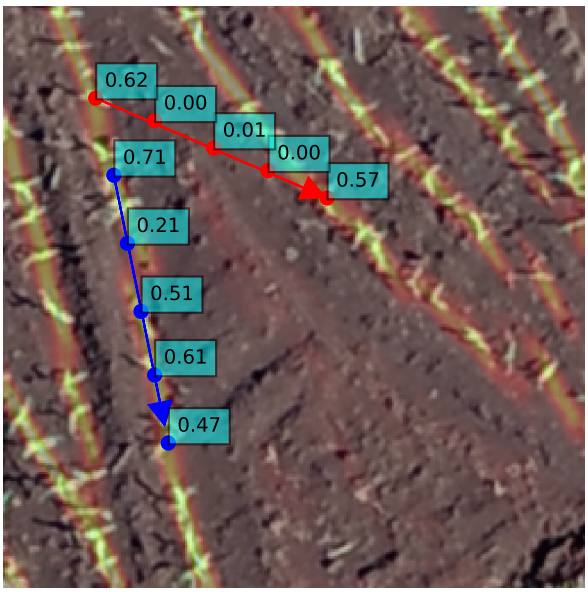


Figure 2.6: Calculation of the probability based on the chance of a pixel belonging to a plantation line.

In preliminary experiments, we defined $\sigma_{max} = 3$ and $\sigma_{min} = 1$. Similarly, $\hat{C}_t^r$ is generated considering all the pixels of the plantation lines and placing a Gaussian kernel with the same parameters as before. On the other hand, $\hat{C}_t^v$ is constructed using unit vectors. Given the position of two plants v_i and v_j, the value $\hat{C}_t^v(l)$ of a pixel l is a unit vector that points from v_i to v_j if l lies on the line between the two plants and both belong to the same plantation line; otherwise, the value $\hat{C}_t^v(l)$ is a null vector. In practice, the set of pixels on the line between two plants is defined as those within the distance limit of the line segment (two pixels in this work).

Figure 3.5 shows examples of ground truths for the three branches of KEM. The RGB image is shown in Figure 3.5(a) while the ground truths for the branches and with three stages are shown in Figures 3.5(b), 3.5(c) and 3.5(d).

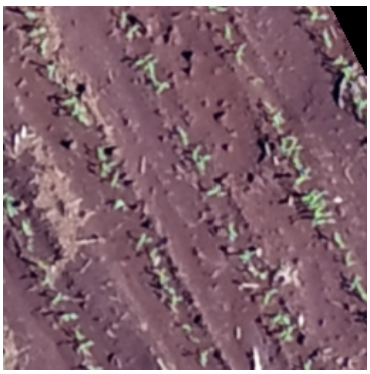
The training of the 1D convolutional layers of the edge classification module is performed using binary cross-entropy loss. Given a set of features that describes an edge e_{ij}, its prediction $y_{e_{ij}}$ is obtained and compared with the ground truth $\hat{y}_{e_{ij}}$ (edge belongs or not to a plantation line) according to:

$$loss = \hat{y}_{e_{ij}} \cdot \log y_{e_{ij}} + (1 - \hat{y}_{e_{ij}}) \cdot \log(1 - y_{e_{ij}}). \tag{2.7}$$

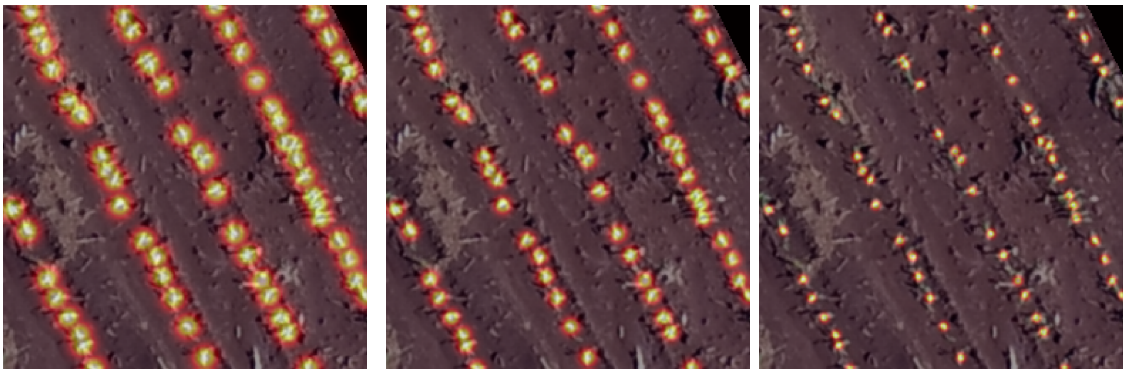
2.2 Experiments and Results

2.2.1 Experimental Setup

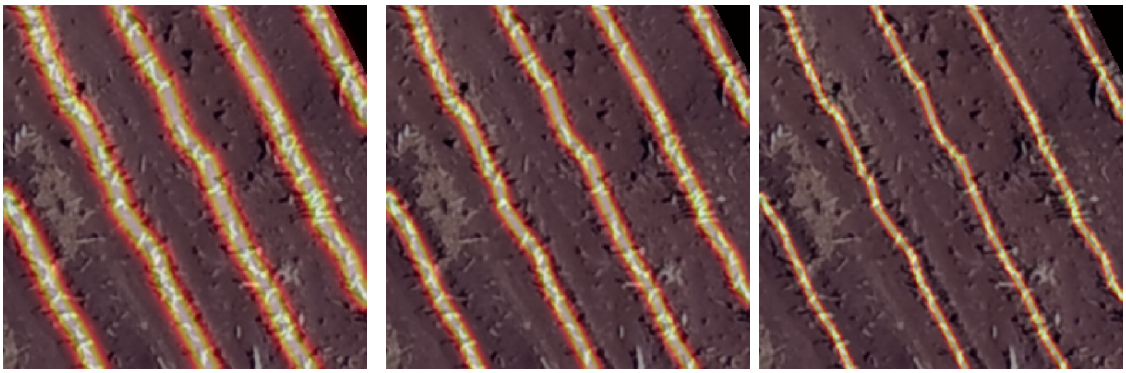
Image dataset: The image dataset used in the experiments was obtained from a previous work (Osco et al., 2021a). The images were captured in an ex-



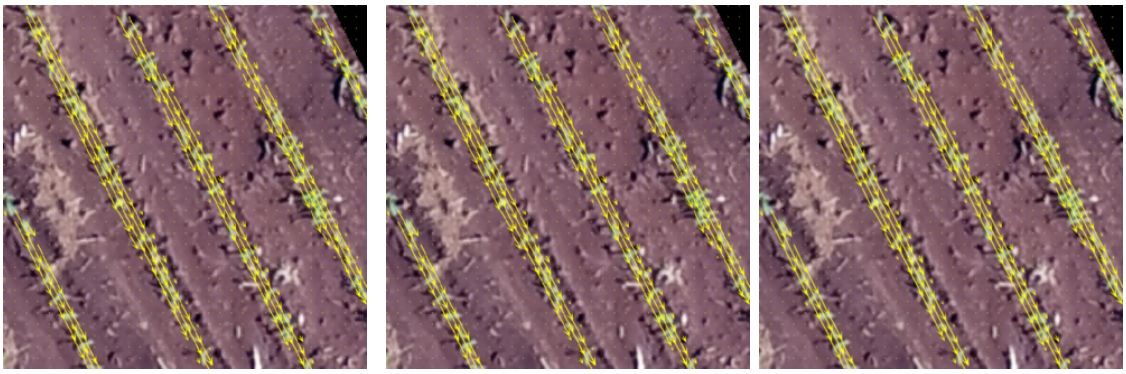
(a)



(b)



(c)



(d)

Figure 2.7: (a) RGB image and ground truths for the three branches ((b) plant positions, (c) lines, (d) displacement vectors) and stages using different values for σ.

perimental area at "Fazenda Escola" at the Federal University of Mato Grosso do Sul, in Campo Grande, MS, Brazil. This area has approximately 7,435 m^2, with corn (*Zea mays L.*) plants planted at a 30×50 cm spacing, which results in 4-to-5 plants per square meter. For two days, the images were captured with a Phantom 4 Advanced (ADV) UAV using an RGB camera equipped with a 1-inch 20-megapixel CMOS sensor and processed with Pix4D commercial software. The UAV flight was approved by the Department of Airspace Control (DECEA) responsible for Brazilian airspace. The images were labeled by an expert initially detecting the plantation-lines. Then, each line was inspected and the plants were manually identified. The entire labeling process was carried out in the QGIS 3.10 open-source software.

The images were split into 564 patches with 256×256 pixels without overlapping. The patches were randomly divided into training, validation, and test sets, containing 60%, 20%, and 20%, respectively. Since the patches have no overlap, it is guaranteed that no part of the images is repeated in different sets.

Training: The backbone weights were initialized with the VGG16 weights pretrained on ImageNet and all other weights were started at random. The methods were trained using stochastic gradient descent with a learning rate of 0.001, momentum of 0.9, and batch size of 4. KEM was trained using 100 epochs while the 1D convolutional layers of ECM was trained using 50 epochs. These parameters were defined after preliminary experiments with the validation set. The method was implemented in Python with the Keras-TensorFlow API. The experiments were performed on a computer with Intel (R) Xeon (E) E3-1270@3.80GHz CPU, 64 GB memory, and an NVIDIA Titan V graphics card, that includes 5120 CUDA (Compute Unified Device Architecture) cores and 12 GB of graphics memory.

Metrics: To assess plant detection, we use the Mean Absolute Error (MAE), Precision, Recall and F1 (F-measure) commonly applied in the literature. These metrics can be calculated according to Equations 2.8, 3.8, 3.9, and 3.10.

$$MAE = \frac{1}{N}\sum_{i} \mid n_i - m_i \mid \tag{2.8}$$

$$Precision = \frac{TP}{TP+FP} \tag{2.9}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.10}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.11}$$

where N is the number of patches, n_i is the number of plants labeled for patch

i and m_i is the number of plants detected by a method. To calculate precision, recall and therefore F1, we need to calculate True Positive (TP), False Positive (FP), and False Negative (FN). For plant detection, TP corresponds to the number of plants correctly detected, while FP corresponds to the number of detections that are not plants and FN corresponds to the number of plants that were not detected by the method. A detected plant is correctly assigned to a labeled plant if the distance between them is less than 8 pixels. This distance was estimated based on the plant canopy (see Figure 2.9 for examples).

Similarly, we use the Precision, Recall and F1 metrics to assess the detection of plantation lines. In contrast, the values of TP, FP and FN correspond to the number of pixels in a plantation line that have been correctly or incorrectly detected by the method compared to the labeled lines. A plantation line pixel is correctly assigned to a labeled one if the distance is less than 5 pixels.

2.2.2 Ablation Study

In this section we individually evaluate the main modules of the proposed method. The first module is the plant detection that has a direct result in the construction of the graph. The next module consists of the edge classification and, in this step, the appropriate number of sampling points L and the influence of each knowledge learned by KEM were evaluated.

Plant Detection

An important step of the proposed method is to detect the plants in the image that will compose the graph for later detection of the plantation lines. Detections occur by estimating the confidence map and detecting its peaks. The results of plant detection varying the number of KEM stages are shown in Table 2.1.

We can see that by increasing the number of stages from 1 to 2, a significant improvement is obtained in the plant detection(e.g., F1 from 0.843 to 0.915). On the other hand, the results stabilize with the number of stages above 2, showing that two stages are sufficient for this step. This is because when using two or more stages, the proposed method is able to refine the detection of the first stage. Figure 2.8 shows the confidence map of the first and second KEM stages for three images of the test set. It is possible to notice that the second stage provides a refinement in the plant detection, which reflects an improvement since two nearby plants can be detected separately.

Examples of plant detection are shown in Figure 2.9. In these figures, a correctly predicted plant (True Positive) is illustrated as a blue dot. The red dots represent false positives, that is, detections that are not plants. Plants that were labeled but were not detected by the method are shown by red circles

Stages	MAE	Precision(%)	Recall(%)	F1(%)
1	10.221	78.9	91.0	84.3
2	**3.531**	**92.7**	**90.5**	**91.5**
4	3.478	91.0	91.4	91.0
6	3.495	91.4	90.9	91.0
8	3.885	89.5	92.0	90.6

Table 2.1: Evaluation of the number of stages in the plant detection.

(the radius of the circle corresponds to the metric threshold). The method is able to detect the vast majority of plants, although it fails to detect some plants very close to each other.

Despite this step, obtaining good results (Precision, Recall, and F1 score of 92.7%, 90.5%, and 91.5%), the detection of all plants in the image is not necessary for the correct detection of the plantation lines. However, the more robust the plant detection is, the greater the chance that the line will be detected correctly.

Edge Classification

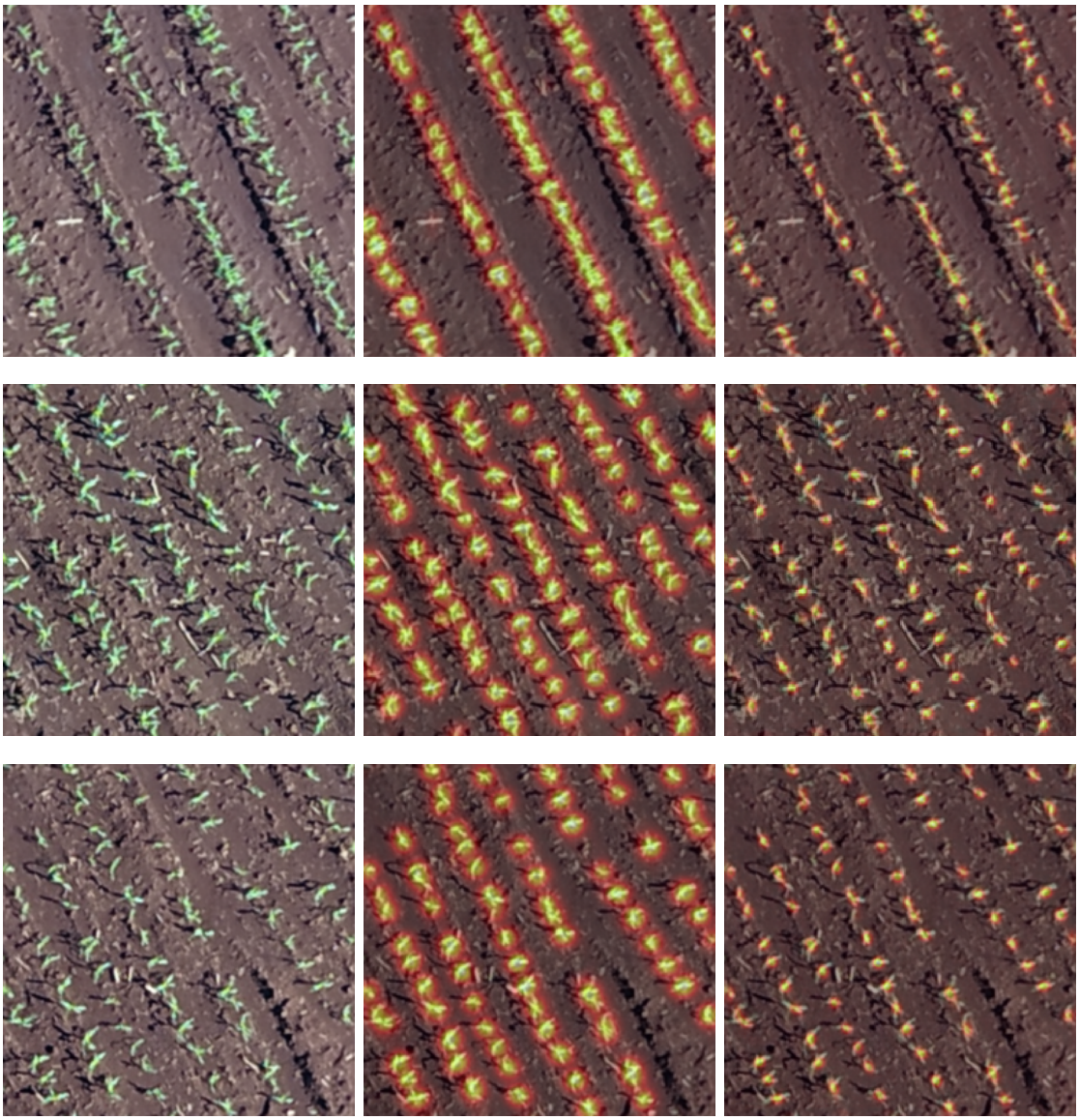
The edge classification module extracts information from the backbone and KEM using L equidistant points along the edge. Then the edges are classified and the plantation lines can be detected. The quantitative assessment of the number of sampled points is shown in Table 2.2. When few points are sampled (e.g., $L = 4$), the features extracted are insufficient to describe the information, especially when two plants are spatially distant in the image. On the other hand, $L \geq 8$ presents satisfactory results for images with a resolution of 256×256 pixels. The best results were obtained with $L = 16$, reaching F1 score of 95.1%.

Number of points	Precision(%)	Recall(%)	F1(%)
4	52.4 ($\pm$37.2)	11.2 ($\pm$9.7)	16.8 ($\pm$13.7)
8	98.5 ($\pm$1.8)	91.0 ($\pm$5.3)	94.5 ($\pm$3.6)
12	98.5 ($\pm$1.8)	91.5 ($\pm$4.8)	94.7 ($\pm$3.3)
16	98.7 ($\pm$1.6)	91.9 ($\pm$4.3)	95.1 ($\pm$2.9)
20	98.6 ($\pm$1.8)	91.9 ($\pm$4.3)	95.0 ($\pm$2.9)

Table 2.2: Evaluation of the number of sampled points L in the detection plantation lines.

Combined Information in the Plantation Line Detection

The edge classification module considers three features to classify an edge as a plantation line: visual, line, and displacement vector features. To assess the influence of each feature, Table 2.3 presents the results considering



(a) RGB Image (b) First stage (c) Second stage

Figure 2.8: Confidence map of the first and second stages for plant detection.

different combinations of features for the edge classification.

When using only the visual features from the backbone, the results are satisfactory with an F1 of 90.7%. When visual features are combined with line or displacement vector features, F1 is increased to 92.3% and 94.9%, respectively. This shows that the features estimated by the KEM are important and assist in the detection of plantation lines. Furthermore, by combining the features as proposed in this work, the best result is obtained.

Examples of plantation line detection are presented in Figure 2.10. Figure 2.10(a) presents the RGB image of three examples, while Figures 2.10(b), 2.10(c), 2.10(d), and 2.10(e) present the detection using visual features, visual + vector displacement features, visual + line features, and all features, respectively. The main challenges occur when two plantation lines are very close. The first example shows that the visual features and the visual + displacement

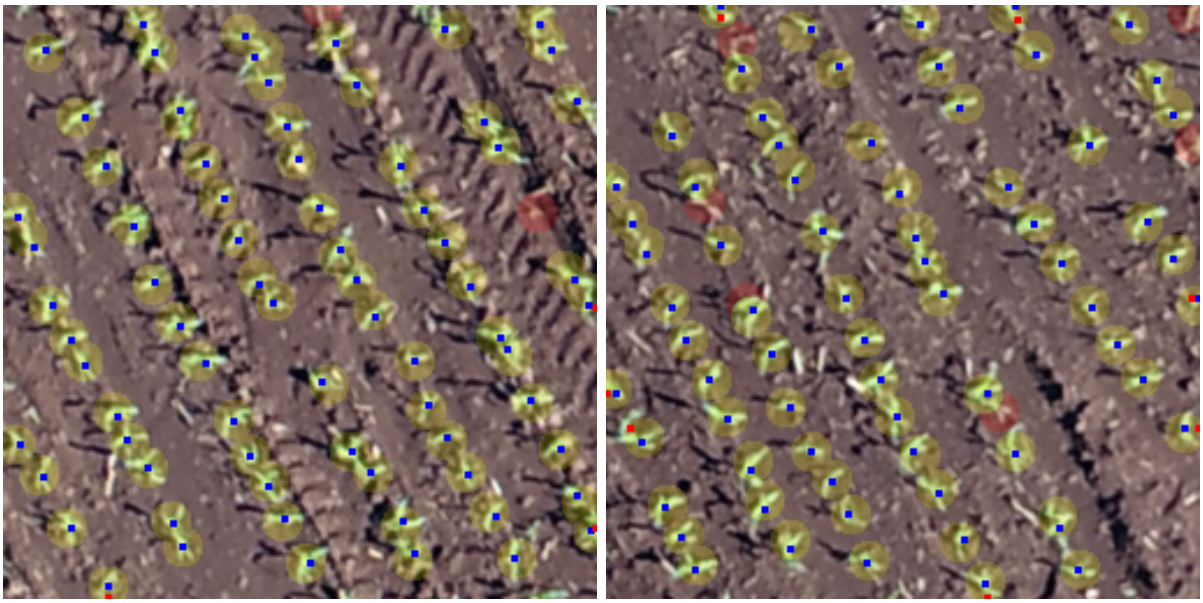


Figure 2.9: Examples of plant detection. Blue dots mean correctly predicted plants, red dots are false positives and red circles are false negatives.

Features	**Precision(%)**	**Recall(%)**	**F1(%)**
Visual Features	94.7 (±6.0)	87.5 (±9.5)	90.7 (±7.8)
Visual + Vector Features	96.3 (±4.4)	89.0 (±7.9)	92.3 (±6.2)
Visual + Line Features	98.4 (±1.9)	91.9 (±4.3)	94.9 (±2.9)
All Features	98.7 (±1.6)	91.9 (±4.3)	95.1 (±2.9)

Table 2.3: Results obtained for different combinations of the features used in the edge classification module.

vector features joined two lines in a single one while the other combinations of features were able to detect them independently. The second and third examples show that the visual features ended up joining two lines at the end, which did not happen with the other combinations. This is because the visual features do not extract structural and shape information, making two plants close in any direction a plausible connection.

