UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

Dissertação de Mestrado

Segmentação de Ervas Daninhas da Soja usando VT-Net: Um Modelo *Transformer*-Convolucional

Lucas de Souza Silva

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande - FURG, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Orientador: Prof. Dr. Rodrigo Andrade de Bem
Co-orientador: Prof. Dr. Paulo Lilles Jorge Drews Jr.

Rio Grande, 2024

UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
CENTRO DE CIÊNCIAS COMPUTACIONAIS
GRADUATE PROGRAM IN COMPUTER ENGINEERING
MASTER OF SCIENCE IN COMPUTER ENGINEERING

Master of Science Dissertation

Soybean Weeds Segmentation using VT-Net: a Convolutional-Transformer Model

Lucas de Souza Silva

Master of Science Dissertation presented to the Graduate Program in Computer Engineering of the Federal University of Rio Grande, as a partial requirement to obtain a Master of Science degree in Computer Engineering

Supervisor: Prof. Dr. Rodrigo Andrade de Bem
Co-supervisor: Prof. Dr. Paulo Lilles Jorge Drews Jr.

Rio Grande, 2024

Ficha Catalográfica

S586s Silva, Lucas de Souza.
Segmentação de ervas daninhas da soja usando VT-Net: um modelo *Transformer-Convolucional* / Lucas de Souza Silva. – 2024.
63 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2024.
Orientador: Dr. Rodrigo Andrade de Bem.
Coorientador: Dr. Paulo Lilles Jorge Drews Jr.

1. Segmentação 2. Ervas daninhas 3. Soja 4. Agricultura de precisão 5. *Vision Transformer* 6. *Deep Learning* 7. Visão computacional I. Bem, Rodrigo Andrade de II. Drews Jr., Paulo Lilles Jorge III. Título.

CDU 004

Catalogação na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344

Dissertação de Mestrado

Segmentação de Ervas Daninhas da Soja usando VT-Net: um Modelo *Transformer*-Convolucional

Lucas de Souza Silva

Banca examinadora:

Prof. Dr. Rafael Alceste Berri

Prof. Dr. Thales Sehn Körting

Prof. Dr. Paulo Lilles Jorge Drews Jr.

Prof. Dr. Rodrigo Andrade de Bem
Orientador

I dedicate this project to my family, especially my father, who always showed me my potential, and I thank God for the opportunity to have a bright future.

ACKNOWLEDGEMENTS

I would like to thank especially my supervising professor, Rodrigo Andrade de Bem, for having proposed this great challenge of developing an in-depth work such as a master's thesis, on an unprecedented topic, which has not yet been explored for applications in the agro-industry. I am also grateful to my co-advisor Paulo Drews Jr., for having accompanied, instructed, and encouraged me to carry out this work that, in addition to being a personal achievement, contributes to the scientific community.

I also thank my family, who always supported and encouraged me to be the best. Lastly, I would like to thank all my college colleagues and the projects I collaborated on.

“Do not conform to the structures of this world,
but be transformed by the renewal of the mind,
in order to distinguish what God’s will is:
what is good, what is pleasing to Him, what is perfect.
— (Holy Bible, Romans 12, 2)

ABSTRACT

SILVA, Lucas de Souza. **Soybean Weeds Segmentation using VT-net: a Convolutional-Transformer Model**. 2024. 63 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

Given the growth of neural network studies and their applications, the question arose: How could Vision Transformer Networks be used in the field of agronomy? To answer this question, this work aims to develop a neural network for semantic segmentation of weeds in soybean cultivation, using the Vision Transformer (ViT) model, a neural network that uses a mechanism of self-attention, to identify and carry out the weed segmentation. To address the problem, the dataset Deepweeds was used. The ViT model was compared with the networks: Segmenter, CvT, Resnet 50 v2, Deeplab v3+, Mobilenet, and Swin-Transformer. The model developed is composed of a ViT-Base backbone, with 12 layers and 86 million parameters. This network has components from a Resnet50 architecture, used for feature extraction, forming the final segmentation model with 16 layers and 120 million parameters. The segmentation results, presenting an accuracy of 93.89% pixel-by-pixel and Mean Intersection over Union (mIoU) of 0.626, were close to the BEiT model, a state-of-the-art network for the problem.

Palavras-chave: Segmentation, Weeds, Soybean, Precision Agriculture, Vision Transformer, Deep Learning, Computer Vision.

RESUMO

SILVA, Lucas de Souza. **Segmentação de Ervas Daninhas da Soja usando VT-Net: um Modelo *Transformer*-Convolucional**. 2024. 63 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

Diante do crescimento dos estudos de redes neurais e suas aplicações, surgiu a questão: Como as Redes Vision Transformer podem ser utilizadas na área da agronomia? Para responder a essa questão, este trabalho tem como objetivo desenvolver uma rede neural para segmentação semântica de plantas daninhas na cultura da soja, utilizando o modelo Vision Transformer (ViT), uma rede neural que utiliza um mecanismo de autoatenção, para identificar e realizar a segmentação de plantas daninhas. Para abordar o problema, foi utilizado o conjunto de dados Deepweeds. O modelo ViT foi comparado com as redes: Segmenter, CvT, Resnet 50 v2, Deeplab v3+, Mobilenet e Swin-Transformer. O modelo desenvolvido é composto por um backbone ViT-Base, com 12 camadas e 86 milhões de parâmetros. Essa rede possui componentes de uma arquitetura Resnet50, utilizados para extração de características, formando o modelo final de segmentação com 16 camadas e 120 milhões de parâmetros. Os resultados da segmentação, apresentando uma precisão de 93,89% pixel por pixel e *Mean Intersection over Union* (mIoU) de 0,626, foram próximos ao modelo BEiT, rede que consiste no estado-da-arte para o problema.

Palavras-chave: Segmentação, Ervas Daninhas, Soja, Agricultura de Precisão, Vision Transformer, Deep Learning, Visão Computacional.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ViT	Vision Transformer
VT-Net	*Vision Transformer Resnet* - Name of proposed model
AI	Artificial Inteligence
CNN	Artificial Convolutional Neural Networks
DNN	Deep Neural Networks
NLP	Natural Language Processing
MLP	Multilayer Perceptron
mIoU	Mean Intersection Over Union
RNN	Recurrent Neural Network
CFTV	Circuito Fechado de Televisão
SVM	Support Vector Machine
PE	Positional Embedding - Position Identifier
ASPP	Atrous Spatial Pyramid Pooling

SUMMARY

1 INTRODUCTION

One of the most recurrent problems in agriculture is the occurrence of weeds (TERRA et al., 2020). These plants grow parasitizing the host or the ground, such as creeping weeds (TERRA; ROSA; DREWS, 2019). Weed control is generally relatively simple and quick and is obtained through herbicides or pest control. However, they are momentary solutions and can cause several negative side effects, such as changes in the quality of the final product, damage to the plant structure, the risk of killing the host, and soil contamination (NICOLOPOULOU-STAMATI et al., 2016; BENEDETTI et al., 2018). It is well known that the presence of weeds implies a loss of productivity and decreases product quality, as weeds are parasite organisms that lodge in crops and feed on soil nutrients (FLECK; CANDEMIL, 1995). This process limits the uptake of nutrients from the crop under study.

Weeds are often not a single species, they are normally presented through several different species, with different types of interaction with the crop, some being more frequent in certain crops than others. Some examples of soybean weeds are illustrated in Figure 1.

Often, weed control is done by applying heavy herbicides, which are chemical reagents designed to attack certain species (CORREIA; DURIGAN, 2010). Several formulations are necessary to complete weed control, resulting in a high cost of material, infrastructure for storage and application, and manpower for handling. Therefore, the successful development of intelligent systems for precision agriculture is likely to reduce these losses and improve productivity (TERRA et al., 2021). Such systems often rely on image-based techniques to monitor and extract relevant information from the environment (WEBER et al., 2018; NASCIMENTO et al., 2019). These techniques are frequently the primary stage of autonomous weed control systems that promise a step-change in agricultural productivity, Potentially reducing labor costs and herbicide use with applications precisely located at weed targets.

Computer vision has the potential to revolutionize agriculture by providing innovative solutions to common challenges faced by farmers. Through the use of cameras, drones, and advanced image processing algorithms, computer vision can monitor crop health, detect diseases, identify pests, and assess soil conditions in real time (PRIYA; G;

Figure 1: Example of weeds found in soybean cultivation. The image presents only weeds and dead foliage, no soybean is present.

SOURCE: (OLSEN et al., 2019, p.5).

RAJAMANI, 2020). This technology enables precision agriculture, where farmers can target interventions more accurately, applying water, fertilizers, or pesticides only where needed, reducing waste and environmental impact (SHARMA, 2023). Additionally, computer vision can assist in automating labor-intensive tasks such as harvesting, sorting, and grading, increasing efficiency and reducing costs (NITIN; GUPTA, 2024). By harnessing the power of computer vision, farmers can improve yield prediction, optimize resource use, and make data-driven decisions that enhance sustainability and profitability in agriculture.

1.1 Objectives

The general goal of this work is to explore the use of Vision Transformer Networks (DOSOVITSKIY et al., 2021) in the field of agronomy and precision agriculture. More specifically, we tackle the challenging task of soybean weeds segmentation in unconstrained images, aiming to allow a more precise application of herbicides and potentially increase the productivity and biological quality of the crops.

The specific objectives are the extension of a dataset of soybean weeds in a real application environment, with segmentation masks annotation, and comparison with state-of-the-art Vision Transformer methods focusing on the same application.

1.2 Organization of This Work

This work is organized as follows:

- Chapter 1: Introduction and work motivation;
- Chapter 2: Theoretical foundation of weeds in soybean cultivation, computer vision, Transformers networks, and related work;
- Chapter 3: Description of the methodology used to solve the segmentation tasks, covering the dataset and its annotation, creating the ViT model, and adaptation for segmentation.
- Chapter 4: Experiments carried out and results obtained from the segmentation model, with comparison to state-of-the-art Vision Transformer networks;
- Chapter 5: Conclusion and future work.

1.3 Contributions

To the best of our knowledge, the VT-Net is the first hybrid vision transformer method with a pure ViT backbone employed to tackle soybean weed segmentation. Our approach to the task achieved results on par with the state-of-the-art models on the proposed dataset, yet using fewer layers than the baseline competitors. Additionally, we can highlight the following:

- The annotation with segmentation masks of a dataset of soybean weeds in a real application context, with high quality;
- The development and comparison of a hybrid Transformer model built with fewer parameters than the state-of-the-art Transformer networks;
- The application and benchmark of Transformers networks in a specific and challenging task such as weed segmentation.

Part of the contributions mentioned above were published in the following paper:

- L. Silva, P. Drews, and R. de Bem, “Soybean Weeds Segmentation Using VT-Net: A Convolutional-Transformer Model,” 2023 36th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), 2023, pp. 127-132.

2 FOUNDATIONS AND RELATED WORK

This chapter covers the main foundational knowledge necessary to understand the research problem. Additionally, it examines the relevant literature, highlighting the closely related contributions to the field, and demonstrating how the different methods differ in their nature.

2.1 Theoretical Foundation

2.1.1 Weeds in Soybean Cultivation

Weeds are extremely recurrent pests that are difficult to control (FLECK; CANDEMIL, 1995). They are pests that spread quickly and are often not able to be contained efficiently. They lodge around, and sometimes in, the host's body, thus causing the need to apply herbicide extremely close to the host plant.

The negative effects of weeds on crops include competition for water, light, nutrients, and space, increased production costs, difficulty in harvesting, depreciation of product quality, hosting pests and diseases, and decreased commercial value of cultivated areas. To control weed competition, four types of management are used: exclusion, prevention, suppression, and eradication (CORREIA; DURIGAN, 2010).

When an area has a high infestation of weeds, there can be a 78% reduction in the number of soybean pods per plant (SILVA et al., 2008). This value can reach a loss of productivity of 80% (VARGAS; ROMAN, 2006), making it impossible to harvest in places that have not implemented any control method. In areas where control is carried out late, losses can be 12 to 15% (BIANCHI et al., 2011) of production depending on the species of weed and the cultivar sown.

2.1.2 Deep Learning and Computer Vision

Deep learning is a topic of machine learning in which algorithms process data, and learn its representation, inspired by the processing done by the human brain (GOODFELLOW; BENGIO; COURVILLE, 2016). Deep Learning uses layers of mathematical calculations to process data, understand human speech (PANDA, 2017), and recognize

objects visually (SHIT; DAS; RAY, 2023). Information is passed through each layer, and the output of the previous layer provides input to the next layer. The first layer in a network is called the input layer, while the last is called the output layer. All layers in between are referred to as hidden layers, and each layer is typically a simple, uniform algorithm containing one type of activation function. Feature extraction is another aspect of Deep Learning. This process uses an algorithm to automatically construct meaningful "aspects" of data for training, learning, and understanding purposes (LECUN; BENGIO; HINTON, 2015).

Regarding computer vision, it may be defined as the task of extracting useful information from images fed to a model (PRINCE, 2012). This task proves to be quite challenging, and the main purpose is to make useful and intelligent decisions often in physical environments through data. To make such decisions it is necessary to build a model capable of reading these images, processing and evaluating them. In computer vision, there are several conditions to be presented to the model: color, luminosity, and color distribution, among others, which contribute to the greater difficulty of the problem (SZELISKI, 2010). Nowadays, computer vision is widely used in photo and video applications (PAVLOV et al., 2013), CCTV systems (monitoring cameras) (RAKSHITHA; SELVAN, 2023), embedded systems in smart cars (IBRAHIM et al., 2020), and many other environments. Image recovery can also be highlighted, which has recently gained prominence in these applications (JAIN; KASTURI; SCHUNCK, 1995).

Lately, Deep Learning excels in tasks such as image recognition, natural language processing, and speech synthesis. It requires large labeled datasets for training. Neural networks learn features through backpropagation and are computationally expensive due to the large number of parameters and layers.

With those definitions established, it is possible to contrast with Classical Machine Learning, which encompasses a broader range of algorithms. It includes decision trees, Naive Bayes, support vector machines (SVMs), and more. Unlike deep neural networks, classical machine learning models don't necessarily involve deep architectures. It can work with smaller datasets, relies on explicit programming and feature engineering (handicraft algorithms), and is generally less computationally intensive. They are widely used in various applications as recommendation systems (e.g., personalized movie recommendations) (PU; HU, 2023), fraud detection (e.g., credit card fraud) (LOKANAN; LIU, 2021), and regression tasks (e.g., predicting house prices) (CHEN; HUANG, 2023).

2.1.3 Transformers Networks

It is important to bring basic knowledge of the functioning of Transformers networks, which derive from applications in Natural Language Processing (VASWANI et al., 2017), such as language modeling, translation, interpretation, and others (SUTSKEVER; VINYALS; LE, 2014; BAHDANAU; CHO; BENGIO, 2015; CHO et al., 2014).

Transformers networks are similar to Recurrent Neural Networks (RNN), differing only in the form of processing, where in Transformer models the data is processed in parallel via the self-attention mechanism, and in RNNs they are processed sequentially.

By default, Transformer networks have the encoder-decoder structure. It has a sequential computation of information (information vectorization), maintaining a memory through a patching system. That system inserts a position token for each element in a sequence. Due to its nature of maintaining memory, the network must present a balance in the size of patches and batches. Patches are the amount of information that each element carries, and batches how much each processing will progress. The performance improvement of Transformers is done through weight adjustments and the optimization of patches and batch sizes of the model architecture.

Attention-based systems (BELLO et al., 2020; RAMACHANDRAN et al., 2019; ZHAO; JIA; KOLTUN, 2020) have become a basis for studying Transformers networks. From them, we can describe the architecture of a Transformer network, shown in Figure 2.

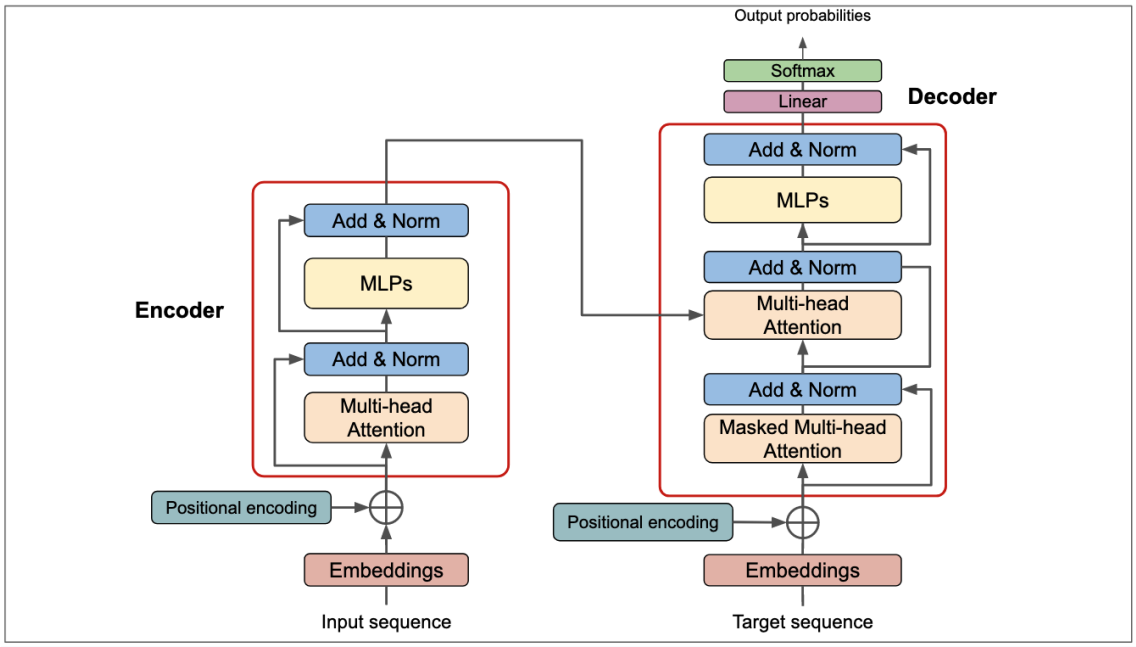

Figure 2: Transformer Architecture.

SOURCE: (VASWANI et al., 2017, p.3)

Moving on to the functioning of the encoder and decoder, both are composed of 6 layers. The Encoder is composed of two internal levels, the first being a multi-head mechanism, and the second level of concatenation and positioning. The decoder also presents the same two levels, however, it then presents the multi-attention self-attention heads to output the network.

At the core of the model, the attention of multiple Multilayer Perceptron (MLP) heads receives this linearized information in vectors, with their respective positions. We feed

them parallel to the heads, and each one will perform the self-attention process, and then the network output becomes the average of the scores of these heads.

2.1.4 Vision Transformers

The standard Transformer has as input a one-dimensional (1D) sequence of token embeddings. To handle 2D images, the Vision Transformer model (DOSOVITSKIY et al., 2021), illustrated in Figure 3, reshapes the image $x_p \in \mathbb{R}^{H \times W \times C}$ into a sequence of flattened 2D patches $\mathrm{X}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where the pair (H, W) is the dimension of the original image, C is the number of channels, (P, P) is the resolution of each image patch, and $N = HW/P^2$ is the resulting number of patches, which also describes the length of the input sequence for the Transformer. The model uses a constant vector with size *D* through all of its layers, the patches are flattened and mapped to *D* dimensions with a trainable linear projection, then the output of this projection is referred to as the patch embeddings.

The model prepends a learnable embedding to the sequence of embedded patches ($\mathrm{z}_0^0 = \mathrm{x}_{class}$), whose state at the output of the Transformer encoder (z_L^0) serves as the image representation. During pre-training and fine-tuning a classification head is attached to z_L^0. The classification head consists of an MLP with one hidden layer at the pre-training step and a single linear layer at the fine-tuning step.

The Transformer encoder consists of alternating layers of multiheaded self-attention (MSA) and MLP blocks. The layer normalization action is applied before every block and residual connections after every block.

Standard *qkv* self-attention is used and for each element in an input sequence $z \in \mathbb{R}^{N \times D}$, the model computes a weighted sum over all values *v* in the sequence. The attention weights A_{ij} are based on the similarity between two elements of the sequence and their respective query q^i and key k^j representations,

$$[q, k, v] = zU_{qkv}, U_{qkv} \in \mathbb{R}^{D \times 3D_h}, \tag{1}$$

$$A = softmax(qk^T/\sqrt{D_h}), A \in \mathbb{R}^{N \times N}, \tag{2}$$

$$SA(z) = Av. \tag{3}$$

The multi-head self-attention (MSA) is an extension of self-attention (SA) in which the model runs *k* self-attention operations, called “heads”, in parallel, and projects their concatenated outputs. MSA is defined in the equations as per

$$\begin{gathered} MSA(z) = [SA_1(z); SA_2(z); ...; SA_k(z)]U_{msa}, \\ U_{msa} \in \mathbb{R}^{k \cdot D_h \times D}. \end{gathered} \tag{4}$$

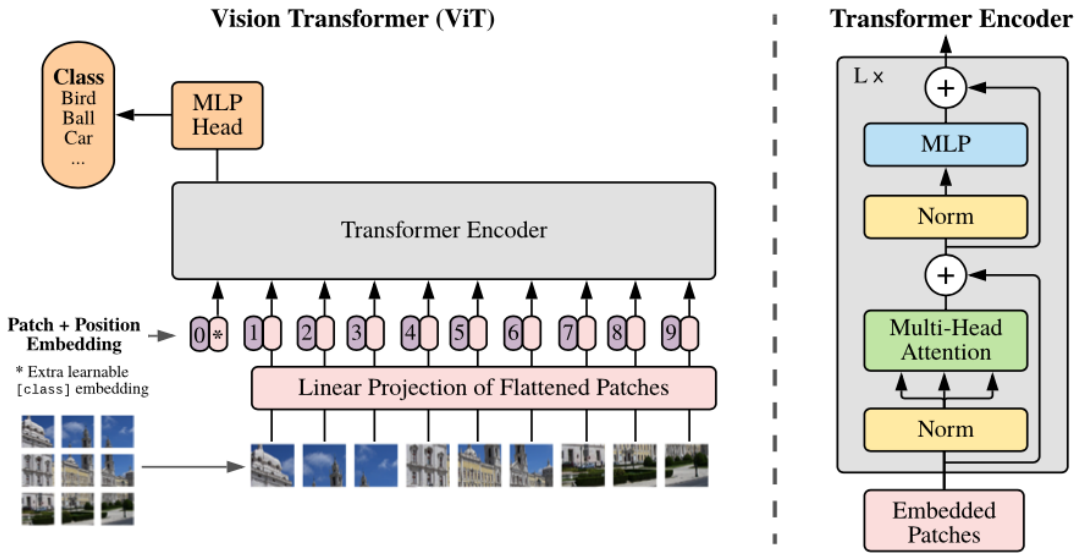

Figure 3: Vision Transformer architecture.

SOURCE: (DOSOVITSKIY et al., 2021, p.3)

The original ViT main model, the ViT-Base, has 12 *transformer* layers, dimension 768, head size (*MLP size*) of 3072, 12 heads, and 86 million parameters. Additionally, there are other variants, named the ViT-Large and the ViT-Huge, which are detailed in Table 1.