2.2.3 Comparison with State-of-the-Art Methods

The proposed method was compared with two recent state-of-the-art methods in Table 2.4. Deep Hough Transform (Lin et al., 2020) integrated the classical Hough transform into deeply learned representations, obtaining promising results in line detection using public datasets. PPGNet (Zhang et al., 2019) is similar to the proposed method since it models the problem as a graph. However, PPGNet uses only visual information to classify an edge, in addition to classifying the entire adjacency matrix, which results in a high computational cost. To address this issue, PPGNet performs block prediction to classify

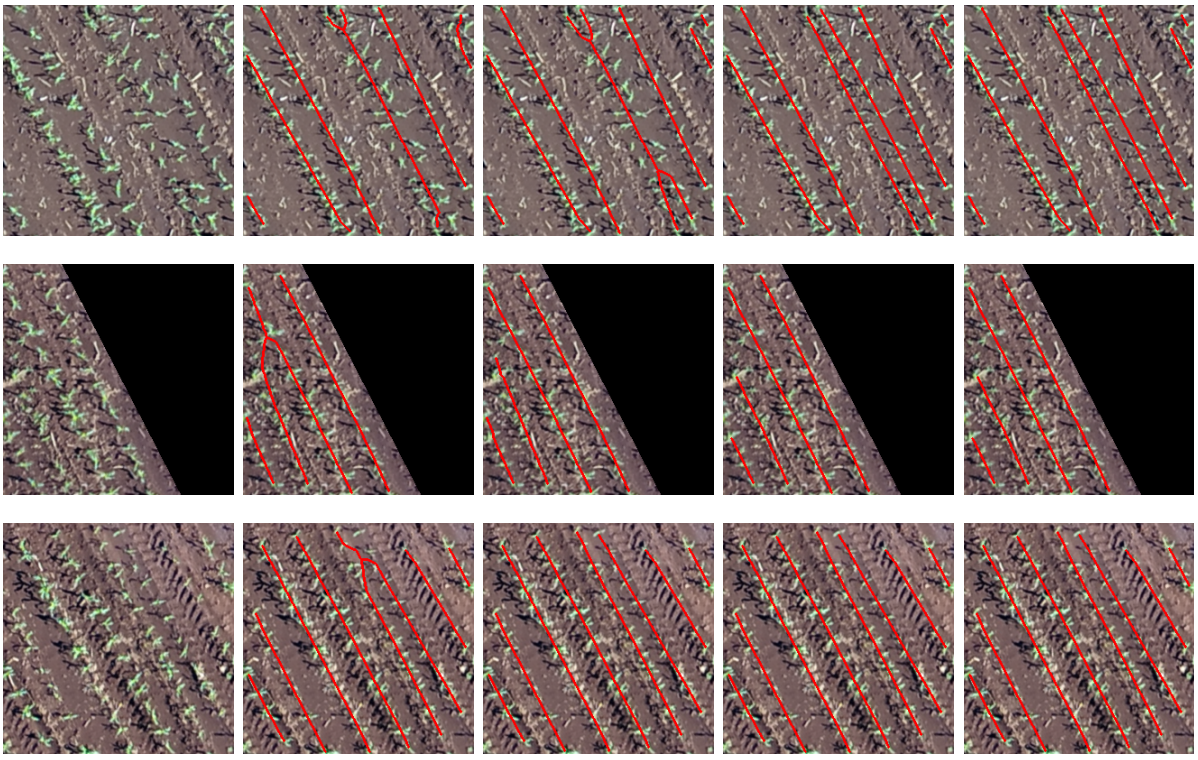


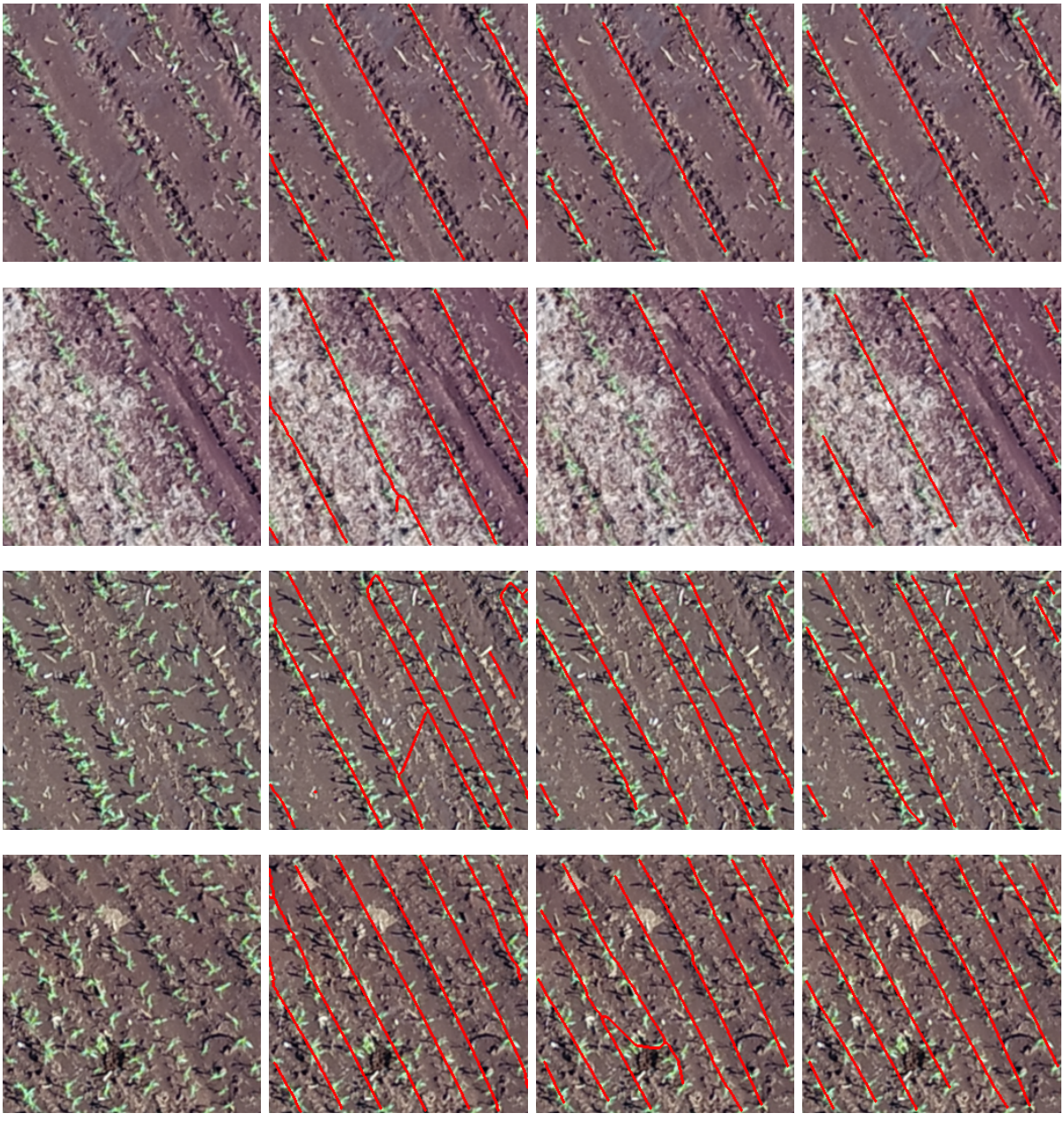
Figure 2.10: Examples of plantation line detection considering different combinations of features in the edge classification module. (a) RGB image, (b) Visual feature, (c) Visual + Vector displacement features, (d) Visual + line features, (e) All features.

the whole adjacency matrix. It is important to emphasize that none of these previous methods has been applied to detect plantation lines.

Experimental results indicate that the proposed method significantly improves F1 score over the traditional approaches, from 91.0% to 95.1%. The same occurs for precision and recall, whose best values were obtained by the proposed method. This shows that the use of additional information (e.g., displacement vectors and line pixel probability) can lead to an improvement in the description of the problem. All methods show good results when the plantation lines are well defined as in the first example of Figure 2.11. On the other hand, Deep Hough Transform has difficulty in detecting lines in regions whose plants are not completely visible (see the second example in Figure 2.11). In addition, some examples have shown that state-of-the-art methods connect different plantation lines (the last two examples in the figure). Hence, the method described here has proven to be effective for plantation line detection.

2.2.4 Generalization in other Cultures

To assess the generalizability of the methods, we report the results in two financially important crops: orange and eucalyptus. The orange dataset is



(a) RGB Image (b) PPGNet (c) Deep Hough Transform (d) Proposed Method

Figure 2.11: Examples of plantation line detection obtained by the compared methods.

Methods	Precision(%)	Recall(%)	F1(%)
Deep Hough Transform (Lin et al., 2020)	94.7 (±6.4)	87.5 (±9.9)	90.1 (±8.7)
PPGNet (Zhang et al., 2019)	95.0 (±3.5)	87.6 (±5.5)	91.0 (±3.5)
Proposed Method	98.7 (±1.6)	91.9 (±4.3)	95.1 (±2.9)

Table 2.4: Comparison of the proposed method with two recent state-of-the-art methods.

composed of 635 images randomly divided into 381, 127 and 127 for training, validation and test. For the eucalyptus dataset, 1813, 604 and 516 images were used for training, validation and test, respectively. The methods were trained with the same hyperparameters as before to show accuracy in crops with different visual characteristics.

Table 2.5 presents the results of the methods in the two crops. In general, the methods achieved adequate results in both crops, with emphasis on the proposed method that achieved the best results. Figure 2.12 illustrates the detection of plantation lines in orange crop. We can see that the orange grove has consistent plantation lines and therefore the methods were successful in detecting. Errors occurred in small disconnections of the lines (last example) and in the detection of trees that are not of the target crop (first example).

Unlike the orange crop, eucalyptus presents a more challenging scenario as illustrated in the examples in Figure 2.13. The presence of other trees is more constant even between the plantation lines (first example), causing PPGNet to make connections between the lines and DHT to leave a disconnected line.

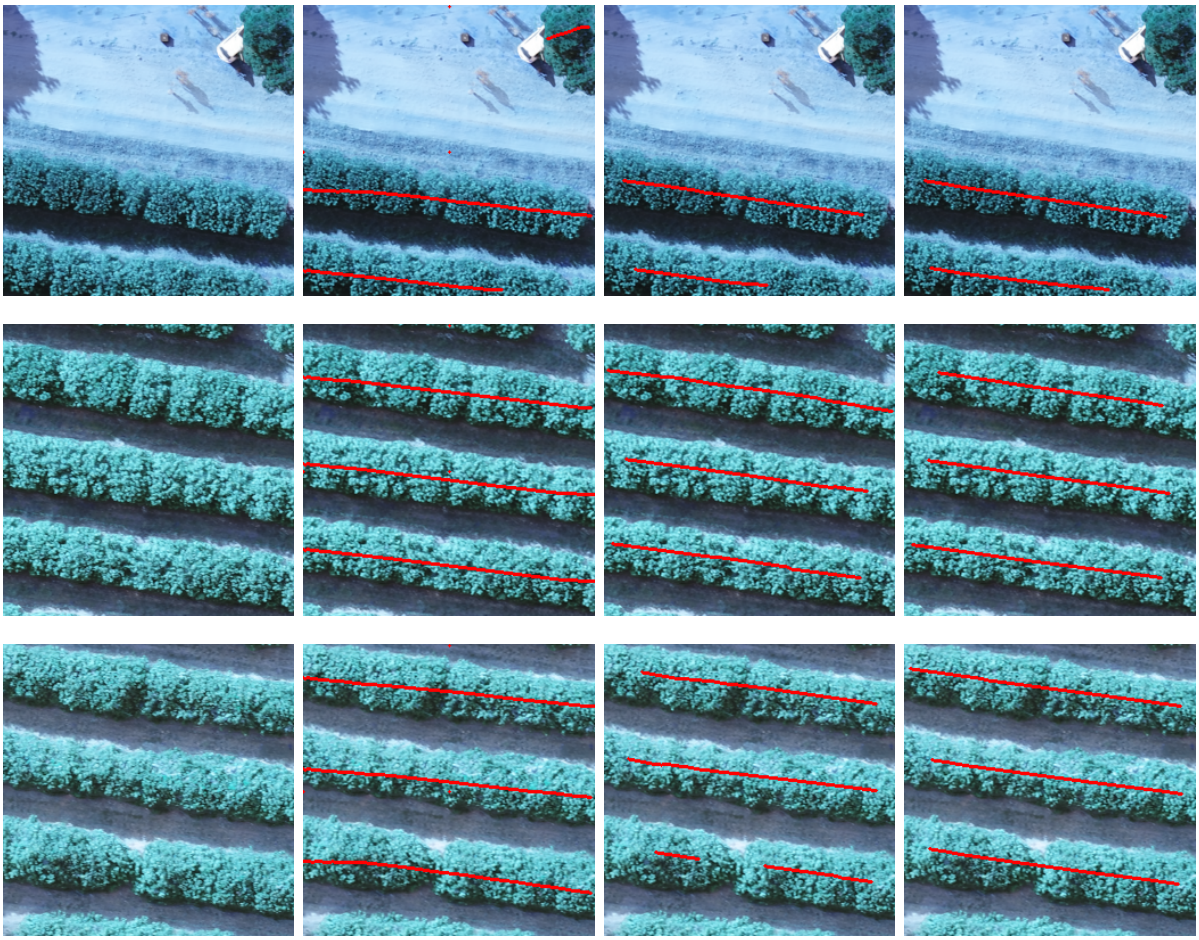
The proposed method on the other hand was more robust to these interferences in most cases. In less challenging scenarios, the methods yield adequate results, such as the second and third example in the figure.

Crop	Methods	Precision(%)	Recall(%)	F1(%)
Orange	Deep Hough Transform (Lin et al., 2020)	96.0 (±6.5)	91.8 (±10.7)	93.2 (±9.1)
	PPGNet (Zhang et al., 2019)	95.0 (±7.1)	91.2 (±9.4)	92.7 (±8.2)
	Proposed Method	98.9 (±2.0)	93.8 (±6.6)	95.9 (±4.3)
Eucalyptus	Deep Hough Transform (Lin et al., 2020)	98.4 (±2.6)	90.6 (±8.9)	93.8 (±6.3)
	PPGNet (Zhang et al., 2019)	84.6 (±10.1)	81.0 (±11.2)	82.3 (±9.9)
	Proposed Method	98.9 (±1.4)	94.4 (±5.4)	96.4 (±3.2)

Table 2.5: Comparison of the proposed method with state-of-the-art methods in two crops (orange and eucalyptus).

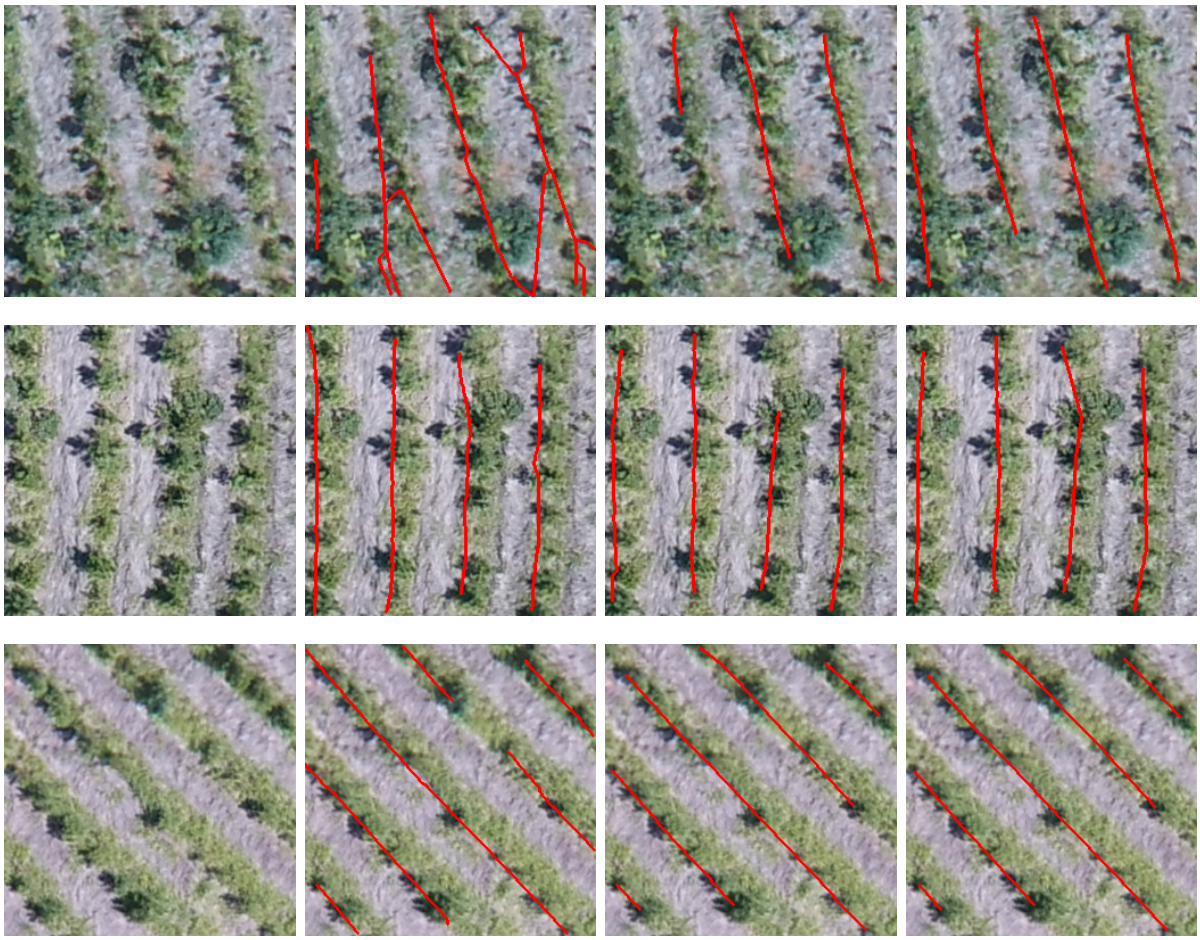
2.3 Discussion

In this study, we investigated the performance of a deep neural network in combination with the graph theory to extract plantation-lines in RGB images to attend agricultural farmlands. For this, we demonstrated the application of our approach in a corn field dataset composed of corn plants at



(a) RGB Image (b) PPGNet (c) Deep Hough Transform (d) Proposed Method

Figure 2.12: Examples of plantation line detection obtained by the compared methods in orange.



(a) RGB Image (b) PPGNet (c) Deep Hough Transform (d) Proposed Method

Figure 2.13: Examples of plantation line detection obtained by the compared methods in eucalyptus.

different growth stages and with different plantation patterns (i.e., directions, curves, space inbetween, etc.). The results from our experiment demonstrated that the proposed approach is feasible to detect both plant and lines positions with high accuracy. Moreover, the comparison of our method against Lin et al. (2020) and Zhang et al. (2019) deep neural networks indicated that our method is capable of returning accurate results, better than those of the state-of-the-art, and, when compared against its baseline (Visual Features), an improvement from 0.907 to 0.951 occurred. As such, we intend to discuss here this improvement and the importance of graphs theory in conjunction with DNN model.

In our approach, we initially identified the plants' position in the image through a confidence map, being this information useful for estimating the plantation lines. Then, the probability that a pixel belongs to a crop line is estimated and, finally, is estimated the displacement of vectors linking one plant to another on the same plantation line. After these estimates, the problem of detecting plantation lines is modeled using a graph, in which each plant identified in the confidence map is assumed as a vertex in the graph, and these vertices are connected forming a complete graph. Each edge between two vertices, then, is used in the edge classification module to classify whether the edges are a planting line. During the aforementioned process, we verified that at the second stage of the KEM the networks' performance works better and that increasing this number of stages would only result in worse results and higher processing time. After this, the plants, which are viewed as the "vertices" by the model, are classified using a given distance between points, where the plantation lines are determined. This information is important since the plantation-line is detected by considering both visual aspects (i.e. spectral and spatial features, texture, pattern, etc.) the line shape itself, and the displacement of the vector features. By considering this displacement of the graph's structure, the network is capable of improving its learning capability concerning the line pattern, especially when differences in the terrain or the direction of the line occurs, since it accounts for the plants' (i.e., vertices) position to one another.

The adoption of graphs theory in deep learning-related approaches is a relatively new concept in remote sensing and has been explored majorly in semantic segmentation tasks (Ouyang and Li, 2021; Hong et al., 2021; Yan et al., 2019; Cai and Wei, 2022). These studies mostly investigated graph convolutional networks and attention-based mechanisms, which differs from the proposal present here. Regardless, there is no denying the graph addition has the potential to assist in learning patterns and positions of most of the surfaces' targets. In remote sensing applied to agricultural problems, the

integration with graphs can help ascertain a series of object detection tasks, especially those that involve certain patterns and geometry information, as any of other anthropic based environments. As such, this approach offers potential not only for plantation line detection but also for other linear forms like river and its margins, roads and side-roads, sidewalks, utility pole-lines, among others, which were already the theme of previous deep learning approaches related to both segmentation and object detection (Weld et al., 2019; Gomes et al., 2020; Weng et al., 2020; Yang et al., 2019)

The detection of plantation lines is not an easy task to be performed by automatic methods, and the usage of graph is necessary to assist it. Some challenges that occurred when considering our baseline, which only considered the visual features and the first two information branches to rely on the plantation lines' position, was the presence of plants outside the plantation lines' range (i.e. highly spaced gaps), as well as isolated plants and weeds, that both offered a hindrance to the plantation-line detection process. Here, when considering the third information branch with the displacement of the vector features, most of these problems were dealt with, resulting in its better performance, both visually and numerically. Regardless, previously conducted approaches that intended to extract plantation lines from aerial RGB imagery were also reportedly successfully, specifically to detect citrus-tress planted in curved rows (Rosa et al., 2020), which form intricate geometric patterns in the image, as well as in an unsupervised manner, in which the plantation line segmentation was a complementary approach to detect weeds outside the line (Dian Bah et al., 2018).It is also important to highlight that most of the works for plantation line detection are based on segmentation that requires dense labeling (i.e., a class has to be assigned to all the pixels). Our approach requires only one point per plant, reducing the labeling effort significantly.