Table 1: ViT Sizes.

Model	*Layers*	Dimension	MLP Size	Heads	Parameters
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

SOURCE: (DOSOVITSKIY et al., 2021, p.5).

Some authors (STRUDEL et al., 2021; LIU et al., 2022; WU et al., 2022) also define a smaller model called ViT-Small, or sometimes ViT-Tiny, which presents 12 layers, as the base model, but differs regarding the dimension of 256, MLP size of 1024, 8 heads, and 5M parameters.

2.2 Related works

2.2.1 Segmentation Using Vision Transformers

There are already some works on adaptations of a Transformer model for image analysis (ZHAO; JIA; KOLTUN, 2020; BELLO et al., 2020; RAMACHANDRAN et

al., 2019), and is shown the performance of this structure *encoder-decoder* with *tokens* generating competitive results with other state-of-the-art networks. However, it became evident in these studies that models based on Transformers require a lot of data, always requiring datasets with a large number of images, normally more than a million samples.

Applications that surpassed the state-of-the-art were chosen as the basis for related work. The following models can be mentioned: Segmenter (STRUDEL et al., 2021), Convolutional Vision Transformer (CvT) (WU et al., 2021), Swin-Transformer (LIU et al., 2022), SegFormer (XIE et al., 2021), BEiT (BAO et al., 2021), and Mask2Former (CHENG et al., 2022), which will be analyzed individually in the following sub-sections.

2.2.1.1 Segmenter

The Segmenter (STRUDEL et al., 2021) model works with a purely Transfomer, multi-class network, which performs segmentation through the interpretation of the different attention maps generated by the Transformer decoder. The network presents seven variants of its main model, expressed through Table 2, and its architecture through Figure 4.

Table 2: Composition of SegFormer models

Method	*Backbone*	Layers	Parameters
Mon-Ti	ViT-Ti	12	6M
Mon-S	ViT-S	12	22M
Mon-B	ViT-B	12	86M
Mon-B	DeiT-B	12	86M
Mon-L	ViT-L	24	307M

SOURCE: (STRUDEL et al., 2021, p.4).

In short, it is possible to say that the work presented is a pure application of a ViT network, where differentiation occurs by introducing the separation of classes through the Mask Transformer and its handling through the scalar product, and *upsample* to perform multi-class segmentation.

In addition to the main feature (pure Transformer model), the work presents metrics that in certain cases surpass the state-of-the-art for two popular benchmarks: Pascal (EVERINGHAM et al., 2015) and Cityscapes (CORDTS et al., 2016), made available in Figure 5. Through Figure 6 it is possible to interpret the functioning of the model's multi-layers by interpreting the layers' attention maps.

2.2.1.2 Convolutional Vision Transformer (CvT)

Addressing hybrid networks, as this work proposes, we find the Convolutional Vision Transformer (CvT) (WU et al., 2021) proposal, which hybridizes the ViT model with

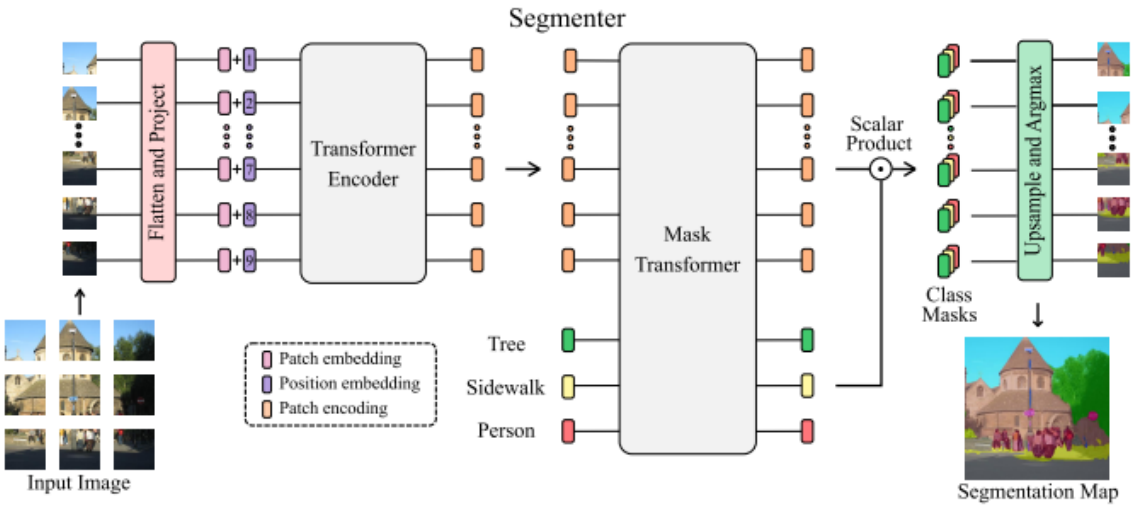

Figure 4: Approach description *Segmenter*. (Left) Encoder: Image *patches* are projected into a vector and then encoded with a transformer. (Right) Decoder: A mask transformer (*mask transformer*) takes as input the encoder output and class vectors to predict segmentation masks.

SOURCE: (STRUDEL et al., 2021, p.4).

the Convolutional Networks (CNNs) method. The change proposed by the authors is to adopt a network that uses the concept of *tokens* (as opposed to *patches* of Transformers) the results of each *Stage* (Stages), as called by the author, the MLP heads are submitted, unlike the standard ViT operation, which passes the data to the MLP heads only at the end of the network processing. CvT details characteristics are summarized in Table 3, while the architecture is shown in Figure 7.

Table 3: Composition of CvT models

Method	Layers	Parameters
CvT-13	13	20M
CvT-21	21	32M
CvT-W24	24	277M

The authors do not consider backbone applicable to the model developed, given that there are major changes to the network architecture, only Transformers "blocks"were used.

SOURCE: (WU et al., 2021, p.6)

The CvT project features convolutions for two main sections of the ViT architecture. First, the Transformers network is partitioned into several stages (*Stages*) that form a hierarchical structure. The start of each stage consists of an embedding of a convolutional *token* that performs an overlapping convolution operation on a 2D reshaped *token*

Method	Backbone	mIoU (MS)
DeepLabv3+ [10]	ResNet-101	48.5
DANet [23]	ResNet-101	52.6
ANN [69]	ResNet101	52.8
CPNet [58]	ResNet-101	53.9
CFNet [64]	ResNet-101	54.0
ACNet [24]	ResNet-101	54.1
APCNet [26]	ResNet101	54.7
DNL [57]	HRNetV2-W48	55.3
DRANet [22]	ResNet-101	55.4
OCR [60]	HRNetV2-W48	56.2
SETR-L MLA [67]	ViT-L/16	55.8
Seg-B†/16	DeiT-B/16	53.9
Seg-B†-Mask/16	DeiT-B/16	55.0
Seg-L/16	ViT-L/16	56.5
Seg-L-Mask/16	ViT-L/16	**59.0**

(a)

Method	Backbone	mIoU (MS)
PSANet [66]	ResNet-101	79.1
DeepLabv3+ [10]	Xception-71	79.6
ANN [69]	ResNet-101	79.9
MDEQ [5]	MDEQ	80.3
DeepLabv3+ [10]	ResNeSt-101	80.4
DNL [57]	ResNet-101	80.5
CCNet [31]	ResNet-101	81.3
Panoptic-Deeplab [12]	Xception-71	81.5
DeepLabv3+ [10]	ResNeSt-200	**82.7**
SETR-L PUP [67]	ViT-L/16	**82.2**
Seg-B†/16	DeiT-B/16	80.5
Seg-B†-Mask/16	DeiT-B/16	80.6
Seg-L/16	ViT-L/16	80.7
Seg-L-Mask/16	ViT-L/16	81.3

(b)

Figure 5: (a) Comparison in task *Pascal Context (Table 7)*. (b) Comparison in task *Cityscapes (Table 8)* - Tables extracted from said work.

SOURCE: (STRUDEL et al., 2021, p.8).

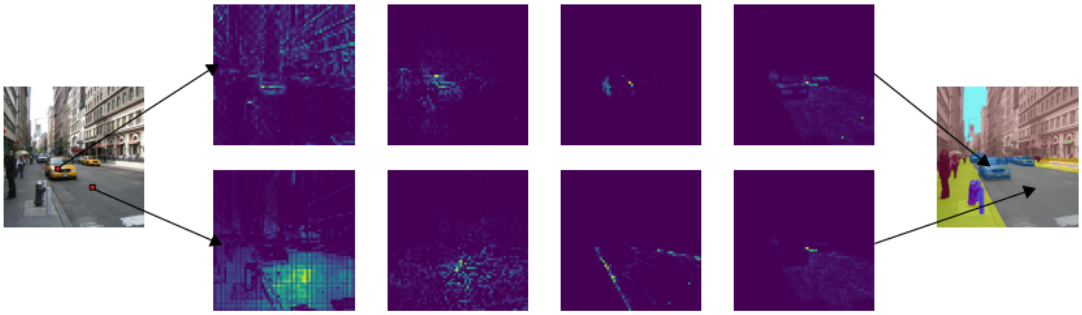

Figure 6: Presentation of attention maps per layer, generating the final multi-class segmentation.

SOURCE: (STRUDEL et al., 2021, p.13).

(i.e., reshaping flattened *token* sequences back to the spatial grid, something equivalent to the vectorization of ViTs), followed by layer normalization. This allows the model to not only capture local information but also to progressively decrease the length of the sequence while increasing the dimension of *tokens* resources between stages, achieving spatial downsampling while simultaneously increasing the number of feature maps, as is done in CNNs.

Second, the linear projection before each self-attention block in the Transformer module is replaced by a convolutional projection proposed by the author, which employs a depth-separable convolution operation s × s on a 2D reshaped *token*. This allows the model to further capture local spatial context and reduce semantic ambiguity in the attention mechanism. It also allows the management of computational complexity, as the processing step convolution can be used to subsample the key and value matrices to improve efficiency by 4× or more (according to the author), with minimal performance degradation.

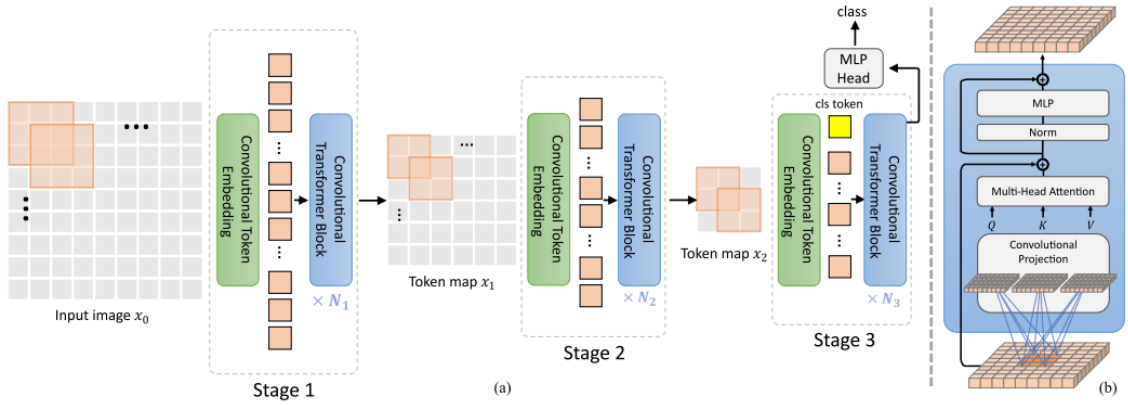

Figure 7: The pipeline proposed by the CvT architecture. (a) General architecture, demonstrating the adoption of layers with *Convolutional Token Embedding*. (b) Description of the Convolutional Transformer Block, which contains the convolution projection as the first layer.

SOURCE: (WU et al., 2021, p.3).

Given the authors' adoptions, CvT presents some peculiarities that greatly distance it from the original Transformers networks, we can cite these peculiarities as forming the identity of CvT, in other words, what makes CvT's unique functioning: Does not require *Positional Embedding* (position identifier). The tokens are overlapping, adding a greater field of perspective to the model. The Projection to the self-attention layer of a convolutional nature, instead of linear (as in the original ViT proposal), and improvement of the global view of the model.

2.2.1.3 Swin Transfomer

Another work that presents a pure network application Transformer is the Swin Transformer (LIU et al., 2022) which performs segmentation through a pure model, which has a mechanism named *Shifted Windows*.

The operation is extremely similar to the "original" model, the change proposed by the authors, which generates improvement in the model, is the use of *Sifted Windows*, which allows, according to the authors, in addition to the global view of the operation of Transformers, a local view, called *local window* by the authors. The windows are predefined through the algorithm and can be of fixed or variable size, their representation is visually presented by Figure 8, and their architecture through Figure 9. The network presents three variants of its main model, described in Table 4.

Interpreting the table, it is possible to conclude that it is a network that has a high number of parameters, and its models above Swin-T differ by the dimensions of the MLP heads, expressed through Figure 10:

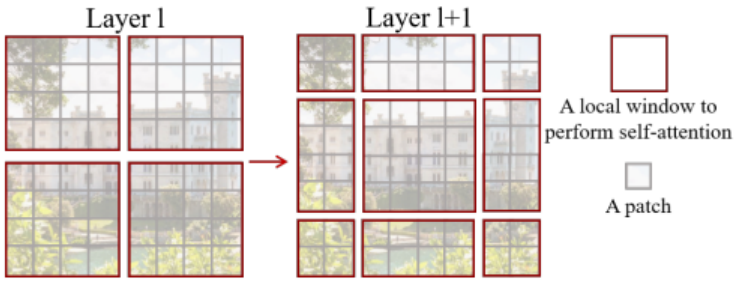

Figure 8: An illustration of *Shifted Windows*'s approach to calculating self-attention in the *Swin Transformer* architecture. In layer l (left), a regular window partitioning scheme is adopted and self-attention is computed within each window. In the next l+1 layer (on the right), the window partitioning is shifted, resulting in new windows. Self-attention computation in the new windows crosses the boundaries of the previous windows in layer l, providing connections between them.

SOURCE: (LIU et al., 2022, p.2).

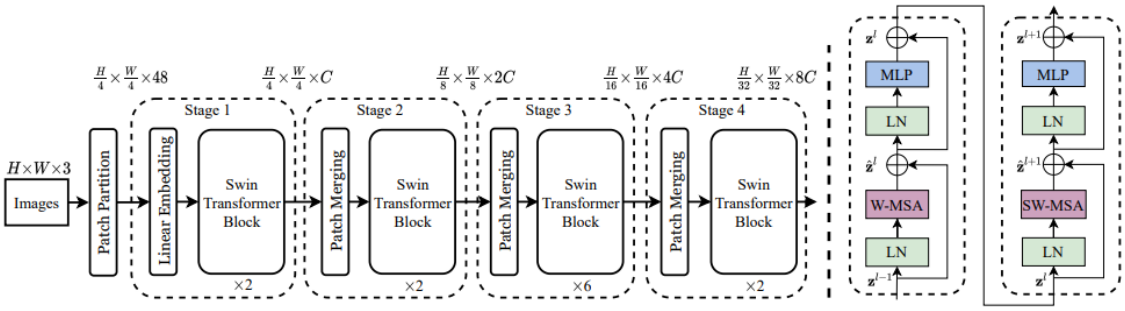

Figure 9: *Swin Transformer* Architecture .

SOURCE: (LIU et al., 2022, p.4).

Table 4: Composition of Swin Transformer models

Method	*Backbone*	Layers	Parameters
Swin-T	ViT-Ti	16	29M
Swin-S	ViT-S	28	50M
Swin-B	ViT-B	28	88M
Swin-L	ViT-L	28	197M

SOURCE: (LIU et al., 2022, p.6,p.10).

2.2.1.4 SegFormer

The two main proposals of the SegFormer (XIE et al., 2021) model are to develop a new encoder Transformer free of positional encoding (*Positional Embedding*) and hierarchical, and a lightweight *decoder design "All-MLP"* that produces a powerful representation without complex, computationally demanding modules.

	downsp. rate (output size)	Swin-T	Swin-S	Swin-B	Swin-L
stage 1	4× (56×56)	concat 4×4, 96-d, LN	concat 4×4, 96-d, LN	concat 4×4, 128-d, LN	concat 4×4, 192-d, LN
		[win. sz. 7×7, dim 96, head 3] × 2	[win. sz. 7×7, dim 96, head 3] × 2	[win. sz. 7×7, dim 128, head 4] × 2	[win. sz. 7×7, dim 192, head 6] × 2
stage 2	8× (28×28)	concat 2×2, 192-d , LN	concat 2×2, 192-d , LN	concat 2×2, 256-d , LN	concat 2×2, 384-d , LN
		[win. sz. 7×7, dim 192, head 6] × 2	[win. sz. 7×7, dim 192, head 6] × 2	[win. sz. 7×7, dim 256, head 8] × 2	[win. sz. 7×7, dim 384, head 12] × 2
stage 3	16× (14×14)	concat 2×2, 384-d , LN	concat 2×2, 384-d , LN	concat 2×2, 512-d , LN	concat 2×2, 768-d , LN
		[win. sz. 7×7, dim 384, head 12] × 6	[win. sz. 7×7, dim 384, head 12] × 18	[win. sz. 7×7, dim 512, head 16] × 18	[win. sz. 7×7, dim 768, head 24] × 18
stage 4	32× (7×7)	concat 2×2, 768-d , LN	concat 2×2, 768-d , LN	concat 2×2, 1024-d , LN	concat 2×2, 1536-d , LN
		[win. sz. 7×7, dim 768, head 24] × 2	[win. sz. 7×7, dim 768, head 24] × 2	[win. sz. 7×7, dim 1024, head 32] × 2	[win. sz. 7×7, dim 1536, head 48] × 2

Figure 10: Reproduction of *Table 7*, with specific details of each architectural model.

SOURCE: (LIU et al., 2022, p.10).

First, the proposed encoder avoids the need for *Positional Embedding* when performing inferences on images with resolutions different from the training one. As a result, the encoder can easily adapt to arbitrary test resolutions without affecting performance. Additionally, the hierarchical part allows the encoder to generate high-resolution "fine" features and low-resolution "coarse" features, in contrast to ViT, which can only produce single feature maps of low resolution with fixed resolutions. Another consequence of choosing the encoder is generating an overlap of *patches* (*overlapped matches merging*).

Secondly, there is a proposal for a "lightweight" MLP decoder, where the key idea is to take advantage of the resources induced by the Transformer architecture, where the attentions of the lower layers tend to remain local, while those of the upper layers become global. By aggregating information from different layers, the MLP decoder combines local and global attention. As a result, we obtain a straightforward decoder that renders local and global information together.

The architecture of the model is illustrated through Figure 11, it is noted that there is the presence of several convolutions and blocks of *upsamble*, thus allowing the classification of the SegFormer network as a hybrid network, which features a *backbone* Transformer and convolutional.

The decoder, called by the authors "All-MLP" consists of four main steps. First, the multi-level resources of the "MiT" encoder pass through an MLP layer to unify the channel dimension. Then, in the second step, the features are sampled to 1/4 and concatenated. Third, an MLP layer is adopted to merge the concatenated *features*. Finally, another MLP layer uses the fused *features* to predict the segmentation mask.

Finally, the authors do not provide how many layers the model presents, only the number of parameters for each model, expressed through Table 5.

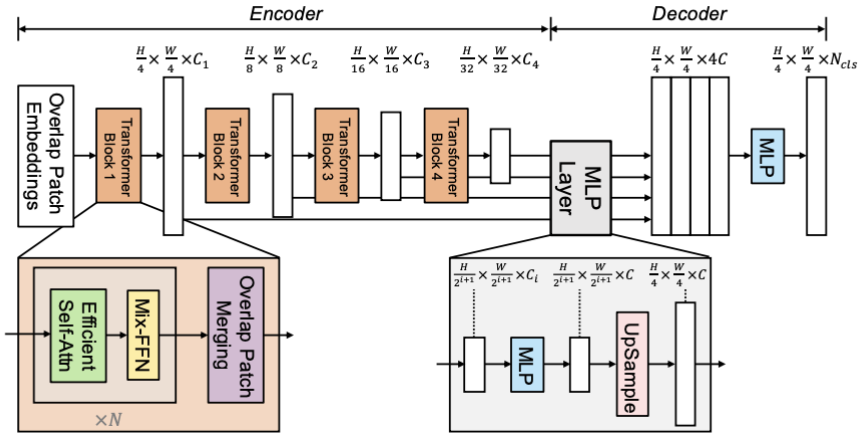

Figure 11: The *SegFormer* framework consists of two main modules: A hierarchical Transformer encoder to extract *features*, and a *"All-MLP"* decoder to directly aggregate these multi-level features and generate the semantic segmentation mask. "FFN" indicates a *feed-forward* block.

SOURCE: (XIE et al., 2021, p.3).

Table 5: Number of parameters according to the SegFormer model variant

Model	Encoder (M)	Decoder (M)	Total (M)
MiT-B0	3.4	0.4	3.8
MiT-B1	13.1	0.6	13.7
MiT-B2	24.2	3.3	27.5
MiT-B3	44	3.3	47.3
MiT-B4	60.8	3.3	64.1
MiT-B5	81.4	3.3	84.7

SOURCE: (XIE et al., 2021, p.7).

2.2.1.5 BEiT

The work BEiT (BAO et al., 2021) presents a self-supervised Vision Transformers model, which stands for Bidirectional Encoder representation from Image Transformers.

Inspired by BERT, the author proposes a pre-training task (*pre-train*), called *Masked Image Models* (MIM). MIM uses two views for each image, namely image *patches* and visual *tokens*. The image is divided into a grid of *patches* that are the input representation of the *backbone* of the Transformer. Furthermore, it "tokenizes" the image into discrete visual *tokens*.

During pre-training, it randomly applies masks to some *patches* of the image and feeds the "corrupted" input to the *Encoder*. The model learns to recover the visual *tokens* of the original image, rather than the raw *pixels* of the masked *patches*. These processes are illustrated through Figure 12.