Future perspectives on graph application in combination with deep convolutional neural networks (or any other type of network) for remote sensing approaches should be encouraged. Deep networks are a powerful method for extracting and learning patterns in imagery. However, they tend to ignore the basic principles of the object pattern in the real world. Graphs, on the other hand, can represent these features and their relationship accordingly. As such, this combination of knowledge provided by both methods is quickly gaining attention in remote sensing and photogrametric field, where most real-life patterns are represented. In this regard, discovering learning patterns related to automatic agricultural practices, such as extracting plantation-line information, is one of the many types of geometric-related mappings that could be potentially benefited from the addition of graphs into the DNN model. In summary, our approach demonstrated that the network improved its perfor-

mance when considering this novel information into its learning process by achieving better accuracies than its previous structure and other state-of-the-art methods, as aforementioned.

2.4 Conclusion

This paper presents a novel deep learning-based method to extract plantation lines in aerial imagery of agricultural fields. Our approach extracts knowledge from the feature map organized in three extraction and refinement branches for plant positions, plantation lines, and for the displacement vectors between the plants. A graph modeling is applied considering each plant as a vertex, and the edges are formed between two plants. As the edge is classified as belonging to a certain plantation line based on visual learning features extracted from the backbone, our approach enhances this since there is also a chance that the plant pixel belongs to a line, which is extracted by the KEM method and is refined with information from the alignment of the displacement vectors with the plant/object. Based on the experiments, our approach can be characterized as an effective strategy for dealing with hard-to-detect lines, especially those with spaced plants. When it was compared against the state-of-the-art deep learning methods, including Deep Hough Transform and PPGNet, our approach demonstrated superior performance with a significant margin considering datasets from different cultures. Therefore, it represents an innovative strategy for extracting lines with spaced plantation patterns, and it could be implemented in scenarios where plantation gaps occur, generating lines with few-to-none interruptions.

CHAPTER

3

Convolutional neural network for fingerlings counting using a multi-task learning approach

Preface

Fingerling counting is an important task for decision-making in the aquaculture context. The counting is usually performed visually by a human, which is time-consuming and prone to errors. Artificial intelligence methods applied to image interpretation can be a great strategy for this task-solving, providing as a result an automatic approach to deal with a handling task. However, applying machine learning or even deep learning algorithms to attend aquaculture issues is an underexplored field that requires novel investigations. For this reason, in this study, we propose a new method to locate and count fingerlings in sequence of images using convolutional neural networks. This method performs the predictions of the following tasks: the probability of a fingerling occurring in each pixel; the movement of the fingerlings from one frame to another, and the estimation of the movement direction vector for each pixel. The feature map given as an input to the multi-task step listed above was extrated from two frames using a backbone. The evaluation of the main parameters of the proposed method showed that the stages responsible for refining the predictions reached best results to locate and count fingerlings when two and three stages were used with F1 of 98.11 e 97.89, respectively. The analyzes indicated that the use of temporal information considerably in-

creases the results, reaching F1 of 97.89. The proposed method was evaluated in frames with different numbers of fingerlings (from 0 to 10) and all obtained relevant results, with F1 of 95.42 or higher. The study also showed that, in most cases, the proposed method is able to detect the joining of two fingerlings visually forming a single one, which is considered the main challenge of the detection and counting of fingerlings.

Relation of the paper with MTL. In this second paper we extend the method proposed in the first to process temporal information. As in the first paper, the proposed method belongs to the hybrid strategy, as the tasks share the same backbone (hard parameter sharing), but there is also an exchange of information between the branches through the concatenation of feature maps in specific layers of the method (soft parameter sharing).

3.1 Introduction

Fingerling counting is the task of estimating the number of animals in a given area for decision making. This kind of data is important to calculate the production potential, the necessary amount of feed and even the sale of a specific amount of animals. Despite the importance, counting is usually performed visually by a human. This visual count is time-consuming and prone to errors due to tiredness and difficulties inherent to the human being.

To overcome these issues, automatic systems using images have been proposed (França Albuquerque et al., 2019; Garcia et al., 2020). These systems collect images and the counting is performed by analyzing the images, making this task faster and less costly. Counting occurs by detecting each fingerling (detection-based methods) or by regressing a number to the entire image or parts of it (regression-based methods). In addition, these methods can include temporal information from a sequence of images.

Detection-based methods consist of detecting and obtaining the position of each fingerling in the image. Garcia et al. (2020) e França Albuquerque et al. (2019) presented a system for counting fingerlings using background subtraction, blob detection and Kalman filter. Although relevant results have been achieved, the system is susceptible to a large amount of parameterization (e.g., average fingerling size, average distance, etc.), which makes it difficult to use on a large scale. Some studies already used neural networks in aquaculture cases (Sveen et al., 2021; Zhao et al., 2018b; Zhou et al., 2019). Recently, counting methods have been proposed using convolutional neural networks (CNNs), such as R-CNN used in (Salman et al., 2019). These methods consider a bounding box for each object and can provide both the position (center of the bounding box) and the count (number of bounding boxes).

On the other hand, regression-based methods directly estimate the number of fingerlings establishing a correlation between the features extracted from the image and the target number. Zhang et al. (2020a) proposed a method that divides the image into sub-images containing one or more fish using segmentation. For each subimage, regression is applied to estimate the number of local fish and contributes to the total image count. Fan and Liu (2013) proposed a method that estimates the number of fingerlings based on geometric features (e.g., area, perimeter). The features are inputs for least squares support vector machine (LS-SVM) that performs the regression. CNNs have also been used for regression and counting fish. Zhang et al. (2020b) proposed a hybrid neural network model to estimate a density map and the total number of fish in the image.

Although recent methods have obtained promising results, the high density with overlapping fingerlings is a challenge for counting. In general, object detection methods are not suitable for dense object scenarios (Goldman et al., 2019). In this case, the overlapping of the bounding boxes due to occlusion makes detection and counting difficult. To assist counting in occlusion scenarios, counting in a sequence of images can be important. Analysis of the movement that objects perform in frames can provide valuable information that is not always taken into account when counting objects in an image. In this context, studies show that the movement of objects can assist in the detection and counting, as well as distinguishing them from the background (Ma et al., 2015; Nam and Han, 2016; Danelljan et al., 2015; Wang et al., 2019; Hou et al., 2019; Gonçalves et al., 2020b).

Unlike object tracking, the purpose of this work is to consider information from the past to count and locate fingerlings in a current frame. This work proposes a new method to locate and count fingerlings in sequence of images using convolutional neural networks. Given the current and the previous frames, the proposed method performs the prediction of three tasks. The first task is to estimate a confidence map with the probability of a fingerling occurring in each pixel. The second and third tasks are related to the movement of the fingerlings from one frame to the other, estimating a movement direction vector for each pixel and a confidence map of the movement. Simultaneous learning of tasks results in greater efficiency and accuracy of prediction for each task. After the prediction of the three tasks, they are used to detect and count the fingerlings in a current frame by means of graphs. Experimental results show that the proposed method is able to improve the fingerling count and location using temporal information.

3.2 Materials and Methods

3.2.1 Proposed approach

This section describes the proposed method for detecting and counting fingerlings in sequence of images. Initially, two frames are concatenated and a feature map is extracted using a backbone. This feature map is given as an input to the multi-task step that estimates i) the probability of a fingerling occurring in each pixel of the image, ii) the probability of the pixels belonging to the movement of a fingerling from a previous frame to the current one, and iii) a movement direction vector for each pixel. This temporal information is used together to estimate the position of fingerlings in a current frame.

Feature map extraction

Given two consecutive frames I_t and I_{t-1} with $w \times h$ colored pixels each, they are concatenated to form an input $I = [I_{t-1}, I_t]$ with dimension $w \times h \times 6$. In this way, the previous frame I_{t-1} can add relevant information for the detection and counting of fingerlings in frame I_t.

Input I is passed through a backbone to extract a feature map. The backbone is composed of convolutional layers similar to the VGG16 architecture (Simonyan and Zisserman, 2015) (see Figure 3.1). The first two convolution layers have 64 filters of size 3×3, followed by a maxpooling layer with window 2×2 and stride 2. Then, two layers with 128 filters, one of maxpooling and another four convolution layers with 256 filters are used. As the estimate of the fingerling positions is a dense map, an upsampling layer is applied followed by two convolution layers with 256 and 128 3×3 filters, respectively. The last convolutional layer provides the feature map with resolution $\frac{w}{2} \times \frac{h}{2}$. Reducing the feature map by half is important to extract local features while decreasing the computational cost of the backbone.

Multi-task estimation

For the fingerling count, three tasks are estimated to include temporal information in the detection. The first task is to estimate a confidence map for the position of the fingerlings in frame t (see Figure 3.2(c)). Thus, each pixel represents the probability that it contains a fingerling. The second and third tasks estimate temporal information to assist in detection and counting. The second task (Figure 3.2(d)) is to estimate a confidence map that represents the probability that a pixel belongs to the movement performed by a fingerling from frame $t-1$ to frame t. This information is equivalent to the estimation of the footprint left by the fingerling from one frame to another. The third task

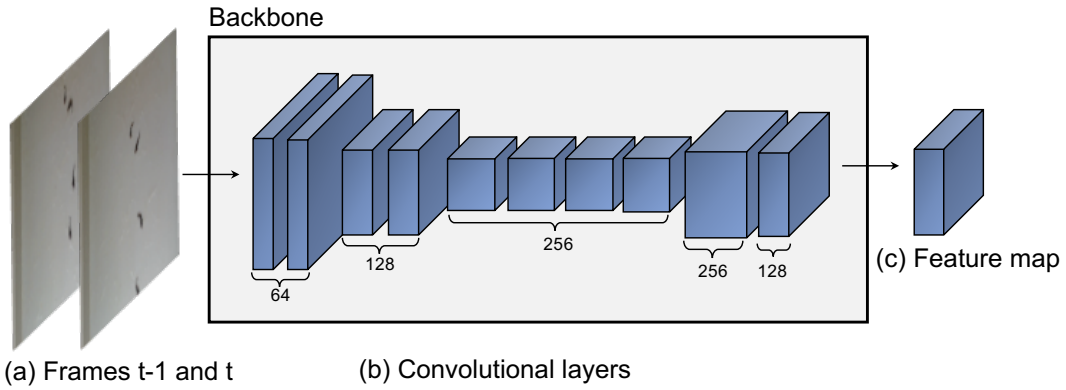


Figure 3.1: Extraction of the feature map from two frames using a backbone based on the VGG architecture.

is similar to the second, but estimates for each pixel a vector that points in the direction of the movement performed by the fingerling (Figure 3.2(e)). This third task is related to the dense optical flow, but here estimated by a branch of the proposed method.

Given the feature map extracted from two frames, each task is estimated on a branch consisting of S stages (Figure 3.2). The first stage of each of the three branches receives the feature map F and performs a series of convolution layers. The first three convolution layers have 128 filters of size 3×3 followed by a layer with 512 filters of size 1×1, all with the ReLU activation function. The last layer of the first and second branches has only one filter to estimate a confidence map for the position of the fingerlings in frame t (C_1^p) and a confidence map corresponding to the temporal movement of the fingerlings (C_1^d). The last layer of the third branch has two filters to estimate a motion vector on the x and y axis (C_1^v).

At the end of the first stage of each of the three branches, estimates C_1^p, C_1^d, and C_1^v could be used to detect fingerlings. However, experimental results have shown that they can be refined by more convolutional layers. For refinement, the later stage s concatenates the estimates from the previous stage $C_{s-1}^p, C_{s-1}^d, C_{s-1}^v$ and the feature map F to estimate the refined information C_s^p, C_s^d and C_s^v. The final $S-1$ stages are composed of seven convolutional layers, five layers with 128 7×7 filters, one layer with 128 1×1 filters and a final layer with the number of filters according to the first stage.

As the information from a previous stage is concatenated, the stages assist in collaborative learning between tasks. The task of detecting the position of the fingerlings in the current frame can be impacted by information related to their movement and direction. In general, the first stage provides a rough prediction that is further refined by the other stages with the exchange of information between tasks.

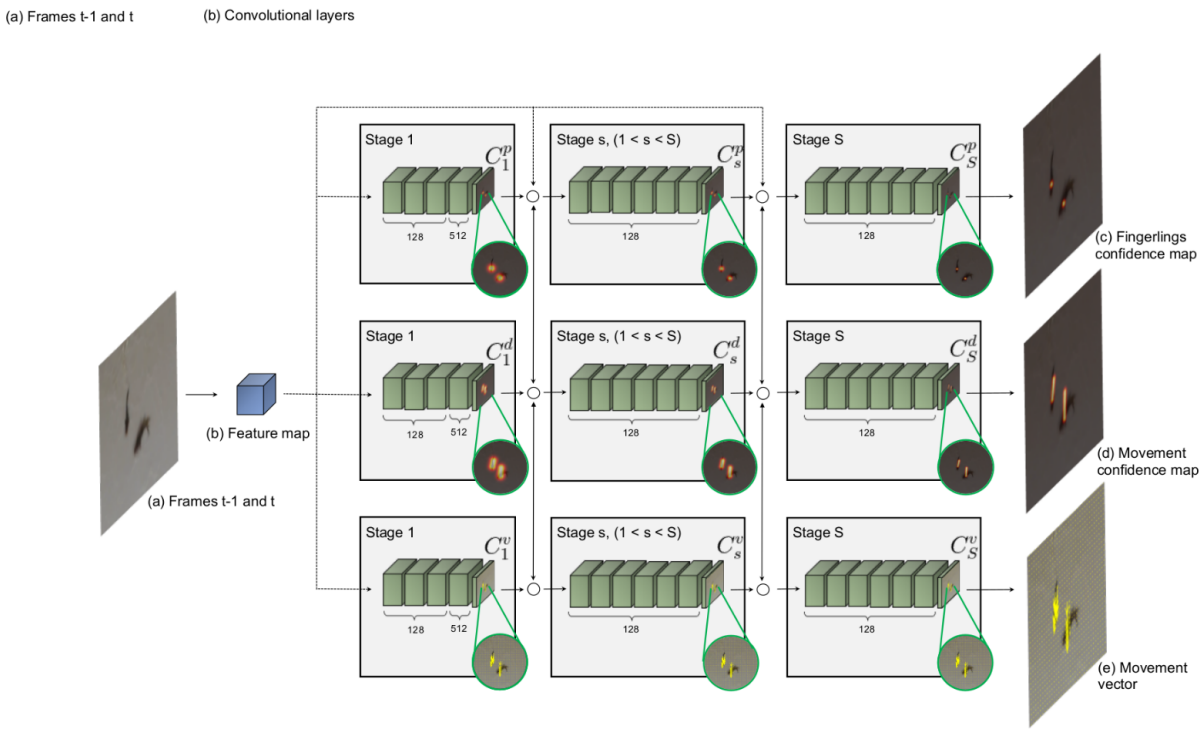


Figure 3.2: Multi-task learning that estimates the position of fingerlings, vector and confidence map of the movement.

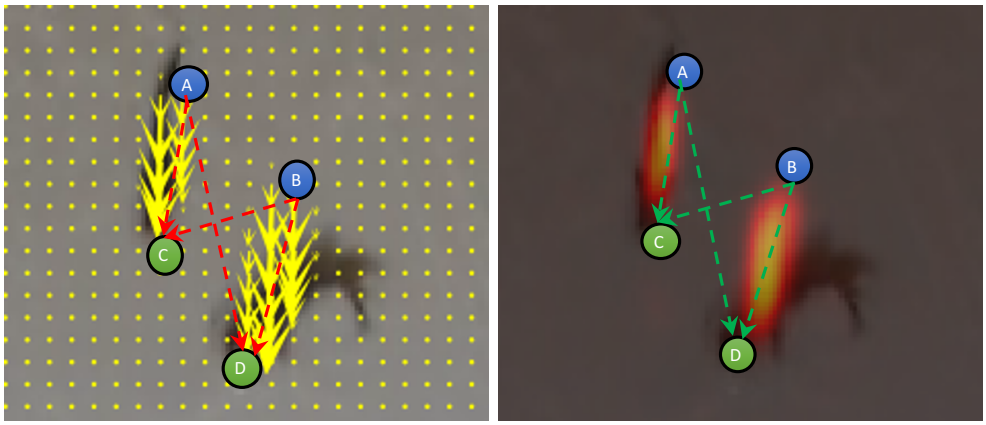
Modeling Fingerlings Movement

The position of the fingerlings in frame t is obtained by the peaks in the confidence map of the last stage C_S^p. A position (x,y) is a peak if $C_S^p(x,y) > C_S^p(x+l,y+m)$ for a 4-pixel neighborhood given by $(l \pm 1, m)$ or $(l, m \pm 1)$. To prevent low probability peaks from being detected as fingerlings, a position (x,y) is considered if $C_S^p(x,y) > \tau$.

The positions detected as fingerlings in the current frame $P_t = \{(x_i,y_i)\} \mid i \in [1,n_t]$ and in a previous frame $P_{t-1} = \{(x_j,y_j)\} \mid j \in [1,n_{t-1}]$ are modeled with a complete bipartite graph. The vertices correspond to the detected fingerlings, being a set of vertices composed of fingerlings from the current frame t and the other set from the previous frame $t-1$. The vertices i of one set are connected by means of edges e_{ij} with all the vertices j of the other set.

To include temporal information, the weight of an edge is calculated using estimates C_S^d and C_S^v. Given an edge e_{ij}, equidistant points are sampled from the line segment between (x_i,y_i) and (x_j,y_j). For each sampled point (x_l,y_l), we calculate the alignment between the line segment $\overline{(x_i,y_i)(x_j,y_j)}$ and the motion vector estimated in $C_S^v(x_l,y_l)$ according to Equation 3.1 (Cao et al., 2017). Finally, the weight of the edge considering the alignment e_{ij}^v is given by the sum of the alignment of all the sampled points, $e_{ij}^v = \sum_l \omega_l^{e_{ij}}$.

$$\omega_l^{e_{ij}} = C_S^v(x_l,y_l) \cdot \frac{(x_j,y_j)-(x_i,y_i)}{\|(x_j,y_j)-(x_i,y_i)\|_2}, \tag{3.1}$$



(a) e^v_{ij} (b) e^d_{ij}

Figure 3.3: Example of calculating the weight of edges e^v_{ij} and e^d_{ij} using predictions C^v_S and C^d_S.

In addition to the previous weight, we calculate a weight e^d_{ij} based on the probability that a pixel belongs to the movement performed by a fingerling. Similarly, we sample points from the line segment $\overline{(x_i,y_i)(x_j,y_j)}$ in C^d_S. The weight is given by the sum of each sampled point l:

$$e^d_{ij} = \sum_l C^d_S(x_l, y_l). \tag{3.2}$$

Finally, the weight of an edge e_{ij} is given by the sum of the two weights to include information from the two tasks:

$$e_{ij} = e^v_{ij} + e^d_{ij} \tag{3.3}$$

Figures 3.3(a) and 3.3(b) illustrate the process for calculating the two weights of an edge based on the motion vector and motion confidence map, respectively. The vertices in blue and green correspond to the fingerlings detected in the previous and current frames. We can see that the edge that connects the vertices A and C is aligned with the motion vectors (Figure 3.3(a)) and the motion confidence map (Figure 3.3(b)), both predicted by the proposed method. Therefore, this edge has a greater weight than the edge that connects the vertices A and D or B and C, for example. A high weight is also associated with the edge that connects vertices B and D.

Fingerlings Detection

Initially we built a complete bipartite graph, where the vertices are the fingerlings detected in the previous and current frame, and the edges are all possible connections between pairs of fingerlings in different frames. Additionally, the edges are weighted according to Equation 3.3.

To detect the fingerlings in the frame t to compose the bipartite graph, we use the confidence map prediction C_S^v searching for peaks even with a low threshold τ_{low}. Thus, the set of fingerlings detected in the frame t is generally greater than ideal and also greater than the number of fingerlings in the previous frame. The fingerlings in frame t are associated with the fingerlings in frame $t-1$, that is, we need to find a pair of fingerlings. The fingerlings that are not associated with any fingerling in the previous frame, but their peak is greater than τ_{high} are maintained as they are probably fingerlings entering the scene. After preliminary experiments, we used $\tau_{low} = 0.005$ and $\tau_{high} = 0.01$.

The optimal association between fingerlings in the complete bipartite graph is reduced to a maximum weight matching problem. Given a bipartite graph, a maximum matching is a subset of edges whose sum of their weights is maximized and that any two edges do not share a vertex. To find the optimal matching, we use the Hungarian algorithm (Kuhn, 1955).

Figure 3.4 shows an example of the detection of fingerlings by means of the complete bipartite graph. The predictions for a frame t are shown in Figures 3.4(b), 3.4(c) and 3.4(d), corresponding respectively to the confidence map of the fingerlings position, movement vectors and confidence movement map. Figure 3.4(a) shows the fingerlings detected in a previous frame (red dots represented by the letters A to D) and the fingerlings detected in the current frame (blue dots with letters from E to K). We can see that the number of fingerlings detected in the frame t is overestimated due to the low threshold used in the confidence map (Figure 3.4(b)). Then, the complete bipartite graph is constructed (Figure 3.4(e)) and the edges are weighted based on the confidence maps illustrated in Figures 3.4(c) and 3.4(d). The optimal match is obtained using the Hungarian algorithm as shown in Figure 3.4(f). Therefore, the fingerlings associated with a previous fingerling are maintained. In addition to these, the fingerlings not associated but with a high peak are also maintained (see the fingerling represented by the letter F in Figure 3.4(g)).

3.2.2 Experimental Setup

Dataset

The dataset used in this work was collected from (Garcia et al., 2020). The frame sequence was obtained with a Logitech C920 PRO WEBCAM HD camera with full hd resolution (720×1280 pixels) and a capture rate of 30 frames per second. The camera was attached to a structure through which the fingerlings slide with the aid of water. In the experiments, the frames were scaled to 512×512 pixels.