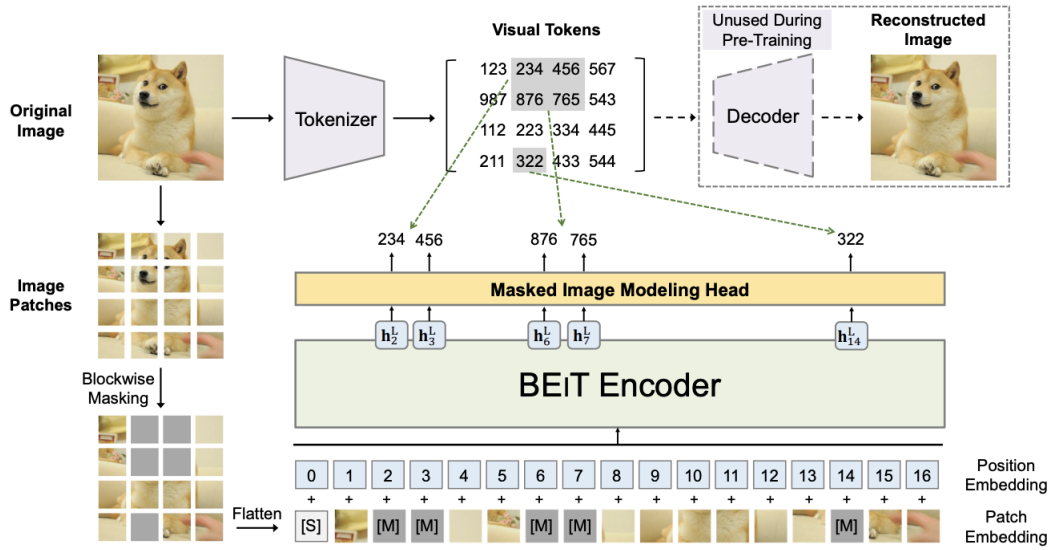

Figure 12: Before pre-training, the "image tokenizer" is applied by reconstructing the image into discrete *tokens*. During pre-training, each image has two views, i.e. image *patches* and visual *tokens*. Masks are randomly applied to some *patches* of the image (gray patches in the figure) [M]. Then the *patches* are fed to the *Encoder*. The pre-training task aims to predict the visual *tokens* of the original image based on the encoding vectors of the corrupted image.

SOURCE: (BAO et al., 2021, p.2).

Self-supervised learning and fine-tuning are done on the pre-trained BEIT on two downstream tasks, namely image classification and semantic segmentation. Experimental results indicate that BEIT outperforms both training from scratch and previous strong self-supervised models. Furthermore, BEIT is complementary to supervised pre-training. The performance of BEIT can be further improved by intermediate fine-tuning with ImageNet labels. Ablation studies show that the proposed techniques are critical to the effectiveness of BERT-style pre-training for image data. In addition to performance, improvements in convergence speed and fine-tuning stability reduce training costs on final tasks. Furthermore, it's demonstrated that self-supervised BEIT can learn reasonable semantic regions via pre-training, releasing the rich supervision signals contained in images.

2.2.1.6 Mask2Former

The work *Mask2Former* (CHENG et al., 2022) mainly uses an operation using mask classification architectures that group pixels into N segments predicting N binary masks, together with N corresponding category labels. Mask classification is general enough to address any segmentation task by assigning different semantics, e.g., categories or instances, to different segments.

Inspired by DETR (CARION et al., 2020), each segment in an image can be represen-

ted as a C-dimensional feature vector ("*object query*") and can be processed by a Transformer decoder, trained with a defined targeting objective. The model architecture of this work consists of three components: a *backbone* that extracts low-resolution features from an image, a pixel decoder that gradually upscales the low-resolution features from the *backbone* output to generate embeddings (*embeddings*) per high-resolution pixel, and finally, a Transformer decoder that operates on image *features* to process object *queries*. The final binary mask predictions are decoded from the *queries* of the *embeddings* per pixel. The architecture is illustrated in Figure 13.

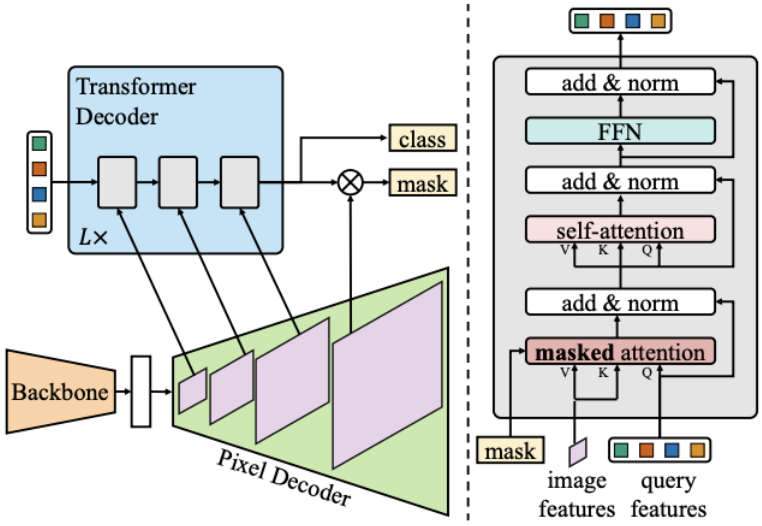

Figure 13: Mask2Former model architecture.

SOURCE: (CHENG et al., 2022, p.3).

Mask2Former adopts the architecture illustrated above, with the Transformer decoder proposed by the authors (Figure 13 on the right) replacing the standard Transformer decoder. The main components of the proposed Transformer decoder include a masked attention operator, which extracts localized features by restricting cross-attention to the foreground region of the predicted mask for each query, rather than attending to the map of *features* complete. To handle small objects, an efficient multiscale strategy is used to utilize high-resolution *features*. This feeds successive maps of *features* from the pixel decoder *features* pyramid into successive layers of the Transformer decoder in a *round-robin* fashion.

2.2.2 Soybean Weed Segmentation

In the context of weed detection, we can mention the work of Herrera, and others (HERRERA; DORADO; RIBEIRO, 2014) where shape and logic descriptors *Fuzzy* were used, more precisely *Fuzzy Decision Making*, for recognition and classification of weeds

among grasses and broad leaves. The shape descriptors were composed of seven Hu moments (ŽUNIĆ; HIROTA; ROSIN, 2010) and six geometric descriptors: perimeter, diameter, length of the shortest axis, length of the longest axis, eccentricity, and area. Support Vector Machines (SVM) were also used to compare results. The work obtained 85.8% accuracy using all extractors and 92.9% using the best set of extractors, as shown in Figures 14 and 15.

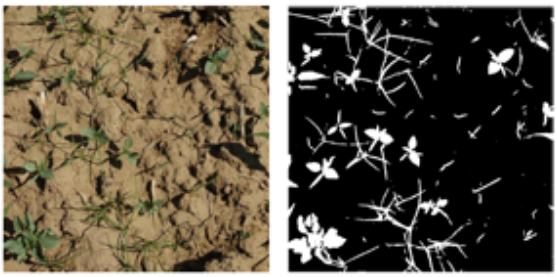

Figure 14: (a) Input image. (b) Segmentation obtained through the Herrera model.

SOURCE: (HERRERA; DORADO; RIBEIRO, 2014, p.3).

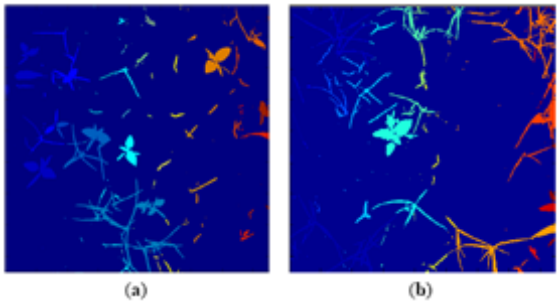

Figure 15: Examples of segmentation with certainty level from the Herrera model.

SOURCE: (HERRERA; DORADO; RIBEIRO, 2014, p.3).

Through the work of Ahmed (AHMED et al., 2012) we see another application of Support Vector Machine, which combines machines with extractors of color, shape, and invariant moments of the image, totaling 14 attributes. Through the *cross-validation* strategy, 97.3% accuracy was obtained. With its results exemplified through Figure 16.

Deep neural networks are a frequently used technology, as mentioned in (UGALE; GUPTA, 2016). The most used architectures are based on convolutional networks (LONG; SHELHAMER; DARRELL, 2015; WANG et al., 2020; CHEN et al., 2017), and the design, for the most part, starts from the same operating concept: *Encoder - Decoder*. *Encoder* is often used in classification networks (also called backbone), being trained on large datasets such as MNIST, CIFAR, and Imagenet. The *Decoder* aggregates

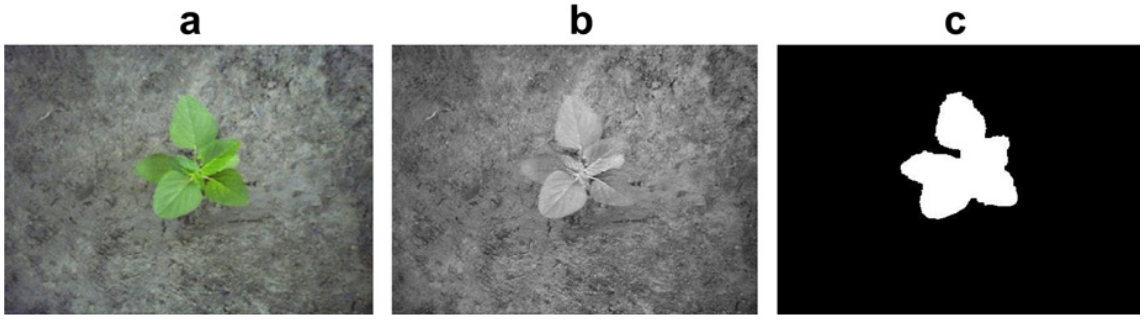

Figure 16: (a) Input image, (b) Adjustment to black and white, (c) Segmentation obtained through the Ahmed model.

SOURCE: (AHMED et al., 2012, p.3).

the attributes, by default called *features*, and later the decoder transforms them into what we call a prediction (response).

For application in soybean studies, there is the work of Ferreira et al. (FERREIRA et al., 2017) which addresses the use of convolutional neural networks, using an architecture that derives from an application of the AlexNet network. Subsequently, the author compares several different algorithms: Support Vector Machine, AdaBoost, and Random Forests, obtaining 98% accuracy in detecting broad-leaved weeds.

Finally, it is important to mention the work of Alex Olsen, and others (OLSEN et al., 2019), which covers the creation of the dataset and creation of the classification model for the captured images. The work involves extracting images unconstrained of weeds in soybeans crops, treating them, applying *data augmentation*, and working with two models: Resnet 50 v2 and Inception v3. The main objective of the article is to evaluate how far it is possible to get in the classification task with networks considered, at the time of the study, to be state-of-the-art in the area of computer vision. The objective proposed by this work aims to implement ViT, a new network that does not use convolutional architecture, subsequently also complementing Alex's work, through the annotation of the dataset and implementation of segmentation.

3 METHODOLOGY

The work is divided into two stages to achieve the proposed objectives. Firstly, the annotation of the dataset was addressed. Secondly, the original Transformer model was adapted for the segmentation task by including convolutional layers aiming for better results. The networks used for comparison were made available by the authors of their respective works, along with their respective weights. The comparisons were made under the same input conditions and training periods, also under the same set of metrics.

The results obtained by pure ViT did not present good accuracy, being explained by the functioning of the Transformer architecture. Apparently, it is not well adapted to the extraction of non-evident attributes, as in the working dataset, which presents weeds next to soybeans and soil (examples are shown following), extremely similar objects. The solution adopted was the addition of a *feature* extractor from a Resnet network (HE et al., 2016). The Resnet is used as *backbone* of one of the networks with the best overall performance in segmentation networks, e.g., the BEiT architecture. Finally, the ViT-hybrid result was extracted, which presents the *backbone* of a purely ViT network, with a feature extractor from a Resnet network.

In the next sections of this chapter, we will detail all the stages of development, starting with Section 3.1 where the segmentation model stage is addressed, and followed by Section 4.1.1, in which the dataset and its annotation are shown.

3.1 Segmentation Model

By default, the choice of backbone contributes considerably to the results. We can highlight that convolutional pipelines progressively apply image downsampling for bulking extraction of features. This enables the network to have a progressive increase in its receptive field. However, this process can bring harm, particularly in the prediction task, causing graininess and low prediction resolution (very generic prediction) (CHO et al., 2014).