Table 3.1 shows the number of frames and the total number of fingerlings

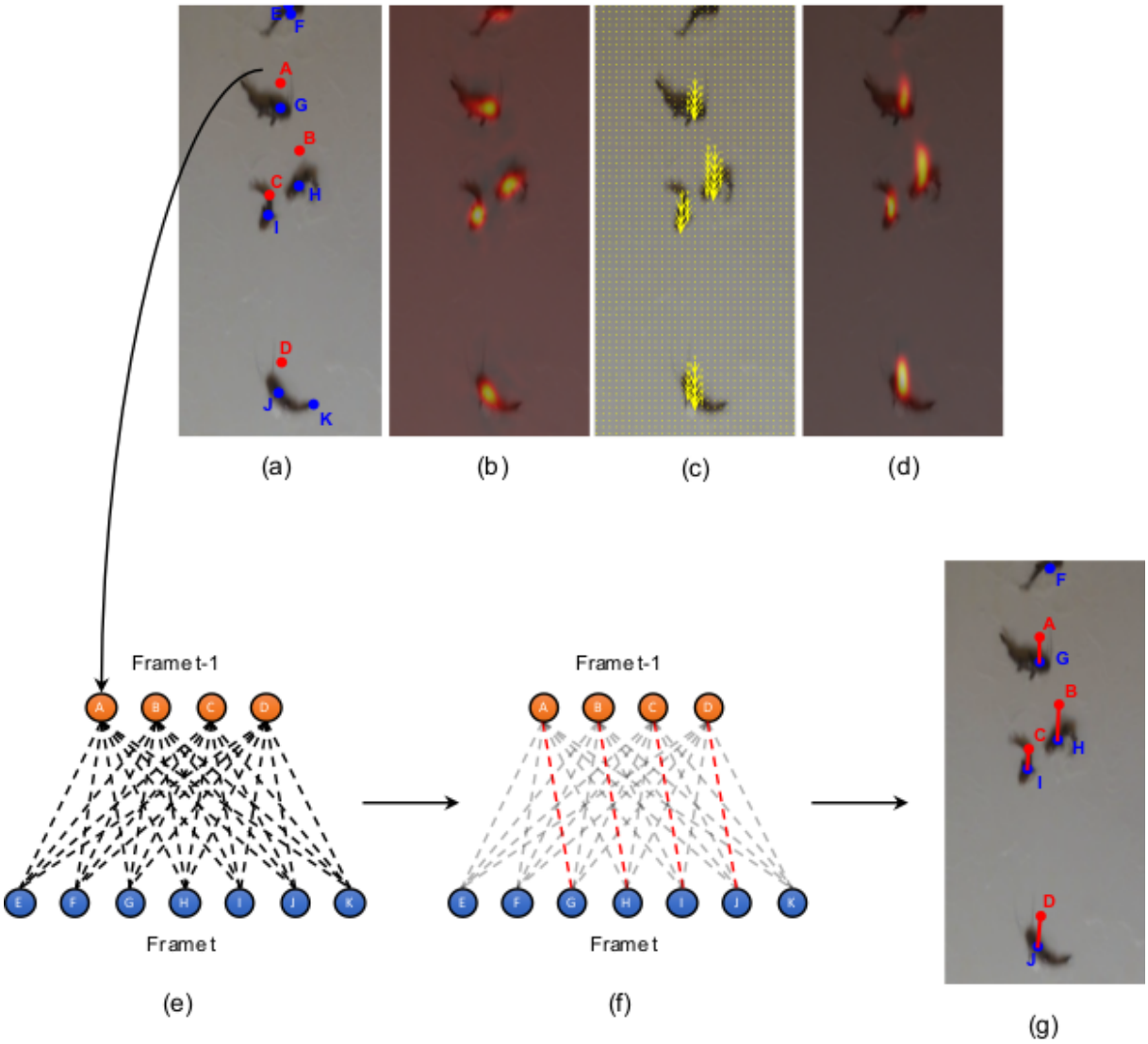


Figure 3.4: Steps for detecting and counting fingerlings in a current frame. For a frame t, Figure 3.4(a) shows the fingerlings detected in the previous frame $t-1$ (red dots) and in the current frame (blue dots). The confidence map of the fingerlings position is shown in Figure 3.4(b). Figure 3.4(c) represents the confidence map of the movement vectors and Figure 3.4(d) represents the confidence movement map. The complete bipartite graph connecting the fingerlings from the previous and current frames is shown in Figure 3.4(e). Figures 3.4(f) and 3.4(g) shows the optimal match obtained using the Hungarian algoritm.

Table 3.1: Description of the dataset in relation to the number of frames and fingerlings.

Set	N. of Frames	N. of Fingerlings
Train	2730	4079
Validation	210	461
Test	1080	2102
Total	4020	6642

Table 3.2: Number of fingerlings per frame.

N. of Fingerlings per frame	N. of frames		
	Train	Val	Test
0-2	2192	158	749
3-5	466	29	262
6-10	72	23	69

for each of the training, validation and test sets. The number of fingerlings per frame is shown in Table 3.2. Most frames have up to two fingerlings, although challenging scenarios with up to 10 fingerlings are present in the dataset.

Each frame was manually annotated with the center of each fingerling. In addition, the position of each fingerling in the previous frame is available to assist in the inclusion of temporal information, as used by the proposed method.

Proposed Method Training

The predictions made by the proposed method using a CNN were trained using stochastic gradient descent. The loss function is applied at the end of each stage s according to Equations 3.4, 3.5 and 3.6 for the predictions of the fingerlings confidence map, movement confidence map and movement vector, respectively. Finally, the overall loss function is given by Equation 3.7.

$$f_s^p = \sum_i \| \hat{C}_s^p(i) - C_s^p(i) \|_2^2 \tag{3.4}$$

$$f_s^d = \sum_i \| \hat{C}_s^d(i) - C_s^d(i) \|_2^2 \tag{3.5}$$

$$f_s^v = \sum_i \| \hat{C}_s^v(i) - C_s^v(i) \|_2^2 \tag{3.6}$$

$$f = \sum_{s=1}^{S} f_s^p + f_s^d + f_s^v \tag{3.7}$$

where $\hat{C}_s^p$,$\hat{C}_s^d$ and $\hat{C}_s^v$ are the ground truths for fingerling positions, movements and vectors, respectively.

Ground truths are generated as follows. $\hat{C}_s^p$ for a stage s is generated by placing a Gaussian in each fingerling position (Osco et al., 2021a). To pro-

mote refinement during the stages, the Gaussian kernel of each stage has a standard deviation equally spaced between $[\sigma_{max}, \sigma_{min}]$. On the other hand, $\hat{C}_s^d$ is generated from the movement of each fingerling. For this, a Gaussian kernel is positioned in each pixel of the line that connects the position of a fingerling in the previous and current frame that were previously labeled. The parameters of the Gaussian kernel of each stage follow as the previous one. Finally, $\hat{C}_s^v$ is constructed similarly to $\hat{C}_s^d$, but using unit vectors. $\hat{C}_s^v$ is a unit vector that points from the position of a fingerling in the previous frame to its position in the current frame.

Figure 3.5 presents the ground truths for three stages using different values of σ. The RGB image is shown in Figure 3.5(a) while the ground truths are shown in Figures 3.5(b), 3.5(c) and 3.5(d). We can see that the first stage (first column of images) has more coarse ground truths while the ground truths of the later stages are more adjusted. This allows the proposed method to learn to refine its predictions in the later stages.

During training, the backbone was initialized with the pre-trained weights on ImageNet. We used the stochastic gradient descent optimizer with a learning rate of 0.01, momentum of 0.9, and batch size of 2 during 100 epochs. These parameters were defined after preliminary experiments with the validation set.

Metrics

To assess the detection of fingerlings, we use the Precision, Recall and F1 (F-measure) commonly applied in the literature. These metrics can be calculated according to Equations 3.8, 3.9, and 3.10.

$$P = \frac{TP}{TP+FP} \tag{3.8}$$

$$R = \frac{TP}{TP+FN} \tag{3.9}$$

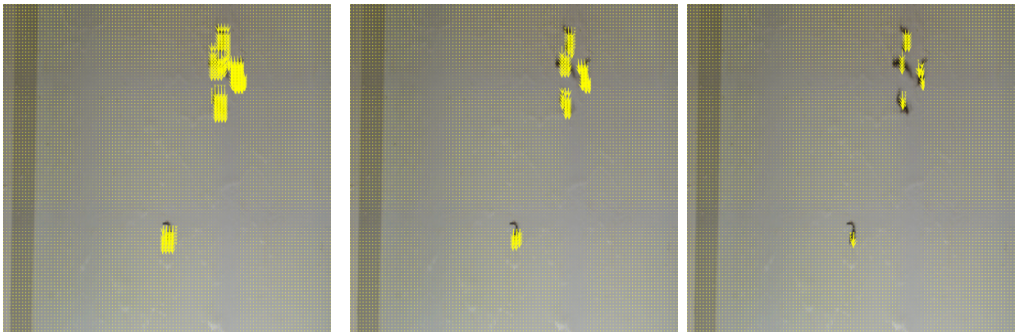
$$F1 = 2 \cdot \frac{P \cdot R}{P+R} \tag{3.10}$$

where TP, FP and FN stand for True Positive, False Positive, and False Negative, respectively. Since the labeling of each fingerling is only one point, a prediction is correctly assigned to a labeled fingerling if the distance between them is less than 20 pixels. This distance was estimated based on the average size of a fingerling in the image.

(a)

(b)

(c)



(d)

Figure 3.5: Example of the ground truths generated for frame. Each column of images presents the ground truth for the stages.

Table 3.3: Influence of σ_{max} on fingerling count using $\sigma_{min} = 1$ and number of stages $S = 2$.

σ_{max}	Precision	Recall	F1
1	90.03	98.20	92.90
2	94.09	98.15	95.37
3	96.96	99.02	97.61
4	**97.51**	**99.37**	**98.12**
5	96.07	99.20	97.17

Table 3.4: Influence of σ_{min} on fingerling count using $\sigma_{max} = 4$ and number of stages $S = 2$.

σ_{min}	Precision	Recall	F1
1	**97.51**	**99.37**	**98.12**
2	95.56	99.35	96.87
3	94.52	98.48	95.90

3.3 Experiments and Results

3.3.1 Parameter Analysis

This section assesses the influence of the main parameters of the proposed method, which includes σ_{max} and σ_{min} used to generate the ground truths and the number of stages S used to refine the predictions. The results for different values of σ_{max} are shown in Table 3.3. In these results, we used two stages $S = 2$, the first varying σ_{max} of the ground truths and the second with σ_{min} set to 1.0. The results show that F1 increases as σ_{max} also increases. In this way, the first stage makes a rough prediction and the second stage refines the predictions. The best result was obtained with $\sigma_{max} = 4$, as it adequately covers the fingerling (see Figure 3.6(a)).

We also assessed the influence of σ_{min} according to Table 3.4. This parameter is responsible for the confidence map of the last stage in which the fingerlings are detected. σ_{max} was set to 4 due to previous results and we maintained two stages. As we can see small values of σ_{min} provide better results in general. This is because the smaller spreading in the last stage allows an accurate refinement of the position of the fingerlings even when they are close. Figure 3.6 shows the prediction of multi-tasks using $\sigma = 1$ and 4. It is possible to observe that the predictions using smaller values are more adjusted.

Finally, we evaluated the number of stages as reported in Table 3.5, which are responsible for refining the predictions. When using only one stage, the results are inferior to the others, which shows that refinement is an important part of the proposed method. Using two and three stages, the proposed method achieves its best results with F1 of 98.11 and 97.89, respectively. With

more stages, the number of layers and consequently the number of weights to be learned increases, which can make training difficult. With these experiments, the best results of the proposed method were obtained using $\sigma_{max} = 4$, $\sigma_{min} = 1$ and number of stages $S = 2$ or 3.

Table 3.5: Influence of the number of stages on fingerling count using $\sigma_{min} = 1$ and $\sigma_{max} = 4$.

Stages (S)	Precision	Recall	F1
1	83.73	97.80	88.40
2	**97.51**	**99.37**	**98.11**
3	**97.45**	**98.99**	**97.89**
4	94.81	98.96	96.26

3.3.2 Temporal Analysis

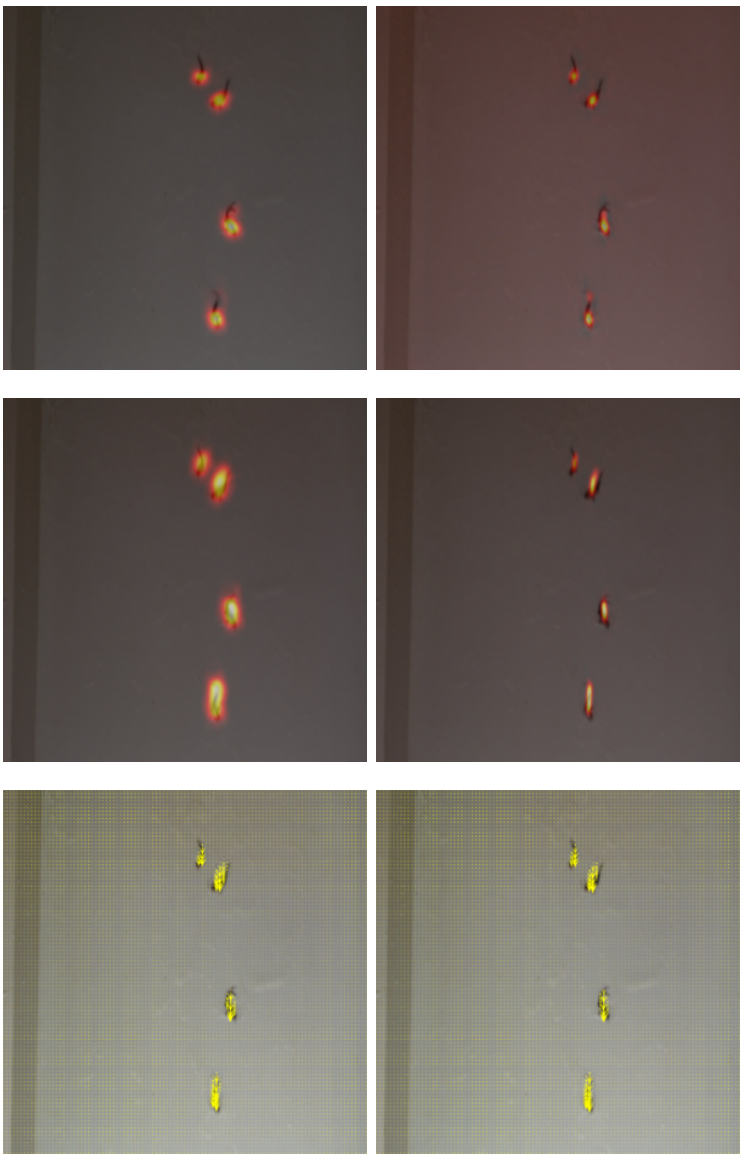
The detection of fingerlings in the current frame occurs after association in the complete bipartite graph. The weight of the edges is calculated based on two predictions related to the movement of the fingerlings. This section assesses the influence of the weight of the edges on the association and, consequently, on the detection of fingerlings according to Table 3.6.

The first line of Table 3.6 presents the results by performing the detection directly on the confidence map of the fingerlings' positions using the best parameters found in the previous section. Thus, although this prediction may contain temporal information due to the sharing of multi-tasks on CNN, they are not used directly in detection. The other lines of the table show the results when the bipartite graph and temporal information are used explicitly. The second and third lines present the results considering separately the two predictions in the calculation of the weight of the edges. Finally, the last line of the table presents the results of the proposed method, in which the two predictions are used (Equation 3.3). The results show that using temporal information individually leads to overestimation of fingerlings (false positive), resulting in lower precision. On the other hand, the proposed method decreases the detection of false fingerlings improving the precision without decreasing the recall.

Table 3.6: Comparative results using temporal information on edge weight.

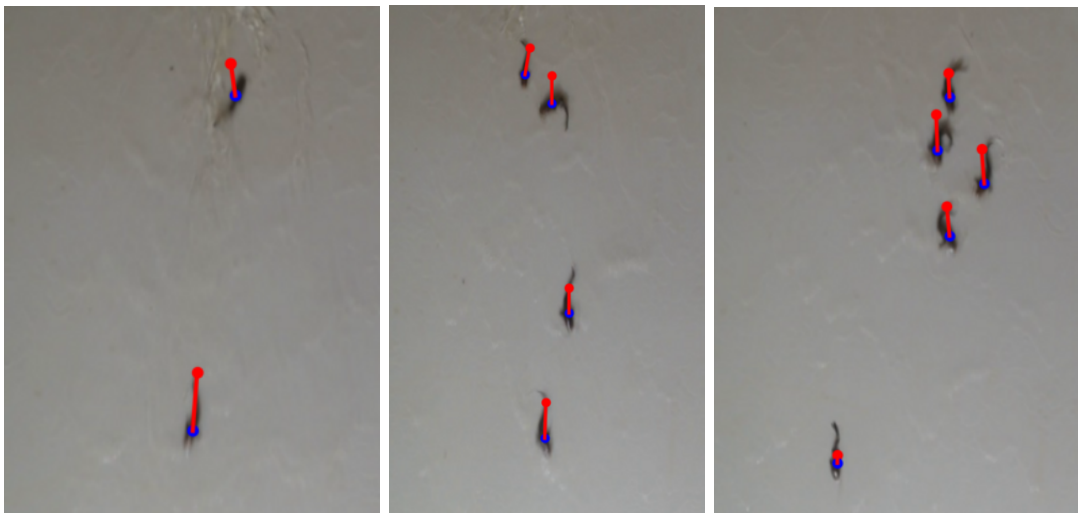
Temporal Information	Precision	Recall	F1
Confidence map	90.27	98.74	93.25
C_S^d	95.97	98.82	96.94
C_S^v	95.87	98.72	96.82
Both	97.45	98.99	97.89

The experimental results show that the detection without the use of tempo-



(a) $\sigma = 4$ (b) $\sigma = 1$

Figure 3.6: Example of predictions made in (a) the first and (b) last stages. Each row of images represents the predictions for the position of the fingerlings, the confidence map of the movement and the movement vectors.



(a) (b) (c)

Figure 3.7: Examples of detection of (a) two, (b) four and (c) five fingerlings.

ral information is not adequate, especially when the fingerlings are very close, forming only one visually. The use of temporal information increases results considerably (e.g., from 93.25 to 96.94 and 96.82). The combination of the two predictions further increases the results, reaching F1 of 97.89.

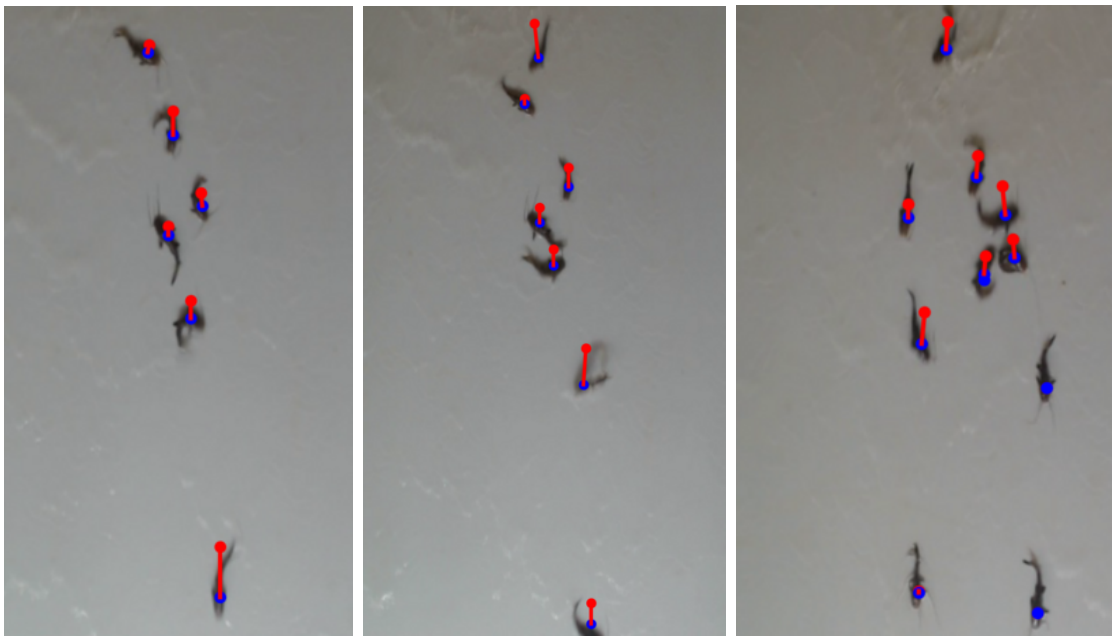
3.3.3 *Density Analysis*

The proposed method was evaluated in frames with different numbers of fingerlings as shown in Table 3.7. The ranges comprise frames containing from 0 to 2 fingerlings, from 3 to 5 and from 6 onwards. As expected, the results decrease as the number of fingerlings per frame increases. With up to two fingerlings per frame, the proposed method reached F1 of 98.61 while from three to five fingerlings, F1 of 97 was obtained. Figure 3.7 shows examples of detection of fingerlings in the 0-2 and 3-5 ranges. Red and blue dots indicate the position of fingerlings detected in the previous and current frames, respectively. The connections show the result of the association of the bipartite graph.

Table 3.7: Results of detection and counting in frames with different amounts of fingerlings.