There are strategies to mitigate these issues, including training at higher resolutions (WANG et al., 2020), dilated convolutions, optimization of skip connections, and parallel

processing. Even though these techniques contribute greatly to the functioning of convolutional networks, networks with this backbone present bottlenecks due to their main block, convolution.

Convolutions, by definition, are linear operations that limit the receptive field. These operations need to be numerous and compiled on top of each other, causing a large need for computational power and memory to store each downsample.

Therefore, it was decided to adopt a model based on the standard (DOSOVITSKIY et al., 2021) encoder-decoder model. In operation, which will be detailed later, there is the deconstruction of an image in batches (replacement of downsampling), processing of batches, and then feeding the MLP heads. The heads will perform pixel classification (segmentation) through their respective attention maps.

Given the way the model works, it is noticeable that there is no need to store residual images, as in convolutional ones, per step of *downsample*, because, in the way a ViT works, the image is fragmented into *batches*, and these entities are all processed by the same heads, not losing the receptive field of the developed model.

As described previously, the ViT network is an application of *Transformers* networks, but for computer vision tasks. To develop the network, the steps suggested by the authors were used (DOSOVITSKIY et al., 2021). In short, we start with the *patches* creation part, and the *encoding* of these patches, that is, the aggregation of the withdrawal position to this *patch*. Next, we sequentially assemble our *backbone* using the pre-defined structure and the blocks already assembled modularly. These are the normalization blocks, multi-head attention, and the *Multi Layer Perceptron* (MLP). For the classification to work, we forward this data to the multi-attention heads, reaching the final pixel classification through the average of the results of these heads, obtained through a dense *softmax* activation layer.

One can cite as positive points the low complexity of a ViT, in relation to convolutional networks, in terms of layers and parameters, the global attention, and the low need for volatile storage, given that the ViT does not require *upsampling* and *downsampling*. Finally, the possibility of extracting intermediate results is highlighted, to investigate the functioning of the network. This extraction is obtained through the interpretation of the attention maps generated through the average of *scores* of the MLP block.

3.1.1 ViT-Base: a Transformer-only model for Segmentation

For the segmentation task, the structure of the MLP heads was adapted to perform the segmentation, based on the average of the weights obtained, generating a response similar to the attention map, which is a hidden response from the system, extracted through *scores* of the heads. The adaptation consists of a *dropout* frequency of 0.1 (recommended by authors) (DOSOVITSKIY et al., 2021), adding a bi-linear interpolation in the last upsampling operation. Finally, segmentation is carried out through pixel-by-pixel classification

into their respective class, a process popularly called *logits*.

Despite the good performance of the model in the classification task, in the segmentation task, there were some inconsistencies, which were highlighted by the extraction of the attention maps, as shown below, represented in Figures 17 and 18.

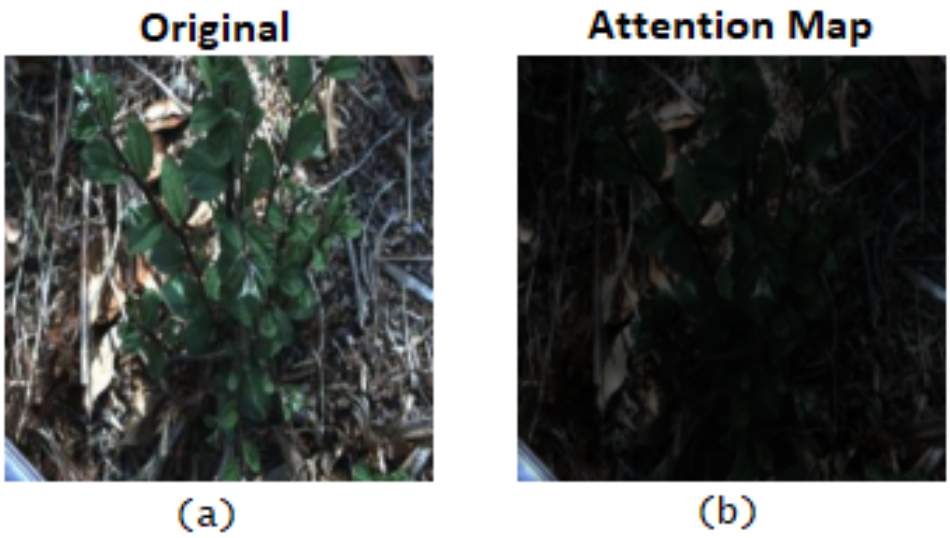

Figure 17: (a) Original image. (b) Attention Map extracted from the *ViT-Base* model.

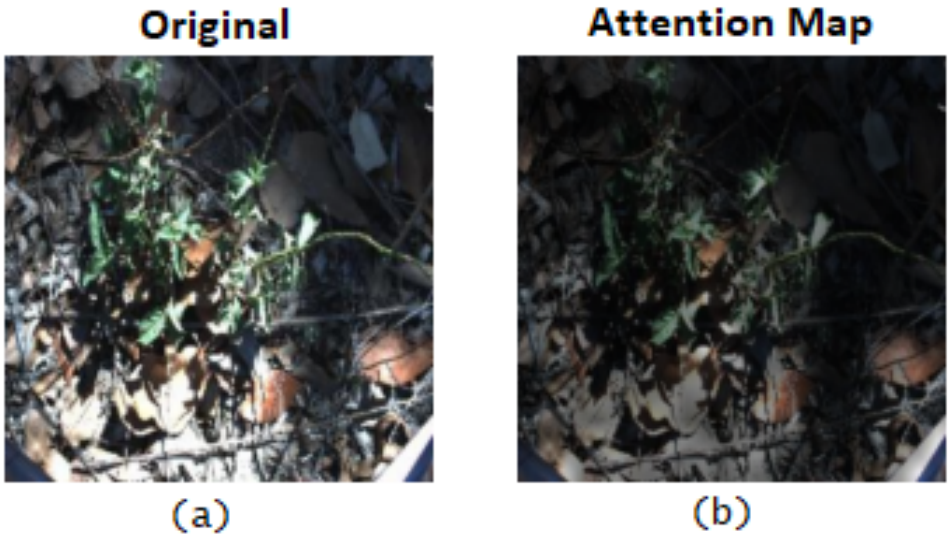

Figure 18: (a) Original image. (b) Attention Map extracted from the *ViT-Base* model.

3.1.2 The Hybrid Convolutional-Transformer Model: VT-Net

To overcome the problems mentioned in Section 3.1.1, we introduce a novel segmentation model, called VT-Net (ViT + Resnet). In this model, a convolutional *decoder* was used, which uses the extraction of attributes from a convolutional network of the Resnet type (HE et al., 2016). This change was proposed because, due to the nature of the problem, the ViT did not present acceptable results. In the tackled task, the images *on the wild* have a lot of information that the model heads could not process efficiently, causing high grain in the resulting image, therefore, an attribute extractor was implemented previously in the *Transformer* layers. In other words, it is replacing the way the network is fed, as illustrated in Figure 19.

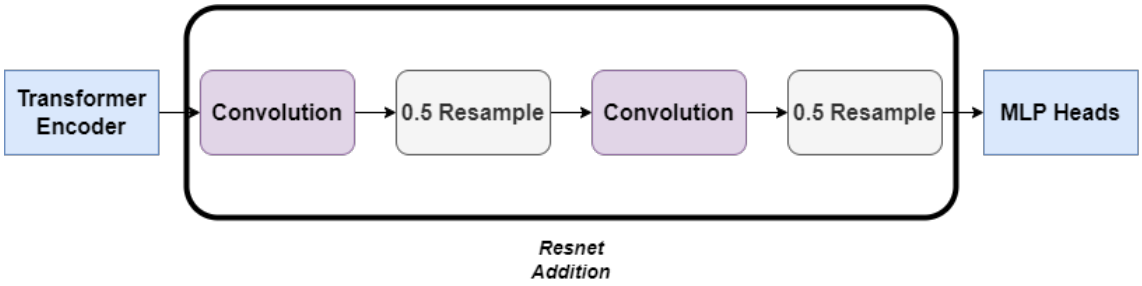

Figure 19: Description of the hybrid adaptation - *Resnet* Block. The information from the Transformer Encoder is fed to the Resnet Addition, instead of directly into the MLP Heads. Inside the Resnet Addition, the information data goes through two convolutions and two resample blocks, as per the Resnet method.

SOURCE: (HE et al., 2016, p.2).

Some inconsistencies were found, as previously presented. These inconsistencies are due to the way attributes are extracted, through ViT's original self-attention system. Given the low resolution, due to the nature of the data obtained, there was a negative effect on the generation of (XIE et al., 2021) attention maps. The idea of applying a hybrid model came from studies on attribute treatments (*features*), and then there was the study and application of a Resnet attribute extractor. The choice derives from the fact that Resnet is a frequent choice within convolutional segmentation networks (UGALE; GUPTA, 2016; FERREIRA et al., 2017). The result, then, is a network with the same structure as *backbone*, but with the addition of some layers, which are expressed through the code present in Appendix B. Attention maps generated by the VT-Net model are seen in Figures 20 and 21. The final VT-Net architecture is shown in Figure 22.

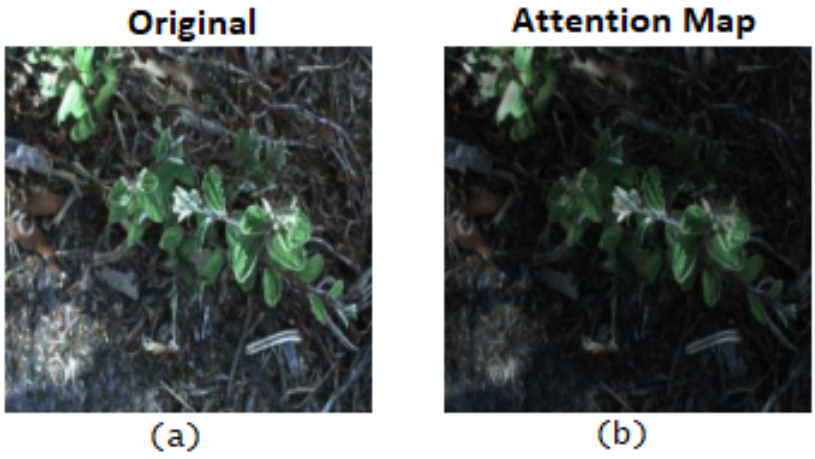

Figure 20: (a) Original image. (b) Attention Map extracted from the VT-Net model.

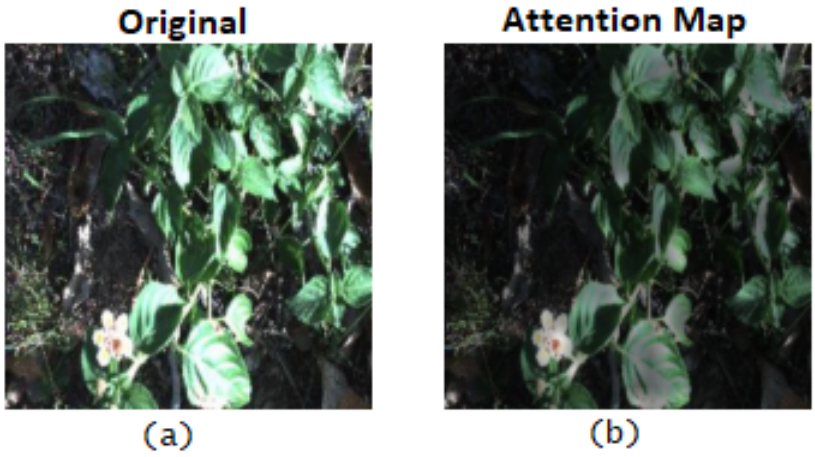

Figure 21: (a) Original image. (b) Attention Map extracted from the VT-Net model.