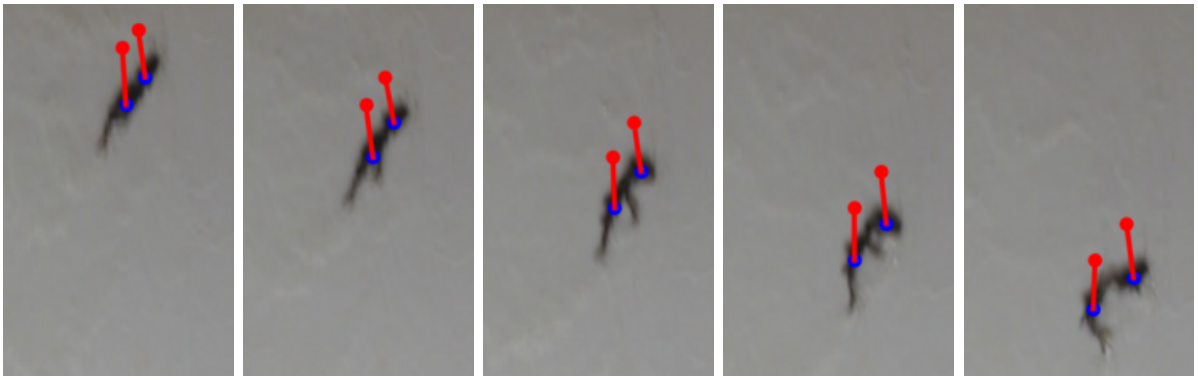
N. of Fingerlings per frame	Precision	Recall	F1
0-2	98.01	99.81	98.61
3-5	95.94	98.87	97.00
6-10	96.81	94.41	95.42

Relevant results were also obtained in frames with a large number of fingerlings (6-10) with an F1 of 95.42. Examples of detection with high density of



(a) (b) (c)

Figure 3.8: Examples of detection of (a) six, (b) seven and (c) ten fingerlings.



(a) #159 (b) #160 (c) #161 (d) #162 (e) #163

Figure 3.9: Example of counting and detecting fingerlings in contact.

fingerlings are shown in Figure 3.8. The proposed method was able to detect six, seven and ten fingerlings even when they are close and moving due to the use of multi-tasks approach.

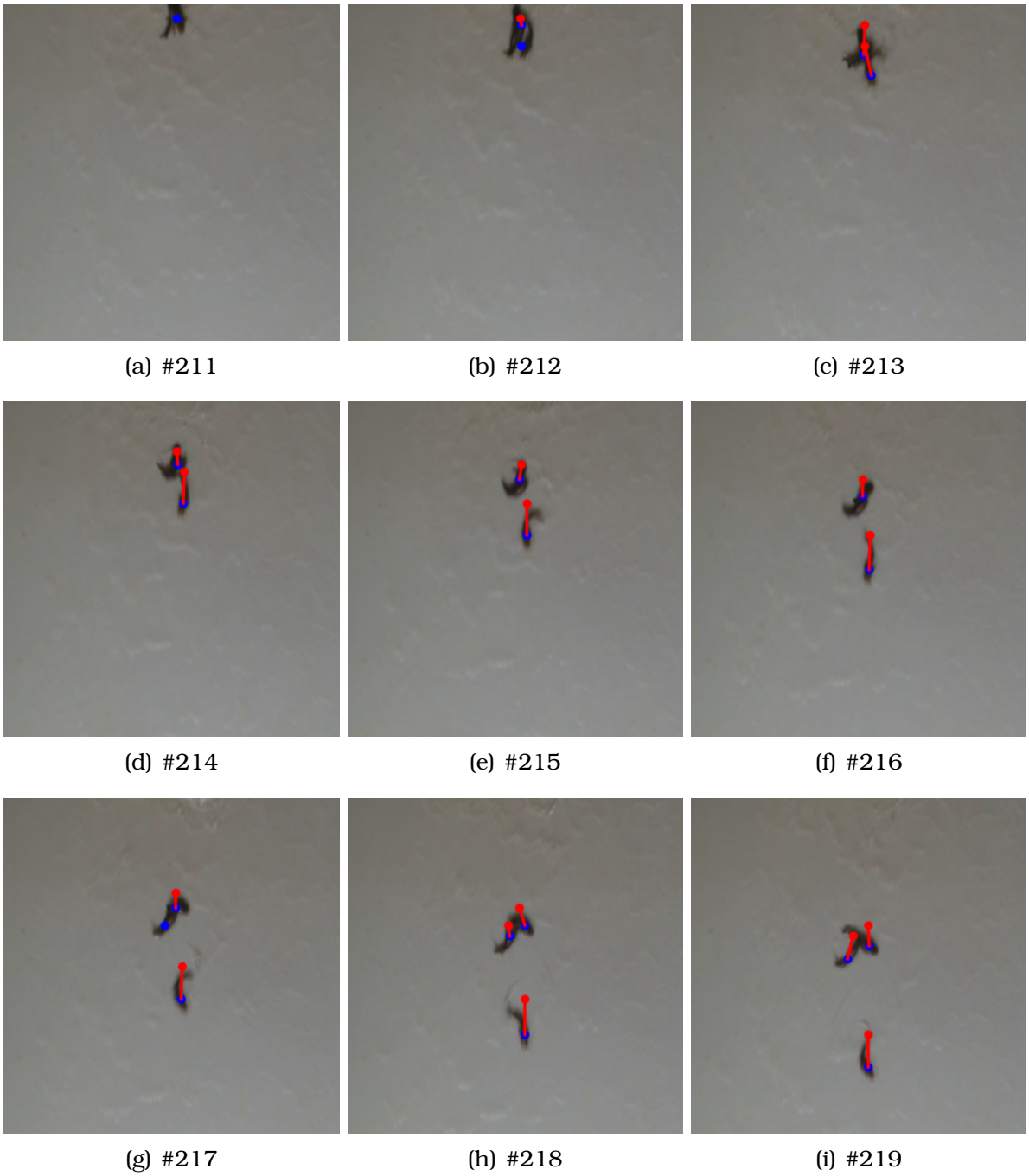
The main challenge of the detection and counting of fingerlings is the joining of two fingerlings visually forming a single one. Despite the challenge, the proposed method is able to detect the two fingerlings in most cases, as shown in Figure 3.9. This is possible due to the association of a fingerling detected with low probability in the current frame with a fingerling in the previous frame. Without this association and the use of multi-task, one of the fingerlings would be discarded due to its low probability.

On the other hand, the errors of the proposed method occur mostly when two or more fingerlings enter the scene connected. The sequence of frames in

Figure 3.10 illustrates this situation. Although it is not possible to visually observe, three fingerlings enter the scene, but only one fingerling is detected initially. In the following frame, the proposed method detects two fingerlings while the third fingerling is only detected in the seventh frame of that sequence.

3.4 Conclusions and future work

The proposed method showed satisfactory results to locate and count fingerlings in sequence of images using convolutional neural networks. Regarding the analyzed parameters, the best results of the proposed method were obtained using $\sigma_{max} = 4$, $\sigma_{min} = 1$ and number of stages S = 2 or 3. Using two and three stages, the proposed method achieves its best results with F1 of 98.11 e 97.89, respectively. The analyzes also indicated that the use of temporal information increases results considerably (e.g., from 93.25 to 96.94 and 96.82), and the combination of the two predictions further increases the results, reaching F1 of 97.89. The proposed method was evaluated in frames with different numbers of fingerlings, the results showed that with up to two fingerlings per frame, the proposed method reached F1 of 98.61, from three to five fingerlings, F1 of 97, but also obtained relevant results in frames with a large number of fingerlings (6-10) with an F1 of 95.42. These results prove that the proposed method was able to detect six, seven and ten fingerlings even when they are close and moving due to the use of multi-tasks. Another advantage of this study is that, in most cases, the proposed method is able to detect the joining of two fingerlings visually forming a single one, which is considered the main challenge of the detection and counting of fingerlings. For future works, we suggest applied the proposed approach in images with an even more dense number of fingerlings. Moreover, we suggest testing the developed method using images capture by a camera with less resolution to verify its generalization ability to detect and count the fingerlings in a sequence of images. Fingerling tracking is also a future work to assist the individual fingerling counts.



(a) #211 (b) #212 (c) #213

(d) #214 (e) #215 (f) #216

(g) #217 (h) #218 (i) #219

Figure 3.10: Sequence of frames illustrating one of the main challenges for the proposed method.

CHAPTER

4

MTLSegFormer: Multi-task learning with Transformers for Semantic Segmentation

Preface

Multi-task learning has proven to be effective in improving the performance of correlated tasks, increasing generalization power. Most of the existing methods use a backbone to extract initial features followed by independent branches for each task, and the exchange of information between the branches usually occurs through the concatenation or sum of the feature maps of the branches. However, this type of information exchange does not directly consider the local characteristics of the image nor the level of importance or correlation between the tasks. In this paper, we propose a semantic segmentation method, MTLSegFormer, which combines multi-task learning and attention mechanisms. After the backbone feature extraction, two feature maps are learned for each task. The first map is proposed to learn features related to its task, while the second map is obtained by applying learned visual attention to locally re-weigh the feature maps of the other tasks. In this way, weights are assigned to local regions of the image of other tasks that have greater importance for the specific task. Finally, the two maps are combined and used to solve a task. We tested the performance in two challenging problems with correlated tasks and observed a significant improvement in accuracy mainly in tasks with high dependence on the others.

Relation of the paper with MTL. In this third paper we combine the MTL approach with transformers. Transformers learn weights that indicate the importance of each local region of the image for a specific task, as well as the importance of each task to the others. As in previous papers, we used the hybrid strategy to share information between tasks. Hard parameter sharing occurs because the tasks have the same backbone. Soft parameter sharing also occurs because there is an exchange of information between the branches/tasks, but in this case, each task has a weight that indicates the level of information exchange between them. Therefore, if the learned weight is null, the strategy becomes hard parameter sharing (only with the shared backbone).

4.1 Introduction

Semantic segmentation is one of the fundamental tasks in computer vision, being essential in a variety of applications such as land cover mapping (Ienco et al., 2019; Zhang et al., 2019), autonomous cars (Siam et al., 2017), medical applications (Liu et al., 2022), among many others. In semantic segmentation, the objective is to assign a class to each image pixel, a task recently performed with Convolutional Neural Networks (CNN) due to its promising results. Even with advances in the field (Yuan et al., 2020; Chen et al., 2018, 2017; Ronneberger et al., 2015b), semantic segmentation remains a difficult task due to intra-class variation, context variation among other factors (Strudel et al., 2021).

In recent years, due to the success in natural language processing (NLP), there is a great interest in applying Transformers in computer vision (Xie et al., 2021). Visual Transformer (ViT), proposed by Dosovitskiy et al. (2021), was the first Transformed-based network to achieve state-of-the-art results for visual related tasks. In ViT, the image is split into multiple linearly embedded patches and used as input to a Transformer with positional embedding, achieving outstanding results in the ImageNet dataset (Xie et al., 2021). Since ViT, several transformer-based networks with prominent results have been proposed for image classification (Yuan et al., 2021; Chu et al., 2021b; Chen et al., 2021), object detection (Carion et al., 2020; Zhu et al., 2021) and image segmentation (Strudel et al., 2021; Xie et al., 2021; Zheng et al., 2021; Ranftl et al., 2021).

SETR, proposed by Zheng et al. (2021), was one of the first Transformer-based network to show the potential of Transformer for semantic segmentation. Further, other advances were done with recent networks such as the pyramid vision Transformer (PVT) proposed by Wang et al. (2021), Swin Trans-

former (Liu et al., 2021) and Twins (Chu et al., 2021a). More recently, SegFormer, proposed by Xie et al. (2021), redesign the encoder and the decoder introducing a positional-encoding-free and hierarchical Transformer encoder and a decoder based on Multi-layer perception, achieving state-of-the-art efficiency, accuracy and robustness for semantic segmentation.

In addition to modeling a single task, Transformer-based methods provided more robust solutions for Multi-task learning (MLT) compared to traditional CNNs (Zhou et al., 2021a). In MLT, multiple tasks are trained simultaneously, sharing representation between the tasks to learn common ideas (Crawshaw, 2020). Therefore, the goal is to improve the performance of the tasks with no distinction between them (Zhang and Yang, 2021). For semantic segmentation, several studies have combined CNN and multi-task learning (Osco et al., 2021a; Gonçalves et al., 2021; Zhou et al., 2021b; Ke et al., 2021). Nevertheless, to the best of our knowledge, no work combined MTL with Transformers for semantic segmentation tasks, even though MLT Transformers models have shown strong performance for other domains, such as image classification and language tasks (Hu and Singh, 2021).

Here, we propose MTLSegFormer, which is a multi-task semantic segmentation method with Transformers. MTLSegFormer is composed of two main modules, encoder, and decoder, similar to SegFormer (Xie et al., 2021). The encoder is composed of hierarchical Transformers that generate low and high-resolution features to represent the input image and feed the decoder. The original contribution is the sharing of features between the tasks in the decoder. For this, our decoder extracts two feature maps for each task. The first feature map is obtained from the encoder and can be understood as features learned specifically for a given task. The second feature map is a shared representation whose purpose is to benefit from features extracted from other tasks. With the use of Transformers, a given task can differentially weight the importance of features from other tasks to compose the second feature map. Both feature maps of a given task are summed and used for image semantic segmentation. We compare the proposed method with the state-of-the-art in two new datasets whose tasks/classes are complementary. Experimental results showed the superiority of the proposed method and the importance of exchanging information between complementary tasks.

In summary, our original contributions are described as follows:

1. Development of a new MTL semantic segmentation method with sharing of features between tasks through Transformers, an approach with global attention that has contributed with significant advances in several areas;

2. Results superior to the state-of-the-art in the segmentation of two datasets, showing the importance of exchanging information between tasks;

3. Construction and labeling of two datasets, one for segmenting crop line and gap, and another for segmenting leaf and defoliation. Both datasets have complementary tasks and can be used to evaluate the inclusion of context in segmentation methods.

4.2 Related Work

4.2.1 Semantic Segmentation

In the deep learning age, FCN (Shelhamer et al., 2017) is one of the first and most important networks for image segmentation. FCN is based on purely convolutional layers, therefore the output of the network, the segmentation map, has the same size as the input. Improvements have been done to the deep learning architecture using Conditional and Markov Random Field (Zheng et al., 2015; Liu et al., 2015). Efforts have been made to resolve some limitations of the FCN, such as the lack of context and size of the receptive field. DeepLabV3+ (Chen et al., 2018) and Dilation (Yu and Koltun, 2016) worked on enlarging the receptive field. Other networks focus on context modeling for a better scene comprehension (OCRNet (Yuan et al., 2020)). According to Zheng et al. (2021), more recently, attention models, such as (Zhao et al., 2018a; Fu et al., 2019; Huang et al., 2020), became popular for capturing long range context information, but still based on FCNs, such as VGG (Simonyan and Zisserman, 2015) and ResNet (He et al., 2016).

4.2.2 Transformers for Semantic Segmentation

Axial-deeplab, proposed by Wang et al. (2020), removed all convolutions from the network and took advantage of attention for image classification, but maintaining the FCN design. SETR (Zheng et al., 2021) used the ViT as the backbone and a standard CNN decoder in a sequence-to-sequence model that keeps the same image resolution. Swin Transformer (Liu et al., 2021) used a modification of the ViT and an Upper-Net as a decoder. Segmeter (Strudel et al., 2021) combined a ViT backbone and a mask decoder inspired by DETR (Carion et al., 2020). SegFormer (Xie et al., 2021) used a Transformer encoder with multiscale features and eliminated the need for positional encoding as the decoder is an MLP that aggregates information from different layers, combining local and global attention.

4.2.3 Multi-task learning

Generally, there are two main types of multi-task learning (MLT) models, hard (Caruana, 1993) and soft (Duong et al., 2015) parameter sharing. Hard parameter sharing is most commonly used since there is a low risk of overfitting once the tasks are learned simultaneously, improving generalization (Baxter, 1997) and working better for closely related tasks. For soft parameter sharing, each task have specific hidden layers and parameters (Vafaeikia et al., 2020), however, parameters are regularized (Ruder, 2017). Another type of MLT model, the Multilinear Relationship Networks (MRNs) (Long et al., 2017), uses a common CNN and a fully connected layer to features shared between the tasks and separate stacks of fully connected layers for each individual task. Eigen and Fergus (2014), proposed a multi-scale FCN for semantic labels, depth and surface normals, but trained separately. Jafari et al. (2017) used a joint refinement network, using as input two separate networks trained for depth and semantic prediction, to improve both results using cross-modality influences. Maninis et al. (2019), applied a shared encoder along with soft attention modules to train a network for multiple tasks, and trained each task separately. Osco et al. (2021a), proposed a multi-stage MLT for line and point detection using a VGG19 (Simonyan and Zisserman, 2015) as the backbone and stacks of fully connected layers for each task with shared volumes between the stacks. Related to Transformers, several studies explored the potential of MLT. Kaiser et al. (2017), showed that a Transformer based encoder-decoder network can be used for different input and output domains. Khan et al. (2020), proposed a MLT Transformed-based model for slot tagging, considering each slot type as a problem. Further, Hu and Singh (2021) proposed the UniT (Unified Transforme model). UniT is based on an encoder-decoder Transformer to learn tasks from different domains, such as objective detection and natural language. Nevertheless, to the best of our knowledge, no work applied MLT-Transformed based model for semantic segmentation only.

4.3 Proposed Method

The proposed segmentation method can be divided into two main modules, encoder and decoder, as illustrated in Figure 4.1. Given an input image with resolution $h \times w \times 3$, the first step is to split this image into patches. The encoder receives patches and generates feature maps at different scales through the self-attention mechanism. Finally, the decoder combines the feature maps to produce the segmentation using a new multi-task block. This block is able to exchange and learn features between tasks to generate image segmentation

with contextual information.

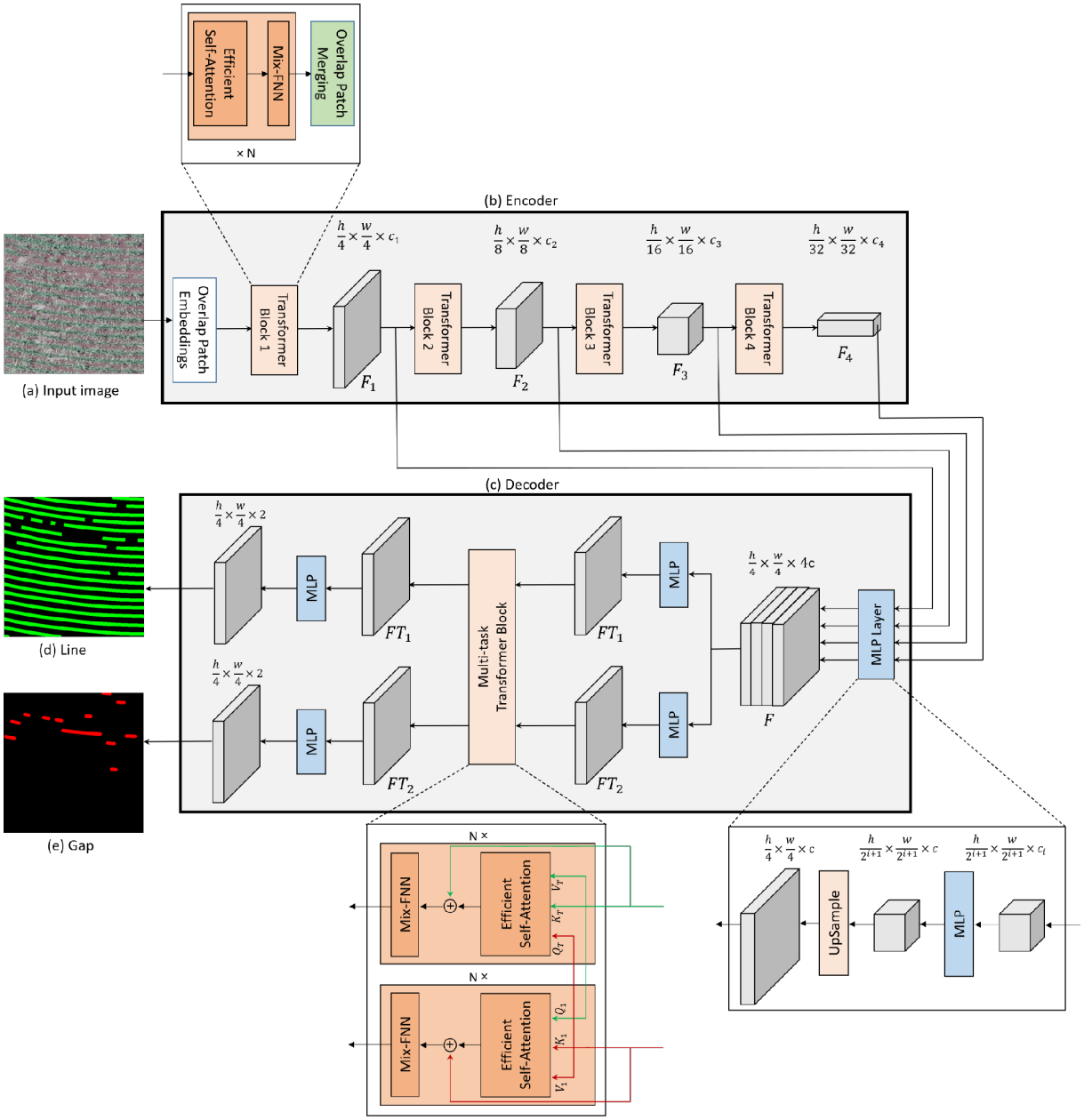


Figure 4.1: Illustration of the method proposed in this work where (a) corresponds to the input image, (b) Encoder, (c) Decoder, (d) segmentation result for the crop lines and (e) for the gaps.

4.3.1 Encoder

The main idea of the encoder is to generate feature maps at hierarchical levels of the image. Following SegFormer (Xie et al., 2021), four maps $F_1, \dots F_4$ are generated with a resolution of 1/4, 1/8, 1/16, 1/32 of the input image resolution. For this, the input image is divided into overlapping patches of size 7×7 using stride $s = 4$ and padding $p = 3$ (Overlap Patch Embeddings in Figure 4.1(b)). This process results in $n = \frac{H}{2} \times \frac{W}{2}$ patches whose feature vector is

the concatenation of raw pixel RGB values. Then, the patches are used in the Transformer Block composed of three steps, Efficient Self-Attention, Mix-FNN and Overlap Patch Merging, as shown in Figure 4.1(b).