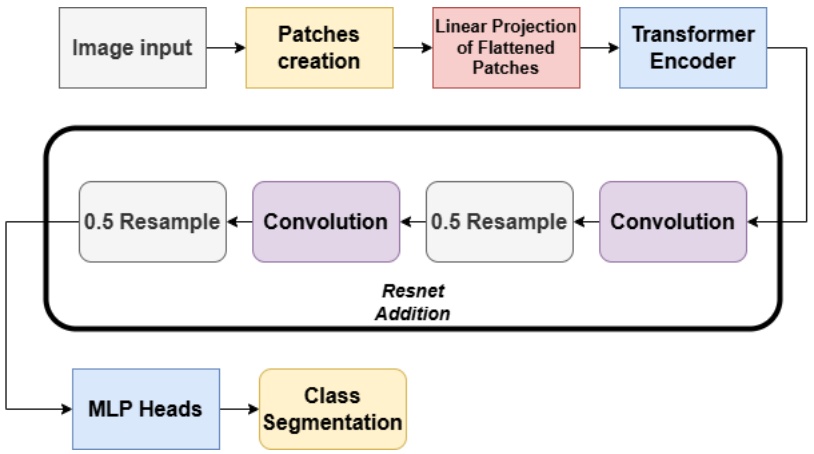

Figure 22: Final architecture of the VT-Net model, which consists of the sample input processing stages of a ViT network, but instead of feeding the data to the MLP heads, we feed it to the Resnet Addition stage.

3.1.3 Adding Shifted Windows: VT-Net-V2

The Swin Transformer method (LIU et al., 2022) consists of replacing the standard multi-head self-attention modules (MSA) with a "shifted window" module inside the Transformer Block, while keeping the other layers the same as the original model. In Figure 23, we can observe the Swin Transformer block. It consists of a shifted window multi-headed self-attention (SW-MSA) module, followed by a two-layer multi-layer perceptron (MLP) module, with a non-linear Gaussian error linear unit (GELU) block between them. A Layer Normalization (LN) is applied before each MSA and MLP block, and a residual connection is also applied after each module or block.

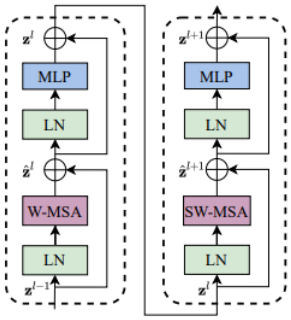

Figure 23: Swin Transformer block.

SOURCE: (LIU et al., 2022, p.4).

The idea of implementing the Shifted Windows technology is based on the premise that this method brings greater efficiency by limiting self-attention computation to non-overlapping local windows while also allowing for cross-window connection (LIU et al., 2022). The model presents an efficient architecture using the self-attention computation within local windows. The windows are dynamically arranged in an even partition of the image in a non-overlapping manner. Assuming that each window contains $M \times M$ patches, the computational complexity of a global MSA and window-based one of an image of $h \times h$ patches is described as,

$$\Omega(\text{MSA}) = 4hwC^2 + 2(hw)^2C, \tag{5}$$

$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC, \tag{6}$$

where the former is quadratic to the number of patches hw, and the latter is linear when M is fixed (set to 7 by author (LIU et al., 2022)). The global self-attention computation in some cases proves to be intractable for a large hw, but the window-based self-attention is scalable (LIU et al., 2022).

The method of window-based self-attention modules presents a lack of connections throughout the windows, causing a limitation in the model's global vision. The solution

presented by the authors is to implement a process of "cross-windows" connections that consist of the implementation of non-overlapping windows that alternate in the consecutive Swin Transformer blocks.

At the start, the module uses regular window partitioning, starting from the top-left pixel, and then the 8×8 feature map is evenly partitioned into 2×2 windows with dimensions of 4×4 ($M = 4$). Following the next module, there's the implementation of the shifting of the data from the layer before. The windows are displaced by $([M/2], [M/2])$ pixels from the original partitioned windows.

By adopting the shifted window partitioning method, the Swin Transformer blocks present the following calculation,

$$\begin{aligned} \hat{z}^{l} &= \text{W-MSA}(\text{LN}(z^{l-1})) + z^{l-1} \\ z^{l} &= \text{MLP}(\text{LN}(\hat{z}^{l})) + \hat{z}^{l}, \\ \hat{z}^{l+1} &= \text{SW-MSA}(\text{LN}(z^{l})) + z^{l}, \\ z^{l+1} &= \text{MLP}(\text{LN}(\hat{z}^{l-1})) + \hat{z}^{l-1}, \end{aligned} \tag{7}$$

where $\hat{z}^{l}$ and z^{l} represent the output features of the Swin-Multi-Head Self Attention (SW-MSA) module in the block *l*, then respectively Window-MSA (W-MSA) and Shifted Window-MSA (SW-MSA) represents the window based multi-head self-attention using regular and shifted window partitioning configurations.

The shifted window partitioning approach introduces connections between neighboring non-overlapping windows in the previous layer and is found to be effective in image classification, object detection, and semantic segmentation, as stated by authors (LIU et al., 2022). The final architecture is represented in Figure 24.

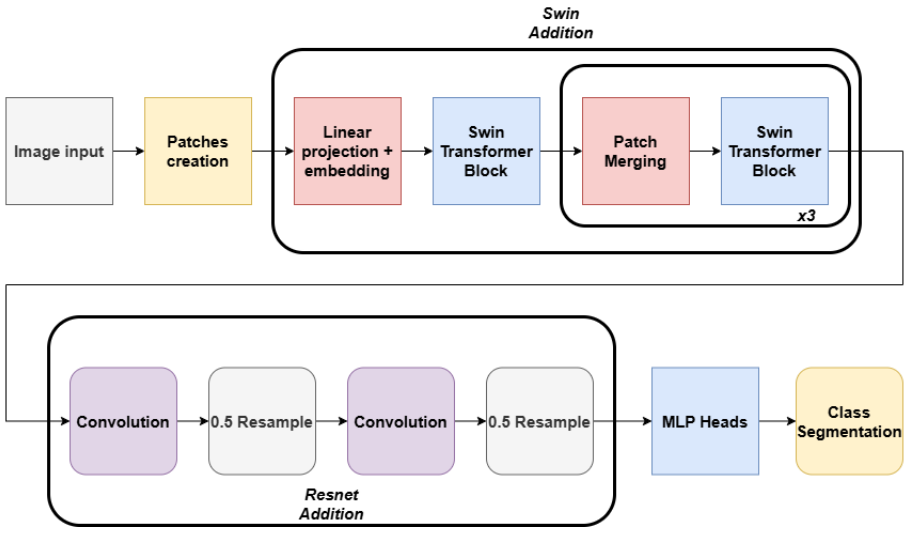

Figure 24: Final architecture of the VT-Net-V2 model. Where the original Transformer Block is swapped for the Swin Addition (Swin Transformer Block), which performs the shifting of the windows. Later that information is fed to the Resnet Addition, as done for the previous model.

4 EXPERIMENTS AND RESULTS

This chapter presents the design, execution, and outcomes of the experiments conducted on both proposed models. The experimental setup, including the research methods, dataset, and dataset annotation, are described in detail. The results of the experiments are then presented systematically, with appropriate performance analysis and visualizations to highlight key findings. This chapter not only provides a comprehensive account of the data collected but also offers an interpretation of the results in relation to the research questions and hypotheses.

4.1 Dataset *Deepweeds*

The dataset used in the present work is a collection of images captured in uncontrolled environments, standing out from the related works presented. The dataset is made with images of Australian soybean weeds that share similar characteristics to Brazilian weeds (VOLL, 1978). The weeds' names are presented in Table 6 and image samples are shown in Figure 25. For illustration, Figure 26 shows three examples of soybean leaves without the presence of weeds.

Table 6: Soybean Weed Species

Popular name	Scientific name
Dão	*Ziziphus mauritiana*
Lantana	*Lantana camara*
Cina-cina	*Parkinsonia aculeata*
White wormwood	*Parthenium hysterophorus*
Yellow Acacia	*Vachellia farnesiana*
Rubber Vine	*Cryptostegia grandiflora*
Zion's Wort	*Eupatorium odoratum*
Broom or Snake	*Gutierrezia sarothrae*

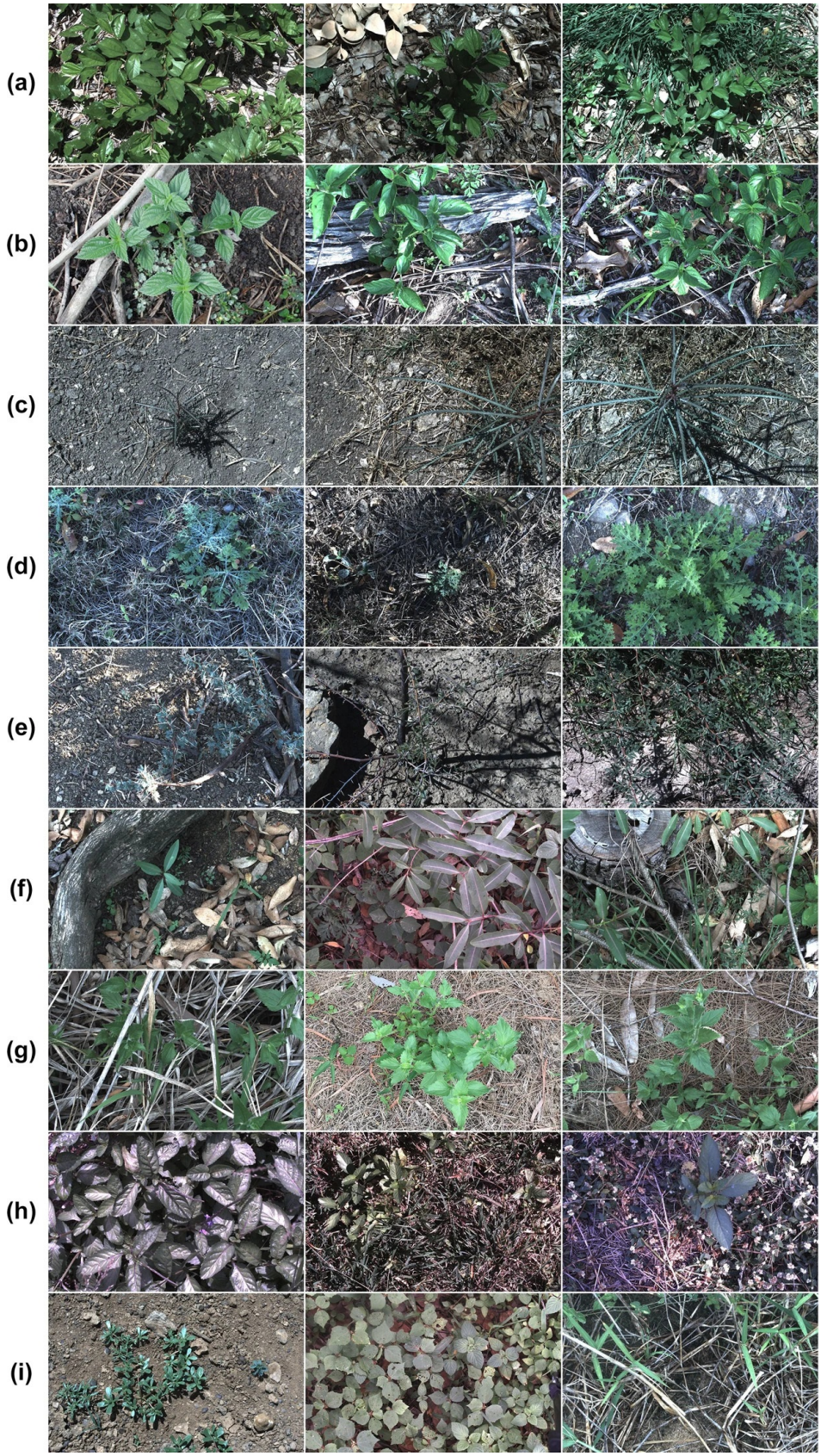

Figure 25: Weed Presentation Table, in order: *a) Dão, b) Lantana, c) Cina-Cina, d) White Lonsa, e) Yellow Acacia, f) Rubber Vine, g) Zion's Wort, h) Broom or Serpent* and i) Weedless images (Soybeans, soil, etc.).