Efficient Self-Attention. This block is composed of the multi-head self-attention process. In this process, the n input patches are subjected to a linear projection layer to obtain keys K, queries Q and values V, all with dimension $n \times c$. Given Q and K, the attention A with dimension $n \times n$ is obtained according to Equation 4.1. The attention weights A_{ij} are calculated based on the similarity between each pair (q_i, k_j).

$$A = softmax\left(\frac{QK^T}{\sqrt{d}}\right), \tag{4.1}$$

Given the attention weights A and the values V, improved features V' are obtained through the weighted sum presented in Equation 4.2. These features describe patches using global attention on the image due to the A calculation between all patches.

$$V' = AV \tag{4.2}$$

To reduce the computational complexity of this block, SegFormer uses a reduction process (Wang et al., 2021) of the keys K by a factor r. The keys of r neighboring patches (considering their positions in the 2D image) are concatenated to generate a single key with dimension rc. Thus, the resolution of K is reshaped from $n \times c$ to $\frac{n}{r} \times rc$. A linear layer projects the keys with dimension rc back to dimension c. Therefore, the new keys K have dimension $\frac{n}{r} \times c$, which is reduced by a factor r. As a result, the complexity of the attention mechanism is reduced from $O(n^2)$ to $O(\frac{n^2}{r})$.

Mix-FNN. SegFormer replaced the traditional positional encoding in transformers with a block called Mix-FNN. This block receives the features of the self-attention module x_{in} and basically applies a convolution layer with filters of size 3×3 and activation function GELU according to Equation 4.3. SegFormer showed that a convolution layer is sufficient to add positional information to patches.

$$x_{out} = \text{MLP}(\text{CONV}(\text{MLP}(x_{in}))) + x_{in}, \tag{4.3}$$

where MLP is a multilayer perceptron.

Overlap Patch Merging. The patch features learned in the previous steps can be organized in 2D and their resolution reduced by a convolution layer. For this, filters of size $k = 3$, stride $s = 2$ and padding $p = 1$ are defined in the convolution layer to perform overlapping patch merging. Therefore, the feature map dimension is halved, for example, from $\frac{h}{2} \times \frac{w}{2} \times c_1$ to $\frac{h}{4} \times \frac{w}{4} \times c_2$. This process

is important for generating multi-level features such as CNNs.

In this work, two encoder configurations, called B0 and B5, were used in the experiments to assess the representation power of the input image. The main hyperparameter is the dimension of the channels $c_1, \ldots, c_4$, which for B0 and B5 are respectively $\{32, 64, 160, 256\}$ and $\{64, 128, 320, 512\}$. More details can be obtained in SegFormer (Xie et al., 2021).

4.3.2 Decoder

Unlike SegFormer, this work proposes a multi-task decoder, since the probability of occurrence of a class or task can be correlated with the existence of another. Thus, the advantage of our decoder is to guarantee the exchange of information to provide global features between tasks. Here, each task can be a single class or a group of classes. As detailed below, the proposed decoder is composed by the concatenation of the encoder features using the MLP Layer followed by a new Multi-task Transformer Block.

MLP Layer. This block merges the four hierarchical encoder feature maps, i.e., $F_1, \ldots F_4$. Initially each map is given as input to an MLP in order to unify the number of channels for c. Then the maps are scaled up so that their dimension is $1/4$ of the input image and then concatenated to provide a feature map F with resolution $\frac{h}{4} \times \frac{w}{4} \times 4c$.

Multi-task Transformer Block. Given the concatenated map F, an MLP is used to learn the initial feature maps FT_t for each task, creating T branches. Figure 4.1(c) presents the example for two tasks ($T = 2$) used in our application. The feature map FT_t could be used to segment each task, however, there is no direct exchange of information between them. To include this mutual exchange of information, the feature maps $FT_t \mid t \in \{1, \ldots, T\}$ are given as input to the Multi-task Transformer Block. In this block, F_t is used to obtain the keys K_t, queries Q_t and values V_t for each task t. Then, the information exchange occurs by using a query Q_t with keys K_u and values V_u of other tasks ($u \neq t$) through the Efficient Self-Attention block.

$$V_t^u = softmax\left(\frac{Q_t K_u^T}{\sqrt{d}}\right) V_u, \text{ for all } u \in \{1, \ldots, T\}, u \neq t \tag{4.4}$$

Figure 4.1(c) illustrates the ideia to obtain multi-task features. The idea is that a task can ask "questions" for other tasks to learn the context of the image and the correlation between them. For example, the segmentation of a gap pixel can benefit from by knowing the direction and characteristics of the crop lines in the image. The multi-task features and initial features F_t are summed and used in the Mix-FNN to include positional information as well as in the encoder.

$$F_t = F_t + V_t^u, \text{ for all } u \in \{1, \ldots, T\}, u \neq t \tag{4.5}$$

Finally, a feature map F_t for each task is provided by the block, which is enriched with the exchange of information between tasks. Finally, each feature map is used to predict segmentation masks through an MLP.

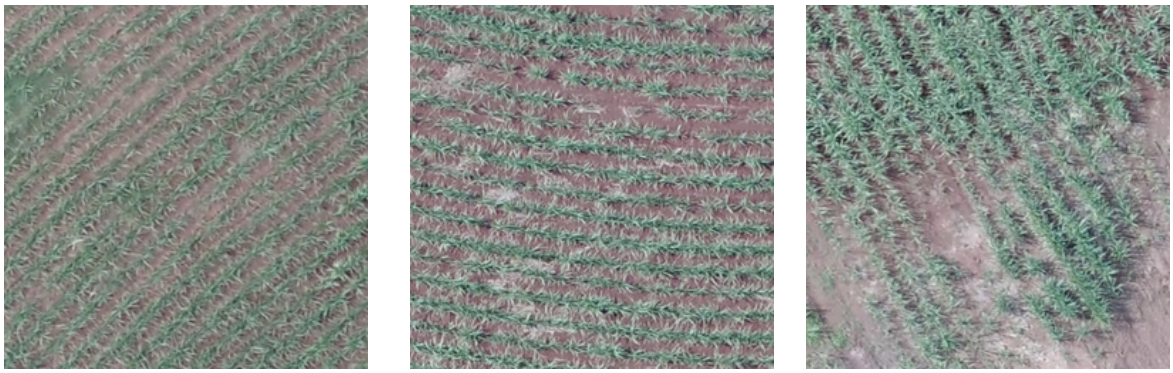
4.4 Experiments and Results

4.4.1 Experimental Setup

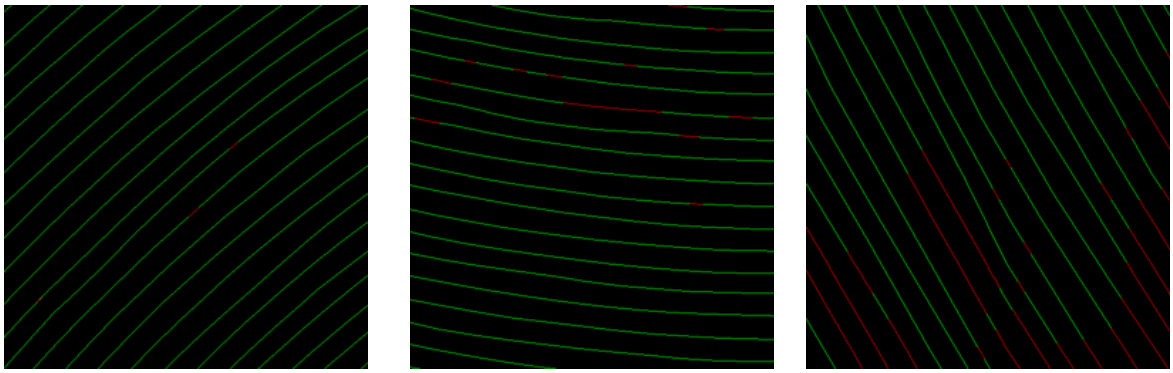
Image Datasets. In this work, we propose two datasets in which the tasks are complementary. The first dataset aims to segment crop lines and gaps, namely Crop Line Dataset. This dataset is relevant because gap segmentation is a challenge due to its similarity with the background and, therefore, it is necessary to estimate the direction of the lines for adequate gap segmentation (see Figure 4.2). The second dataset (Defoliation Dataset) consists of segmenting the leaf and defoliation, a region of the leaf deteriorated by pests. Thus, segmenting the defoliation is a challenging task due to the visual similarities with the background. The challenge is even greater when defoliation occurs at the edge of the leaf, as it is essential to know the leaf shape.

The crop line dataset is composed of three farms in Brazil with sugarcane plantations. Each farm is composed of several plots and, for reasons of processing power, the orthophotos were generated for each plot separately. The images were captured by a UAV with an RGB camera and the orthophotos were generated with Pix4D commercial software. Each orthophoto was manually labeled by an expert with the crop lines and gaps. The labeling process was performed in QGIS open-source software.

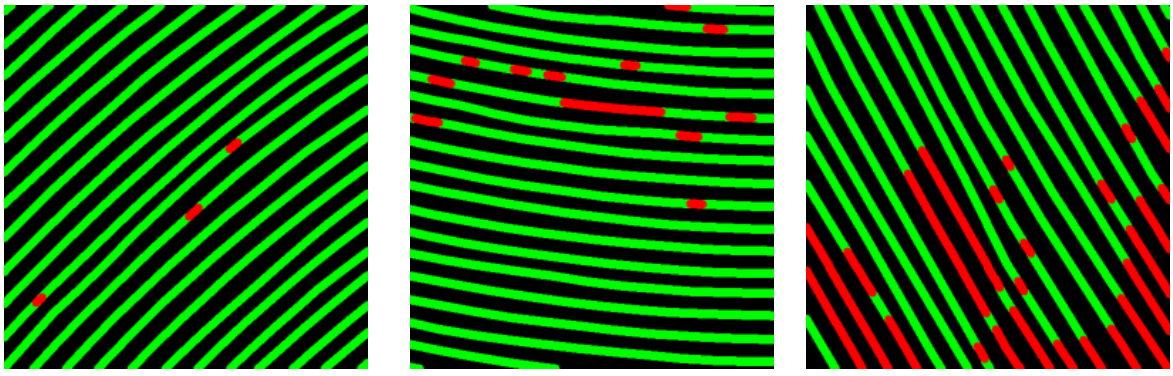
The three farms were used as training, validation and test sets. It is important to emphasize that the farms are in different regions to assess the generalizability of the methods. The largest farm with 61 plots was used for training. The farm used in the validation set has 4 plots, while the test farm is composed of 7 plots. As the segmentation methods receives fixed-size images as input, the training and validation orthophotos were respectively divided into 4669 and 574 patches of 512×512 pixels without overlapping. Figures 4.2(a) and 4.2(b) show examples of patches and their respective labeling. As the lines and gaps are one pixel thick, during training, we dilate the labels with a structuring element of size 6 (Figure 4.2(c)) to avoid the imbalance of the classes with the background. The orthophotos of the seven test plots are divided into 3824 patches of 512×512 pixels and processed as described below to obtain a prediction for the entire plot.



(a) RGB



(b) Labeling



(c) Dilated Labeling

Figure 4.2: Examples of patches (a) RGB, (b) labeling with crop lines in green and gaps in red, and (c) dilated labeling.

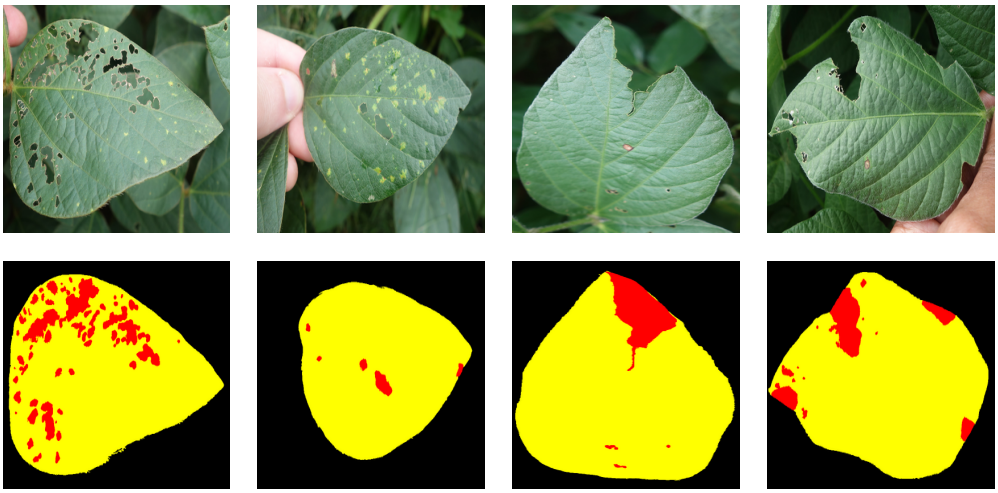


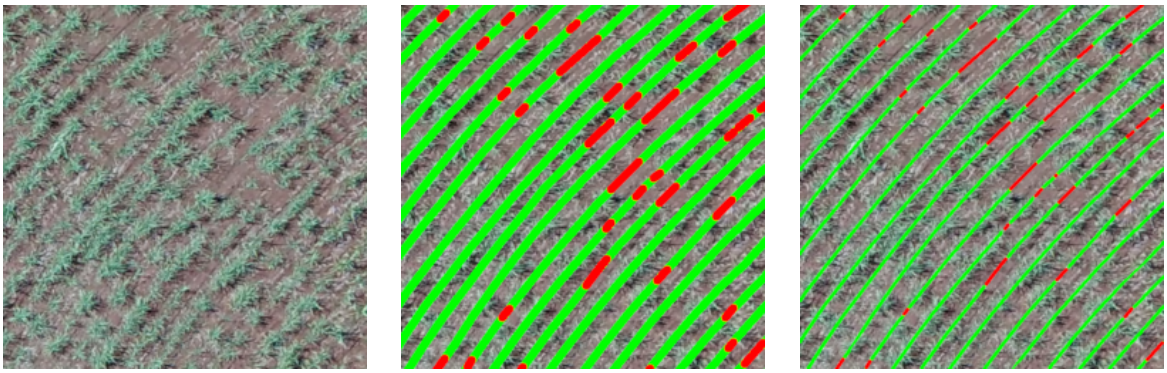
Figure 4.3: Examples of the dataset with leaf and defoliation classes in yellow and red, respectively.

Defoliation dataset consists of 320 images of soybean leaves obtained through PlantVillage (Hughes et al., 2015; da Silva et al., 2019). Photographs were taken at different resolutions using a cell phone in the field with no brightness control. Each image has a leaf in the foreground, however, they have a missing part caused by pests. Thus, the objective of this dataset is to estimate the leaf and defoliation area to obtain a percentage of deterioration. The greater the deterioration, the greater the amount of pests that are attacking the crop. As a result, high defoliation drastically reduces the photosynthesis process and consequently the plant's grain production.

Each image was manually segmented into leaf, defoliation and background as shown in the examples in Figure 4.3. We can see that segmenting the defoliation area is a challenge due to its similarity to the background, especially in leaf edge regions. In these cases, the methods need to predict the shape based on the rest of the leaf. Due to the amount of images, the training, validation and test sets followed the 5-fold cross-validation process. As the images have different resolutions, they have been resized to 512×512 pixels.

Training and Testing. The encoder was initialized with the pre-trained weights on the Imagenet-1K (Xie et al., 2021) dataset and the decoder was initialized randomly. Following SegFormer, we trained our method using the AdamW optimizer for 80K iterations using a batch size of 2, initial learning rate of 0.00006 updated by a Poly LR schedule with a factor of 1. Our method was implemented in Python with the mmsegmentation[1] codebase. The experiments were performed on a computer with Intel (R) Xeon (E) E3-1270@3.80GHz CPU, 64 GB memory, and an NVIDIA Titan V graphics card, that includes 5120 CUDA (Compute Unified Device Architecture) cores and 12

[1]https://github.com/open-mmlab/mmsegmentation



(a) RGB Image (b) Prediction (c) Skeleton

Figure 4.4: Example of the (a) RGB patch, (b) prediction and (c) one-pixel thick skeletonization used to calculate the segmentation and detection metrics, respectively.

GB of graphics memory.

In the test of Crop Line Dataset, an orthophoto is split into patches of 512×512 pixels with 50% overlap. The overlay helps in identifying the lines present at the edge of the patch and in its continuity with neighboring patches. The patches are given as input to the method that performs the prediction for each pixel. Finally, the predictions are combined to obtain the full orthophoto segmentation. Due to patch overlap or multi-task module, a pixel can have more than one prediction and in these cases the class priority follows gap, line and background. This priority is related to the inherent difficulty of each class. For the line detection assessment (see section below), we apply skeletonization (Lee et al., 1994) on the resulting orthophoto segmentation. Skeletonization shrinks the blobs of each class to 1-pixel-thick representations. Figure 4.4 presents an RGB example, prediction and skeletonization.

Metrics. To evaluate the methods, we use two types of metrics. The first type is composed of segmentation metrics widely used in the literature, such as F1-score and Intersection over Union (IoU). For Crop Line Dataset, we also used a second type of metrics, since crop lines and gaps are usually represented by a one-pixel-thick line. Thus, the segmentation metrics whose lines and gaps were dilated do not directly evaluate the performance in the application. Therefore, in addition to the segmentation metrics, we use the F1-score to evaluate the detection of one-pixel thick lines in both prediction and ground truth of the Crop Line Dataset. For this, we calculate the True Positives (TP) as the number of pixels predicted as a line and that are within a maximum distance d of a pixel labeled as a line. We can understand d as the maximum error and in this work it was equal to 3, which is half the width of the plantation line. False Positives (FP) correspond to the number of predicted pixels that are not close (distance d) of any pixel labeled as a line. On the other hand,

Table 4.1: Result for two sizes of encoders for segmentation of lines and gaps.

Encoder Size	Params	Flops	Line		Gap	
			F1-score	IoU	F1-score	IoU
B0	2.55 M	1.8	0.8117	0.6838	0.7478	0.5978
B5	61.41 M	47.0	0.8257	0.7038	0.7857	0.6478

Table 4.2: Comparative results of the decoder channel dimension c.

Channel Dimension	Line		Gap	
	F1-score	IoU	F1-score	IoU
128	0.7975	0.6647	0.7222	0.5659
256	0.8257	0.7038	0.7857	0.6478
512	0.8345	0.7165	0.7704	0.6278

False Negatives (FN) correspond to the number of pixels labeled as a line that are not close to any predicted pixel. With TP, FP and FN, F1-score can be estimated for the crop lines and similarly for the gaps.

4.4.2 Ablation Study

In this section, we compare the influence of different choices of the proposed method using Crop Line Dataset, including encoder size, decoder channel dimension, and multi-task.

Encoder size. Initially, we evaluated the influence of encoder size on segmentation. Table 4.1 presents the results (F1-score and IoU) for segmentation of lines and gaps for two encoder sizes (B0 and B5). As expected, the number of parameters and operations required by B5 encoder is higher compared to B0 encoder. In terms of accuracy, it is possible to observe that increasing the size of the encoder reflects an increase in both the F1-score and the IoU. The increase is greater for gaps (e.g., F1-score from 0.7478 to 0.7857), since this class is a minority in the dataset and, in general, a more powerful model is needed to adequately represent it. It is also important to emphasize that the proposed method with B0 encoder presents competitive results with the state-of-the-art that use computationally heavier backbones (see Table 4.4).

Decoder channel dimension. We also evaluated the influence of the decoder channel dimension c as shown in Table 4.2. When increasing from 128 to 256, the results for both lines and gaps were superior. On the other hand, for $c = 512$, the results were higher for lines, but slightly lower for gaps. Thus, we chose to keep $c = 256$ due to results and computational cost.

Multi-task decoder. Finally, we compare the multi-task decoder proposed in this work with the original SegFormer decoder as presented in Table 4.3. The decoder parameters were the same, except that the one proposed here uses the exchange of information between tasks through the Multi-task Trans-

Table 4.3: Comparative results between the decoder of our method and the SegFormer.

Decoder	Line		Gap	
	F1-score	**IoU**	**F1-score**	**IoU**
SegFormer	0.8074	0.6778	0.7115	0.5525
Our	**0.8117**	**0.6838**	**0.7478**	**0.5978**

former Block. For the encoder of both, we use the B0 version.

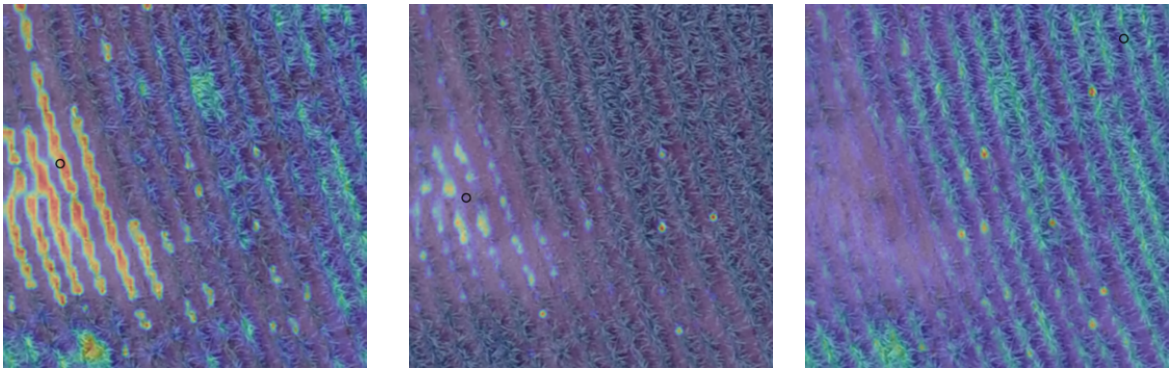
The results show that the multi-task decoder proposed here was superior to the SegFormer decoder both in segmentation of lines and gaps for both metrics. When analyzing the results by task, we can see that there is a greater gain in the gap results (e.g., F1-score from 0.7115 obtained by SegFormer to 0.7478 obtained by ours). In general, segmenting gaps is a more complex task because their visual characteristics are similar to those of the background. In fact, a gap can only be segmented properly if the method is able to understand the crop lines in the image. When predicting gaps, the results suggest that our decoder is able to benefit from the lines because of the exchange of information between tasks.

To corroborate our decoder, we plotted attention weights in the Multi-task Transformer Block (see softmax in Equation 4.4). Figure 4.5(a) presents the RGB image while Figures 4.5(b) and 4.5(c) present the attention weights on gap and plantation line branches, respectively. To facilitate visualization, attention weights were calculated for a given pixel demarcated with a black circle. We can observe that, when considering a gap pixel (figures on the left), the attention weight of the other pixels was adequate for both tasks. For example, Figure 4.5(b) on the left shows that attention is highest on pixels that are likely to be gap, even though this region is similar to the background. The same pixel in the crop line branch has attention with plant pixels (Figure 4.5(c) on the left). It is also interesting to note, according to the figures in the center, that a background pixel with features similar to gap pixels, presents a completely different attention weights. Finally, the figures on the right show a crop line pixel and the attention weights on the two tasks.

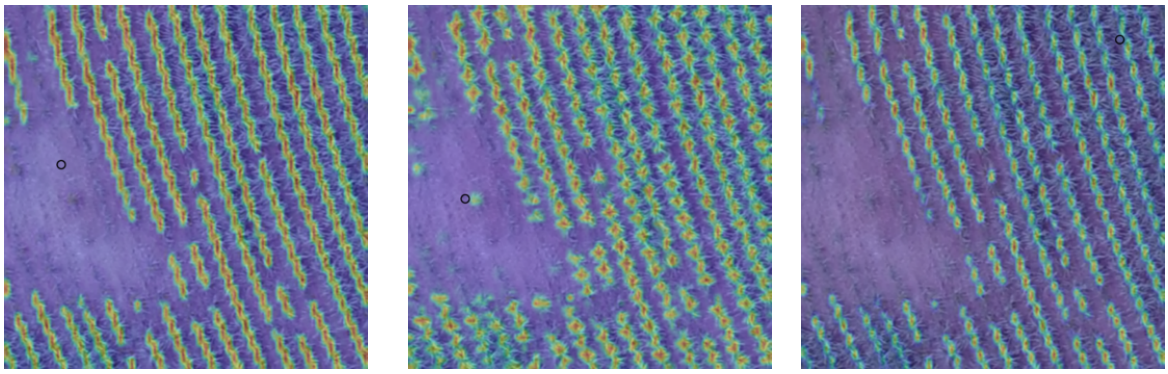
4.4.3 Crop Line Dataset

The results of the proposed method were compared with the state-of-the-art in Table 4.4 for segmentation metrics and in Table 4.5 for line detection metrics. For segmentation metrics (Table 4.4), the proposed method outperformed other methods, including SegFormer, DeepLabV3+ and OCRNet. Considering the crop lines, our method reached F1-score and IoU of 0.8257 and 0.7038 against 0.8192 and 0.6950 from OCRNet. Our method also obtained the best

(a) RGB Image



(b) Gap Task



(c) crop Line Task

Figure 4.5: Example of our decoder's attention weight on important regions of the image. The first, second and third columns of images present the weights related to a gap, background and line pixel, respectively.

results for the gaps, and DeepLabV3+ obtained the second best result for this task.

Table 4.4: Comparison with state-of-the-art methods using segmentation metrics.

Method	**Line**		**Gap**	
	F1-score	**IoU**	**F1-score**	**IoU**
SegFormer (Xie et al., 2021)	0.7793	0.6409	0.7331	0.5790
FCN (Shelhamer et al., 2017)	0.8044	0.6739	0.7363	0.5831
OCRNet (Yuan et al., 2020)	0.8193	0.6950	0.7649	0.6199
DeepLabV3+ (Chen et al., 2018)	0.8168	0.6926	0.7846	0.6468
Proposed method	**0.8257**	**0.7038**	**0.7857**	**0.6478**

Although the results with segmentation metrics are important, it is also important to compare the methods considering line and gap detection metrics (Table 4.5). The methods are able to detect most of the crop lines (second column of the table), with SegFormer, the proposed method and OCRNet presenting the best results. On the other hand, for gap detection, SegFormer reduces the result with F1-score below the proposed method and DeepLabV3+. In the average of the two tasks, the proposed method presents the best results followed by OCRNet, DeepLabV3+ and SegFormer. These results corroborate the accuracy of the proposed method compared to the state-of-the-art, presenting superior results in both segmentation and detection metrics.

Table 4.5: Comparison with state-of-the-art methods using line detection metrics.

Method	**F1-score (Line)**	**F1-score (Gap)**	**Average** (Line/Gap)
SegFormer (Xie et al., 2021)	**0.9804**	0.8673	0.9239
FCN (Shelhamer et al., 2017)	0.9668	0.8686	0.9177
OCRNet (Yuan et al., 2020)	0.9746	0.8948	0.9347
DeepLabV3+ (Chen et al., 2018)	0.9581	0.9049	0.9315
Proposed method	0.9793	**0.9064**	**0.9429**

Figure 4.6 shows the qualitative results of the methods for segmenting crop lines and gaps. Each row of figures presents (a) RGB, (b) ground truth, and the results of (c) FCN, (d) SegFormer, (e) OCRNet, (f) DeepLabV3+, and (g) MTLSegFormer. In general, the methods present robust results for crop lines and gaps even in curves, as shown in the first column of examples in Figure 4.6. On the other hand, the proposed method stands out in regions without sufficient visual information for segmentation and, therefore, it would be necessary to estimate the direction of the lines, as evidenced in the second column of examples. We can see that the shade makes it difficult to identify the plants and the continuity of the crop line for most methods. The same problem occurs for large regions with gaps due to their visual similarity to the background, as

shown in the third column of examples. Also in this example, some methods have difficulty in identifying isolated plants, such as SegFormer.

4.4.4 Defoliation Dataset

The comparative results for leaf and defoliation segmentation are presented in Table 4.6. The leaf segmentation in the image is a simpler task than the defoliation, therefore, the methods present consistent results in this task. From the first two columns, we can see that the methods present results above 0.97 for F1-score and IoU, except for OCRNet. On the other hand, defoliation segmentation is a challenging task as the visual characteristics are identical to the background. Also, when there is defoliation at the leaf edge, the methods need to estimate the shape properly. These challenges corroborate the advantage of the proposed method, which is able to exchange information with the leaf to estimate its shape and then segment the internal defoliation and especially the defoliation at the edge. The proposed method achieved the best results for defoliation, followed by SegFormer, which also provides global attention, although there is no direct exchange of information between tasks. The other traditional methods, even with object-contextual representations such as OCRNet, presented inferior results, which demonstrates the effectiveness of the proposed method.

Table 4.6: Comparison with state-of-the-art methods for defoliation segmentation.

Method	**Leaf**		**Defoliation**	
	F1-score	**IoU**	**F1-score**	**IoU**
SegFormer (Xie et al., 2021)	0.9850	0.9706	0.8779	0.7898
FCN (Shelhamer et al., 2017)	0.9847	0.9701	0.8447	0.7411
OCRNet (Yuan et al., 2020)	0.9310	0.9040	0.7259	0.6127
DeepLabV3+ (Chen et al., 2018)	0.9837	0.9702	0.8475	0.7507
Proposed method	**0.9869**	**0.9743**	**0.8877**	**0.8048**

Qualitative examples of leaf and defoliation segmentation are presented in Figure 4.7, with each row presenting respectively (a) RGB, (b) ground truth, and the results of (c) FCN, (d) SegFormer , (e) OCRNet, (f) DeepLabV3+, and (g) MTLSegFormer. The first and second examples (first two columns) show severe and mild defoliation, mostly in inner regions of the leaf. In this case, the methods present satisfactory results, although the proposed method was able to segment small defoliation regions as in the second example. The main challenge occurs in edge defoliation, as in the third and fourth examples. The proposed method was able to segment the defoliation by following the leaf shape properly, while the other methods presented some difficulty in one or the other example. It is also important to emphasize that the methods were

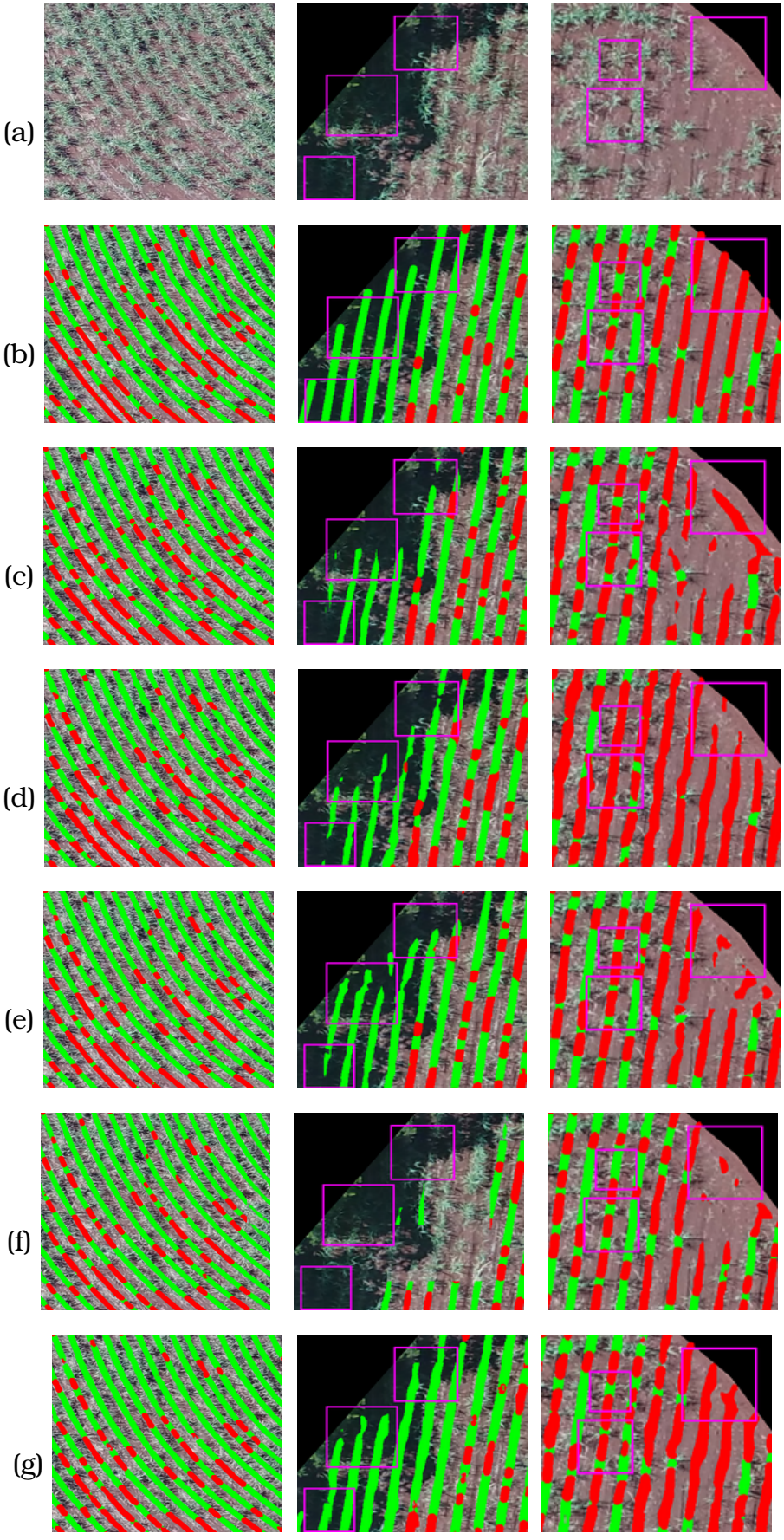


Figure 4.6: Examples of crop line dataset. (a) RGB, (b) ground truth, (c) FCN, (d) SegFormer, (e) OCRNet, (f) DeepLabV3+, and (g) MTLSegFormer.

effective in segmenting the foreground leaf, although other leaves are in the background.

4.5 Conclusion

In this work we propose a new semantic segmentation method, MTLSegFormer, which performs the exchange of information to increase the accuracy in the segmentation of correlated tasks/classes. For this, multi-scale features are extracted by an encoder. Then, a decoder was proposed to learn new feature maps extracted from other tasks based on attention from Transformers. This attention proved to be effective in determining relevant regions for related tasks.

Experiments were carried out on two new datasets whose tasks/classes are correlated and the results showed the superiority of the proposed method compared to the state-of-the-art in semantic segmentation, including SegFormer. The proposed method excelled in the segmentation of tasks/classes that strongly depend on others, such as defoliation in edge regions that depends on the leaf shape. As future work, we intend to evaluate traditional datasets by defining a set of related classes and using other recently published Transformers.

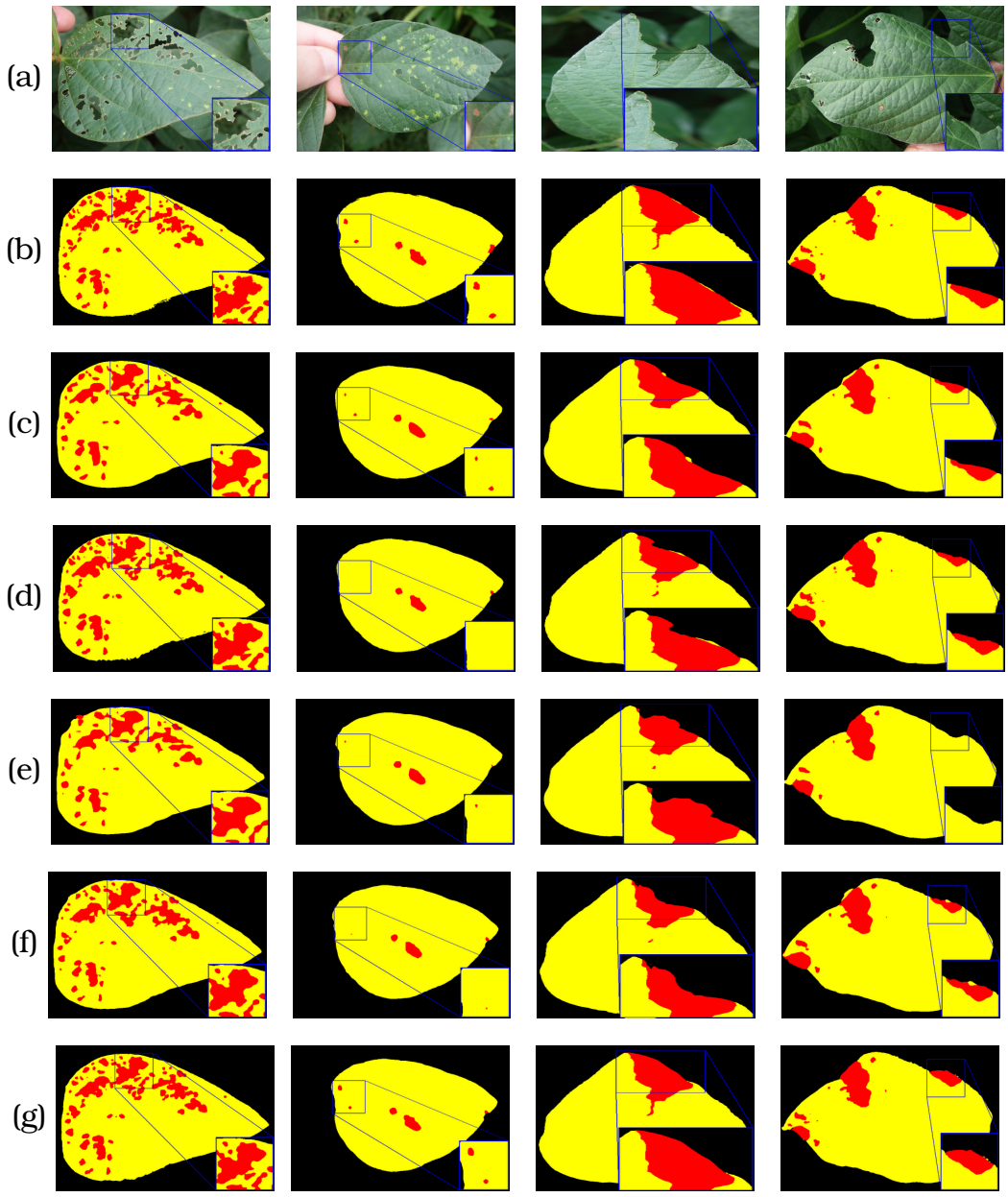


Figure 4.7: Examples of leaf segmentation and defoliation. (a) RGB, (b) ground truth, (c) FCN, (d) SegFormer , (e) OCRNet, (f) DeepLabV3+, and (g) MTLSeg-Former.

CHAPTER

5

Conclusion

Deep learning and multi-task have gained great attention in computer vision, mainly due to the results obtained in the literature. Currently, the combination of these two techniques has improved the performance of several real computer vision problems. In this work, the use of CNNs, transformers and Multi-task learning with several tasks was evaluated in three problems: plantation line detection, fingerling detection and segmentation of crop lines and gaps.

In these applications we propose methods in which each task has a branch in the CNNs and the learning of these tasks occurs simultaneously through MTL characteristics. We also studied ways to extend this learning to temporal information through image sequences. Finally, based on the promising results of attention mechanisms in images, we presented a new semantic segmentation method using MTL and Transformers that allows each task to learn a weight for regions of the image of the other tasks. Thus, high weights indicate that that region is important and should be shared with other tasks.

For the experiments, several image datasets of plantations with different crops were built, including maize, orange, eucalyptus and sugarcane. In addition, an image dataset for fingerling detection was used to assess the learning of temporal tasks. According to the results, it was observed that the use of CNN with MTL is promising in computer vision problems, especially in image detection and segmentation. The results showed that complementary tasks were learned properly and the exchange of information between them improved their individual results. In addition, simultaneous learning tasks was adequate in temporal problems. The results with transformers showed that this sharing is beneficial for tasks, especially those with a strong dependence

on others, achieving results that are superior to the state-of-the-art.

5.1 Contributions

This thesis contributed to the area of multi-task and deep learning applied to images. Specifically, it introduced three methods for sharing information between tasks and their applications to relevant problems. The principal contributions of this thesis are:

- Construction and labeling of image datasets;
- Proposal of multi-task learning methods;
- Extension of the method to temporal problems;
- Inclusion of local sharing between tasks through attention mechanisms;
- Applications of the methods in real and relevant problems;
- Authorship and co-authorship on several papers published or submitted during the doctorate (Gonçalves et al., 2022b; Gonçalves et al., 2021; Gonçalves et al., 2022a; Gonçalves et al., 2020a; Osco et al., 2021a; Bressan et al., 2021; dos Santos de Arruda et al., 2021; Gomes et al., 2020; Miyoshi et al., 2020; Cantero et al., 2020; da Silva et al., 2019; Ribas et al., 2019).

5.2 Future Works

As future works, we suggest the evaluation of the methods in other image datasets with a greater amount of tasks. With the rapid development of CNNs, there have been many well-designed CNN backbones that can replace and improve the initial feature extraction of the proposed methods. Another suggestion for future work is to evaluate the results through a cross-validation and hypothesis testing. Finally, the loss function considers the error of all tasks with the same weight. An alternative to be evaluated is to weigh each task in terms of its importance and difficulty, since some tasks are more difficult to learn than others.

Bibliography

Babapour, H., Mokhtarzade, M., e Zoej, M. J. V. A novel post-calibration method for digital cameras using image linear features. *International Journal of Remote Sensing*, 38(8-10):2698–2716.

Badrinarayanan, V., Kendall, A., e Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495.

Baxter, J. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 1(28):7–39.

Bragman, F., Tanno, R., Ourselin, S., Alexander, D., e Cardoso, J. Stochastic filter groups for multi-task cnns: Learning specialist and generalist convolution kernels. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, páginas 1385–1394.

Bressan, P. O., Junior, J. M., Martins, J. A. C., Gonçalves, D. N., Freitas, D. M., Osco, L. P., de Andrade Silva, J., Luo, Z., Li, J., Garcia, R. C., e Gonçalves, W. N. Semantic segmentation with labeling uncertainty and class imbalance. *CoRR*, abs/2102.04566.

Cai, W. e Wei, Z. Remote sensing image classification based on a cross-attention mechanism and graph convolution. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5.

Cantero, S. V. A. B., Gonçalves, D. N., dos Santos Scabini, L. F., e Gonçalves, W. N. Importance of vertices in complex networks applied to texture analysis. *IEEE Transactions on Cybernetics*, 50(2):777–786.

Cao, Z., Simon, T., Wei, S., e Sheikh, Y. Realtime multi-person 2d pose estimation using part affinity fields. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 1302–1310.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., e Zagoruyko, S. End-to-end object detection with transformers. In *Computer Vision – ECCV 2020*, páginas 213–229. Springer International Publishing.

Caruana, R. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, páginas 41–48. Morgan Kaufmann.

Caruana, R. *Multitask Learning*, páginas 95–133. Springer US, Boston, MA.

Chen, C.-F. R., Fan, Q., e Panda, R. CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification. In *International Conference on Computer Vision (ICCV)*, páginas 357–366.

Chen, L., Papandreou, G., Schroff, F., e Adam, H. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., e Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Ferrari, V., Hebert, M., Sminchisescu, C., e Weiss, Y., editors, *Computer Vision – ECCV 2018*, páginas 833–851. Springer International Publishing.

Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., e Shen, C. Twins: Revisiting the design of spatial attention in vision transformers. In *NeurIPS 2021*, páginas 1–12.

Chu, X., Tian, Z., Zhang, B., Wang, X., Wei, X., Xia, H., e Shen, C. Conditional positional encodings for vision transformers. *CoRR*, abs/2102.10882.

Crawshaw, M. Multi-task learning with deep neural networks: A survey. *CoRR*, abs/2009.09796.

da Silva, L. A., a Bressan, P. O., Gonçalves, D. N., Freitas, D. M., Machado, B. B., e Gonçalves, W. N. Estimating soybean leaf defoliation using convolutional neural networks and synthetic images. *Computers and Electronics in Agriculture*, 156:360–368.

Danelljan, M., Hager, G., Khan, F. S., e Felsberg, M. Convolutional features for correlation filter based visual tracking. In *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, páginas 621–629.

Dian Bah, M., Hafiane, A., e Canals, R. Deep learning with unsupervised data labeling for weed detection in line crops in UAV images. *Remote Sensing*, 10(11):1–22.

Doersch, C. e Zisserman, A. Multi-task self-supervised visual learning. In *2017 IEEE International Conference on Computer Vision (ICCV)*, páginas 2070–2079.

dos Santos de Arruda, M., Osco, L. P., Acosta, P. R., Gonçalves, D. N., Junior, J. M., Ramos, A. P. M., Matsubara, E. T., Luo, Z., Li, J., de Andrade Silva, J., e Gonçalves, W. N. Counting and locating high-density objects using convolutional neural network. *CoRR*, abs/2102.04366.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., e Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, páginas 1–21.

Duong, L., Cohn, T., Bird, S., e Cook, P. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, páginas 845–850. Association for Computational Linguistics.

Dvornik, N., Shmelkov, K., Mairal, J., e Schmid, C. BlitzNet: A Real-Time Deep Network for Scene Understanding. In *ICCV 2017 - International Conference on Computer Vision*, páginas 4174–4182, Venise, Italy.

Eigen, D. e Fergus, R. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *CoRR*, abs/1411.4734.

Fan, L. e Liu, Y. Automate fry counting using computer vision and multi-class least squares support vector machine. *Aquaculture*, 380-383:91–98.

França Albuquerque, P. L., Garcia, V., da Silva Oliveira, A., Lewandowski, T., Detweiler, C., Gonçalves, A. B., Costa, C. S., Naka, M. H., e Pistori, H. Automatic live fingerlings counting using computer vision. *Computers and Electronics in Agriculture*, 167:105015.

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

Gao, Y., Ma, J., Zhao, M., Liu, W., e Yuille, A. L. Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 3200–3209.

Gao, Y., Shi, J., Li, J., e Wang, R. Remote sensing scene classification based on high-order graph convolutional network. *European Journal of Remote Sensing*, 54(sup1):141–155.

Garcia, V., Sant'Ana, D. A., Garcia Zanoni, V. A., Brito Pache, M. C., Naka, M. H., França Albuquerque, P. L., Lewandowski, T., Silva Oliveira Junior, A. D., Araújo Rozales, J. V., Ferreira, M. W., de Queiroz, E. Q. A., Marino Almanza, J. C., e Pistori, H. A new image dataset for the evaluation of automatic fingerlings counting. *Aquacultural Engineering*, 89:102064.

Goldman, E., Herzig, R., Eisenschtat, A., Goldberger, J., e Hassner, T. Precise detection in densely packed scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition*, páginas 5227–5236.

Gomes, M., Silva, J., Gonçalves, D., Zamboni, P., Perez, J., Batista, E., Ramos, A., Osco, L., Matsubara, E., Li, J., Junior, J. M., e Gonçalves, W. Mapping utility poles in aerial orthoimages using atss deep learning method. *Sensors (Switzerland)*, 20(21):1–14.

Gonçalves, D. N., dos Santos de Arruda, M., Pistori, H., Fernandes, V. J. M., Ramos, A. P. M., Furuya, D. E. G., Osco, L. P., He, H., Li, J., Junior, J. M., e Gonçalves, W. N. A deep learning approach based on graphs to detect plantation lines. *ISPRS Journal of Photogrammetry and Remote Sensing (submitted)*, abs/2102.03213.

Gonçalves, D. N., Acosta, P. R., Ramos, A. P. M., Osco, L. P., Furuya, D. E. G., Taís, M., Furuya, G., Li, J., Junior, J. M., Pistori, H., e Gonçalves, W. N. Convolutional neural network for fingerlings counting using a multi-task learning approach (submitted). *Aquaculture*, 1:1.

Gonçalves, D. N., Zamboni, P., Junior, J. M., Pistori, H., e Gonçalves, W. N. Mtlsegformer: Multi-task learning with transformers for semantic segmentation (to be submitted). *Pattern Recognition*, 1:1.

Gonçalves, D. N., de Moares Weber, V. A., Pistori, J. G. B., da Costa Gomes, R., de Araujo, A. V., Pereira, M. F., Gonçalves, W. N., e Pistori, H. Carcass image segmentation using cnn-based methods. *Information Processing in Agriculture*.

Gonçalves, P., Lourenço, B., Santos, S., Barlogis, R., e Misson, A. Computer vision intelligent approaches to extract human pose and its activity from image sequences. *Electronics*, 9:159.

Habib, A., Ghanma, M., Morgan, M., e Al-Ruzouq, R. Photogrammetric and lidar data registration using linear features. *Photogrammetric Engineering and Remote Sensing*, 71(6):699–707.

Hasan, A. S. M. M., Sohel, F., Diepeveen, D., Laga, H., e Jones, M. G. A survey of deep learning techniques for weed detection from images. *Computers and Electronics in Agriculture*, 184:106067.

He, K., Zhang, X., Ren, S., e Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 770–778.

Hong, D., Gao, L., Yao, J., Zhang, B., Plaza, A., e Chanussot, J. Graph convolutional networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 59(7):5966–5978.

Hou, B., Li, J., Zhang, X., Wang, S., e Jiao, L. Object detection and trcacking based on convolutional neural networks for high-resolution optical remote sensing video. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, páginas 5433–5436.

Hu, R. e Singh, A. Unit: Multimodal multitask learning with a unified transformer. In *IEEE International Conference on Computer Vision (ICCV)*, páginas 1439–1449.

Huang, Z., Wang, X., Wei, Y., Huang, L., Shi, H., Liu, W., e Huang, T. S. Ccnet: Criss-cross attention for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:1–1.

Hughes, D., Salathé, M., et al. An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv:1511.08060*.

Ienco, D., Interdonato, R., Gaetano, R., e Ho Tong Minh, D. Combining sentinel-1 and sentinel-2 satellite image time series for land cover mapping via a multi-source deep learning architecture. *ISPRS Journal of Photogrammetry and Remote Sensing*, 158:11–22.

Jafari, O. H., Groth, O., Kirillov, A., Yang, M. Y., e Rother, C. Analyzing modular cnn architectures for joint depth prediction and semantic segmentation. *2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Jou, B. e Chang, S.-F. Deep cross residual learning for multitask visual recognition. In *Proceedings of the 24th ACM International Conference on Multimedia*, pagina 998–1007.

Kaiser, L., Gomez, A. N., Shazeer, N., Vaswani, A., Parmar, N., Jones, L., e Uszkoreit, J. One model to learn them all. *CoRR*, abs/1706.05137.

Kamilaris, A. e Prenafeta-Boldú, F. X. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90.

Ke, R., Bugeau, A., Papadakis, N., Kirkland, M., Schuetz, P., e Schonlieb, C.-B. Multi-task deep learning for image segmentation using recursive approximation tasks. *IEEE Transactions on Image Processing*, 30:3555–3567.

Khan, M. R., Ziyadi, M., e Abdelhady, M. Mt-bioner: Multi-task learning for biomedical named entity recognition using deep bidirectional transformers. *CoRR*, abs/2001.08904.

Kubik, K. Relative and absolute orientation based on linear features. *ISPRS Journal of Photogrammetry and Remote Sensing*, 46(4):199 – 204.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Kumar, A. e Daumé, H. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pagina 1723–1730, Madison, WI, USA. Omnipress.

Lee, C. e Bethel, J. S. Extraction, modelling, and use of linear features for restitution of airborne hyperspectral imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 58(5):289 – 300.

Lee, T., Kashyap, R., e Chu, C. Building skeleton models via 3-d medial surface axis thinning algorithms. *CVGIP: Graphical Models and Image Processing*, 56(6):462–478.

Li, C. e Shi, W. The generalized-line-based iterative transformation model for imagery registration and rectification. *IEEE Geoscience and Remote Sensing Letters*, 11(8):1394–1398.

Lin, Y., Pintea, S. L., e van Gemert, J. C. Deep hough-transform line priors. In *European Conference on Computer Vision*, páginas 1–17.

Liu, S., Johns, E., e Davison, A. J. End-to-end multi-task learning with attention. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 1871–1880.

Liu, X., Yang, L., Chen, J., Yu, S., e Li, K. Region-to-boundary deep learning model with multi-scale feature fusion for medical image segmentation. *Biomedical Signal Processing and Control*, 71:103165.

Liu, Z., Li, X., Luo, P., Loy, C., e Tang, X. Semantic image segmentation via deep parsing network. In *2015 IEEE International Conference on Computer Vision (ICCV)*, páginas 1377–1385.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., e Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, abs/2103.14030.

Lobo Torres, D., Queiroz Feitosa, R., Nigri Happ, P., Elena Cué La Rosa, L., Marcato Junior, J., Martins, J., Olã Bressan, P., Gonçalves, W. N., e Liesenberg, V. Applying fully convolutional architectures for semantic segmentation of a single tree species in urban environment on high resolution uav optical imagery. *Sensors*, 20(2).

Long, J., Shelhamer, E., e Darrell, T. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 3431–3440.

Long, M., CAO, Z., Wang, J., e Yu, P. S. Learning multiple tasks with multilinear relationship networks. In *Advances in Neural Information Processing Systems*, páginas 1593–1602.

Long, T., Jiao, W., He, G., Zhang, Z., Cheng, B., e Wang, W. A generic framework for image rectification using multiple types of feature. *ISPRS Journal of Photogrammetry and Remote Sensing*, 102:161 – 171.

Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., e Feris, R. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. páginas 1131–1140.

Ma, C., Huang, J.-B., Yang, X., e Yang, M.-H. Hierarchical convolutional features for visual tracking. In *The IEEE International Conference on Computer Vision (ICCV)*, páginas 3074–3082.

Ma, F., Gao, F., Sun, J., Zhou, H., e Hussain, A. Attention graph convolution network for image segmentation in big SAR imagery data. *Remote Sensing*, 11(21):1–21.

Maninis, K.-K., Radosavovic, I., e Kokkinos, I. Attentive single-tasking of multiple tasks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 1851–1860.

Marcato Junior, J. e Tommaselli, A. Exterior orientation of cbers-2b imagery using multi-feature control and orbital data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 79:219 – 225.

Misra, I., Shrivastava, A., Gupta, A., e Hebert, M. Cross-stitch networks for multi-task learning. In *Proceedings of (CVPR) Computer Vision and Pattern Recognition*, páginas 3994 – 4003.

Miyoshi, G. T., Arruda, M. d. S., Osco, L. P., Marcato Junior, J., Gonçalves, D. N., Imai, N. N., Tommaselli, A. M. G., Honkavaara, E., e Gonçalves, W. N. A novel deep learning method to identify single tree species in uav-based hyperspectral images. *Remote Sensing*, 12(8).

Nam, H. e Han, B. Learning multi-domain convolutional neural networks for visual tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 4293–4302.

Noh, H., Hong, S., e Han, B. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, páginas 1520–1528.

Osco, L. P., de Arruda, M. d. S., Marcato Junior, J., da Silva, N. B., Ramos, A. P. M., Moryia, É. A. S., Imai, N. N., Pereira, D. R., Creste, J. E., Matsubara, E. T., Li, J., e Gonçalves, W. N. A convolutional neural network approach for counting and geolocating citrus-trees in UAV multispectral imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 160(December 2019):97–106.

Osco, L. P., dos Santos de Arruda, M., 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., e et al. A cnn approach to simultaneously count plants and detect plantation-rows from uav imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 174:1–17.

Osco, L. P., Marcato Junior, J., Marques Ramos, A. P., de Castro Jorge, L. A., Fatholahi, S. N., de Andrade Silva, J., Matsubara, E. T., Pistori, H., Gonçalves, W. N., e Li, J. A review on deep learning in uav remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, 102:102456.

Osco, L. P., Ramos, A. P. M., Faita Pinheiro, M. M., Moriya, E. A. S., Imai, N. N., Estrabis, N., Ianczyk, F., Araújo, F. F. d., Liesenberg, V., Jorge, L. A. d. C., Li, J., Ma, L., Gonçalves, W. N., Marcato Junior, J., e Eduardo Creste, J. A machine learning framework to predict nutrient content in valencia-orange leaf hyperspectral measurements. *Remote Sensing*, 12(6).

Ouyang, S. e Li, Y. Combining deep semantic segmentation network and graph convolutional neural network for semantic segmentation of remote sensing imagery. *Remote Sensing*, 13(1):1–22.

Pan, S., Guan, H., Chen, Y., Yu, Y., Nunes Gonçalves, W., Marcato Junior, J., e Li, J. Land-cover classification of multispectral lidar data using cnn with optimized hyper-parameters. *ISPRS Journal of Photogrammetry and Remote Sensing*, 166:241–254.

Passos, A., Rai, P., Wainer, J., e Daumé, H. Flexible modeling of latent task structures in multitask learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pagina 1283–1290.

Ramos, A. P. M., Osco, L. P., Furuya, D. E. G., Gonçalves, W. N., Santana, D. C., Teodoro, L. P. R., da Silva Junior, C. A., Capristo-Silva, G. F., Li, J., Baio, F. H. R., Junior, J. M., Teodoro, P. E., e Pistori, H. A random forest ranking approach to predict yield in maize with uav-based vegetation spectral indices. *Computers and Electronics in Agriculture*, 178:105791.

Ranftl, R., Bochkovskiy, A., e Koltun, V. Vision transformers for dense prediction. In *IEEE International Conference on Computer Vision (ICCV)*, páginas 12179–12188.

Ranjan, R., Patel, V. M., e Chellappa, R. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1):121–135.

Ravi, R., Lin, Y., Elbahnasawy, M., Shamseldin, T., e Habib, A. Simultaneous system calibration of a multi-lidar multicamera mobile mapping platform. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(5):1694–1714.

Ribas, L. C., Gonçalves, D. N., Silva, J., Castro, A., Bruno, O. M., e Gonçalves, W. N. Fractal dimension of bag-of-visual words. *Pattern Anal. Appl.*, 22(1):89–98.

Ronneberger, O., Fischer, P., e Brox, T. U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9351:234–241.

Ronneberger, O., Fischer, P., e Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, páginas 234–241. Springer.

Rosa, L. E. C. L., Oliveira, D. A. B., Zortea, M., Gemignani, B. H., e Feitosa, R. Q. Learning geometric features for improving the automatic detection of citrus plantation rows in uav images. *IEEE Geoscience and Remote Sensing Letters*, páginas 1–5.

Ruder, S. An overview of multi-task learning in deep neural networks. *ArXiv*, abs/1706.05098.

Ruder, S., Bingel, J., Augenstein, I., e Søgaard, A. Latent multi-task architecture learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4822–4829.

Salman, A., Siddiqui, S. A., Shafait, F., Mian, A., Shortis, M. R., Khurshid, K., Ulges, A., e Schwanecke, U. Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system. *ICES Journal of Marine Science*, 77(4):1295–1307.

Schenk, T. From point-based to feature-based aerial triangulation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 58(5):315 – 329.

Shelhamer, E., Long, J., e Darrell, T. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651.

Siam, M., Elkerdawy, S., Jagersand, M., e Yogamani, S. Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, páginas 1–8.

Simonyan, K. e Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, páginas 1–14.

Strudel, R., Garcia, R., Laptev, I., e Schmid, C. Segmenter: Transformer for semantic segmentation. In *The IEEE International Conference on Computer Vision (ICCV)*, páginas 7262–7272.

Sun, Y., Robson, S., Scott, D., Boehm, J., e Wang, Q. Automatic sensor orientation using horizontal and vertical line feature constraints. *ISPRS Journal of Photogrammetry and Remote Sensing*, 150:172 – 184.

Sveen, L., Timmerhaus, G., Johansen, L.-H., e Ytteborg, E. Deep neural network analysis - a paradigm shift for histological examination of health and welfare of farmed fish. *Aquaculture*, 532:736024.

Tommaselli, A. M. e Junior, J. M. Bundle block adjustment of cbers-2b hrc imagery combining control points and lines. *Photogrammetrie - Fernerkundung - Geoinformation*, 2012(2):129–139.

Vafaeikia, P., Namdar, K., e Khalvati, F. A brief review of deep multi-task learning and auxiliary task learning. *CoRR*, abs/2007.01126.

Vandenhende, S., Georgoulis, S., Gool, L. V., e Brabandere, B. D. Branched multi-task networks: Deciding what layers to share. In *British Machine Vision Conference*, páginas 1–14.

Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., e Chen, L.-C. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *ECCV*, páginas 108–126.

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

Wang, Y., Luo, X., Ding, L., Fu, S., e Wei, X. Detection based visual tracking with convolutional neural network. *Knowledge-Based Systems*, 175:62 – 71.

Wei, D., Zhang, Y., Liu, X., Li, C., e Li, Z. Robust line segment matching across views via ranking the line-point graph. *ISPRS Journal of Photogrammetry and Remote Sensing*, 171:49 – 62.

Wei, Y., Zhang, K., e Ji, S. Simultaneous road surface and centerline extraction from large-scale remote sensing images using cnn-based segmentation and tracing. *IEEE Transactions on Geoscience and Remote Sensing*, 58(12):8919–8931.

Wei, Z., Jia, K., Jia, X., Khandelwal, A., e Kumar, V. Global river monitoring using semantic fusion networks. *Water*, 12(8).

Weld, G., Jang, E., Li, A., Zeng, A., Heimerl, K., e Froehlich, J. E. Deep learning for automatically detecting sidewalk accessibility problems using streetscape imagery. *ASSETS 2019 - 21st International ACM SIGACCESS Conference on Computers and Accessibility*, páginas 196–209.

Weng, L., Xu, Y., Xia, M., Zhang, Y., Liu, J., e Xu, Y. Water areas segmentation from remote sensing images using a separable residual segnet network. *ISPRS International Journal of Geo-Information*, 9(4).

Xia, M., Qian, J., Zhang, X., Liu, J., e Xu, Y. River segmentation based on separable attention residual network. *Journal of Applied Remote Sensing*, 14(3):1 – 15.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., e Luo, P. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Proceedings of the 35th Annual Conference on Neural Information Processing Systems, NeurIPS 2021*.

Xue, Y., Liao, X., Carin, L., e Krishnapuram, B. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(2):35–63.

Yan, X., Ai, T., Yang, M., e Yin, H. A graph convolutional neural network for classification of building patterns using spatial vector data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 150(February):259–273.

Yang, B. e Chen, C. Automatic registration of uav-borne sequent images and lidar data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 101:262 – 274.

Yang, X., Li, X., Ye, Y., Lau, R. Y. K., Zhang, X., e Huang, X. Road detection and centerline extraction via deep recurrent convolutional neural network u-net. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):7209–7220.

Yavari, S., Valadan Zoej, M. J., e Salehi, B. An automatic optimum number of well-distributed ground control lines selection procedure based on genetic algorithm. *ISPRS Journal of Photogrammetry and Remote Sensing*, 139:46 – 56.

Yu, F. e Koltun, V. Multi-scale context aggregation by dilated convolutions. In *4th International Conference on Learning Representations*, páginas 1–13.

Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.-H., Tay, F. E., Feng, J., e Yan, S. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, páginas 558–567.

Yuan, Y., Chen, X., e Wang, J. Object-contextual representations for semantic segmentation. In Vedaldi, A., Bischof, H., Brox, T., e Frahm, J.-M., editors, *Computer Vision – ECCV 2020*, páginas 173–190. Springer International Publishing.

Zhang, C., Sargent, I., Pan, X., Li, H., Gardiner, A., Hare, J., e Atkinson, P. M. Joint deep learning for land cover and land use classification. *Remote Sensing of Environment*, 221:173–187.

Zhang, L., Li, W., Liu, C., Zhou, X., e Duan, Q. Automatic fish counting method using image density grading and local regression. *Computers and Electronics in Agriculture*, 179:105844.

Zhang, S., Yang, X., Wang, Y., Zhao, Z., Liu, J., Liu, Y., Sun, C., e Zhou, C. Automatic fish population counting by machine vision and a hybrid deep neural network model. *Animals*, 10(2).

Zhang, Y. e Yang, Q. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 1:1–1.

Zhang, Z., Cui, P., e Zhu, W. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge Data Engineering*, 34(01):249–270.

Zhang, Z., Li, Z., Bi, N., Zheng, J., Wang, J., Huang, K., Luo, W., Xu, Y., e Gao, S. Ppgnet: Learning point-pair graph for line segment detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, páginas 7098–7107.

Zhao, H., Zhang, Y., Liu, S., Shi, J., Loy, C. C., Lin, D., e Jia, J. Psanet: Point-wise spatial attention network for scene parsing. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Zhao, J., Bao, W., Zhang, F., Zhu, S., Liu, Y., Lu, H., Shen, M., e Ye, Z. Modified motion influence map and recurrent neural network-based monitoring of the local unusual behaviors for fish school in intensive aquaculture. *Aquaculture*, 493:165–175.

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., e Torr, P. H. S. Conditional random fields as recurrent neural networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, páginas 1529–1537.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., e Zhang, L. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*.

Zhou, C., Xu, D., Chen, L., Zhang, S., Sun, C., Yang, X., e Wang, Y. Evaluation of fish feeding intensity in aquaculture using a convolutional neural network and machine vision. *Aquaculture*, 507:457–465.

Zhou, H., Lu, C., Yang, S., e Yu, Y. Convnets vs. transformers: Whose visual representations are more transferable? In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, páginas 2230–2238.

Zhou, Y., Chen, H., Li, Y., Liu, Q., Xu, X., Wang, S., Yap, P.-T., e Shen, D. Multi-task learning for segmentation and classification of tumors in 3d automated breast ultrasound images. *Medical Image Analysis*, 70:101918.

Zhu, X., Su, W., Lu, L., Li, B., Wang, X., e Dai, J. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, páginas 1–16.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., e He, Q. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.