SOURCE: (OLSEN et al., 2019, p.5).

Figure 26: Images showing the soybean leaves.

SOURCE: (OLSEN et al., 2019).

The original images are 256×256 and were resized to 224×224 to carry out the *patching* process correctly, following the suggestion of the author of the ViT network (DOSOVITSKIY et al., 2021). The original dataset presents its image distribution as shown in Table 7.

Table 7: Distribution of Dataset Images

Name in dataset	Number of Images [1]
Dão	1125
Lantana	1064
Cina-cina	1031
White wormwood	1022
Yellow Acacia	1062
Rubber Vine	1009
Zion's Wort	1074
Broom or Snake	1016
Weedless Images	9106
Total images	17509

Finally, it is worth highlighting that, in addition to the presence of images with weeds and negative images (images that only show soybeans or soil), images with visual trash, such as the one below, which presents a plastic bottle together with Lantana leaves, illustrated in Figure 27.

[1]The total amount of weed images is 8403 images, adding to 9106 negatives we reach the grand total of 17509 images.

Figure 27: Examples of negative class. Where the images present foliage, soybeans, and soil.

SOURCE: (OLSEN et al., 2019).

4.1.1 Dataset Annotation

To annotate the images, an application called *Diffgram* (DIFFGRAM, 2022) was used. The raw images (JPEG) were loaded onto the platform, and the image was then annotated using polygons, as shown in Figure 28.

Figure 28: Image annotation procedure. The left column shows the contours of the annotated area. The right column highlights the corresponding annotated areas.

Only weeds that presented the conditions of being outside the shade and above the ground level (excess for low-growing weeds) were noted and were exported in JSON format for use in the segmentation model. The final distribution of annotated images is shown in Table 8.

Table 8: Distribution of Annotated Dataset Images.

Name in dataset	Number of Images
Dão	112
Lantana	106
Cina-cina	103
Losna Branca	102
Yellow Acacia	106
Rubber Vine	101
Zion's Wort	105
Broom or Snake	101
Weedless Images	221

4.2 *Baselines*

To carry out the comparison of the developed network, state-of-the-art networks were chosen, following the respective justifications:

- Segmenter - comparison with a pure *Transfomer* model that performs segmentation using better performing masks,
- CvT - comparison with a hybrid model, but with convolutional *backbone*, with better results,
- Swin Transformer - comparison with Transformer network with moving windows with better overall results,
- SegFormer - comparison with the Transformer network that presents as the second best overall performance, with elements of a Resnet architecture and a purely Transformer architecture,
- BEiT - comparison with the Transformer network with better overall performance, presents a purely Transformer architecture,
- Mask2Former - comparison with the *Transfromer* network that uses masks with better overall results.

4.2.1 Comparison of Models

To consolidate the models, and explain the differences, this sub-section will address the differences between them, starting with the constructive differences in the architectures, expressed through Table 9. For backbones, the last character indicates the model size as defined Table 1, for the Base (B) and Large (L) models, and in (STRUDEL et al., 2021; LIU et al., 2022; WU et al., 2022) for the Small (S) model.

Table 9: Comparison of baseline network architectures.

Method	Backbone	Layers	Params (M)
Segmenter-L	ViT-L	24	307
CvT-W24	NA[1]	24	277
Swin-L	ViT-L	28	197
SegFormer-B5	NA[1]	16	84
BEiT	ViT-L	NI[2]	307
Mask2Former	Swin-L	32[3]	216
VT-Net (Ours)	ViT-B	16	120
VT-Net-V2 (Ours)	ViT-B	18	131

Taking a deeper look at the networks that have the *Transformers* backbone, we highlight the difference between the baseline and the proposed models in Table 10.

Table 10: Comparison of Transformer-based networks architecture.

Method	Requires PE[4]	Features *Overlap*	Projection	Hierarchical
Segmenter-L	yes	*overlap* by masks	linear	no
CvT-W24	no	*overlap* convolutional	convolutional	yes
Swin-L	yes	without *overlap*	linear	yes
SegFormer-B5	no	patches *overlap*	convolutional	yes
BEiT	yes	*overlap* indirect	linear	yes
Mask2Former	yes	without *overlap*	linear	yes
VT-Net (Ours)	yes	without *overlap*	convolutional	no
VT-Net-V2 (Ours)	yes	without *overlap*	convolutional	no

4.3 Ablation Studies

The choice of *backbone* and *patch size* were the two main points analyzed in the ablation experiments shown in Tables 11 and 12. Analyzing Table 11, it is possible to conclude that our models present a number of layers and parameters on par with the other

[1]It is not possible to consider *backbone* applicable to the developed model, given that there are major changes to the network architecture, only the Transformers ”blocks”were used.

[2]Not informed by authors.

[3]It is not reported by the authors in the text, but from the presentation of the architecture and choice of backbone, it can be deduced that there are 32 layers.

[4]PE or *Positional Embedding*, as previously mentioned, is the step of identifying the location of the *patch* taken from the input data.

models. Moreover, one may notice that the Large backbone does not perform as well as the Base version. Finally, through Table 12, it is possible to conclude that medium-sized patches are the best choice for the patch-making process.

Table 11: Pixel-to-Pixel Accuracy (%) and mIoU by choice of *backbone*.

Method	*Backbone*	Layers	Params (M)	Pixel Accuracy	mIoU
VT-NetS (Ours)	ViT-S	14	100	91.54%	0.605
VT-Net-V2S (Ours)	ViT-S	16	110	92.45%	0.622
Segmenter-S	ViT-S	12	22	90.14%	0.595
CvT-13	CvT-13	13	20	**94.66%**	**0.625**
Swin-S	ViT-S	28	50	82.07%	0.542
VT-NetB (Ours)	ViT-B	16	120	93.89%	0.620
VT-Net-V2B (Ours)	ViT-B	18	131	**95.12%**	**0.635**
Segmenter-B	ViT-B	12	86	84.07%	0.555
CvT-21	CvT-21	21	32	88.56%	0.585
Swin-B	ViT-B	28	88	92.20%	0.609
VT-NetL (Ours)	ViT-L	36	881	93.02%	0.610
VT-Net-V2L (Ours)	ViT-L	28	890	**93.95%**	0.621
Segmenter	ViT-L	24	307	93.08%	**0.616**
CvT-W24	CvT-W24	24	277	91.15%	0.603
Swin-L	ViT-L	28	197	88.85%	0.586

Table 12: VT-Net - Pixel to Pixel Accuracy (%) and mIoU for *patch size*.

Method	*Patch Size*	Pixel-to-Pixel Accuracy	mIoU
VT-NetS	32	88.26%	0.607
	16	91.54%	0.605
	8	90.14%	0.567
VT-NetB	32	92.73%	0.618
	16	**93.89%**	**0.620**
	8	90.67%	0.605
VT-NetL	32	90.87%	0.607
	16	93.02%	0.610
	8	92.95%	0.582

4.4 Results

4.4.1 Quantitative Results

To measure the results of the segmentation task, we will use the same metrics used in the classification model, with just the addition of pixel-by-pixel accuracy, and Mean Intersect over Union (mIoU) (see Appendix A), which evaluates the similarity between two images, which will be used to evaluate the formation of the segmentation mask. The model input is a JSON file, which contains the path to the image and the formation vertices

of the annotated regions, the model output is also a JSON file with the vertices of the segmentation regions. The metrics of the results are presented through Tables 13 and 14.

The results were very close to the state-of-the-art networks for the study in question, showing that there are alternatives to convolutional networks. The advantages of using a network with a Transformer backbone, in contrast to a purely convolutional one, can be cited as not needing extensive memory for image processing, due to the *upsampling* and *processes downsampling*, the low complexity of the model and easy interpretation of the model's operation through studies of the network's attention maps.

Table 13: Results Obtained - Accuracy, Precision and *Recall* - VT-Net vs. VT-Net-V2.

Class	Top-1 Accuracy		Precision		Recall	
	VT-Net	VT-Net-V2	VT-Net	VT-Net-V2	VT-Net	VT-Net-V2
Dão	94.07%	95.01%	92%	93%	82%	83%
Lantana	95.53%	96.38%	93%	94%	83%	84%
Cina-Cina	96.35%	97.45%	94%	95%	84%	85%
White wormwood	97.55%	98.75%	95%	96%	85%	86%
Yellow Acacia	97.41%	96.21%	94%	96%	85%	86%
Rubber Vine	96.65%	97.35%	96%	95%	84%	85%
Zion's Wort	98.09%	98.29%	96%	96%	85%	85%
Broom or Snake	97.81%	97.91%	96%	97%	85%	86%
Weedless Images	98.17%	99.07%	96%	97%	85%	87%
Average	96.85%	97.38%	95%	96%	84%	85%

Table 14: Results Obtained - Pixel Accuracy and mIoU - VT-Net vs. VT-Net-V2.

Class	Pixel Acc		mIoU	
	VT-Net	VT-Net-V2	VT-Net	VT-Net-V2
Dão	91.20%	93.11%	0.611	0.623
Lantana	92.62%	93.54%	0.600	0.615
Cina-Cina	93.41%	93.51%	0.649	0.654
White Wormwood	94.57%	95.87%	0.636	0.641
Yellow Acacia	94.43%	95.63%	0.646	0.651
Rubber Vine	93.70%	94.36%	0.594	0.605
Zion's Wort	95.10%	97.04%	0.626	0.632
Broom or Snake	94.83%	96.84%	0.647	0.652
Weedless Images	95.17%	96.18%	0.625	0.643
Average	93.89%	95.12%	0.621	0.635

Table 15 shows the results from our networks using pixel-by-pixel accuracy and mIoU metrics. Next, Table 16 illustrates the results by class relevant to each network.

Table 15: Results Obtained - Pixel Accuracy and mIoU.

Model	Pixel Acc	mIoU
Segmenter-L	89.13%	0.563
CvT-W24	89.32%	0.603
Mask2Former	87.51%	0.596
BEiT	**95.51%**	**0.645**
SegFormer	91.22%	0.616
Swin-L	87.08%	0.586
VT-Net (Ours)	93.89%	0.620
VT-Net-V2 (Ours)	95.12%	0.635

Table 16: Pixel Accuracy (%) by Class.

Class	Segm.	CvT	SegF.	BEiT	M2F	Swin.	VT-Net	VT-Net-V2
Dão	88.34	88.53	89.19	94.66	90.41	88.74	91.20	93.11
Lantana	89.71	89.91	87.78	96.13	91.82	87.35	92.62	93.54
Cina.	90.48	90.68	88.53	96.95	92.61	88.09	93.41	93.51
White..	91.6	91.80	89.63	98.16	93.75	89.19	94.57	95.87
Yellow.	91.47	91.67	89.50	94.60	93.62	89.06	94.43	95.63
Rubber.	90.76	90.96	88.81	97.26	92.89	88.37	93.70	94.36
Zion's.	92.12	92.32	90.14	98.71	94.28	89.69	95.10	97.04
Snake.	91.85	92.06	89.88	98.43	94.01	89.43	94.86	96.84
Weedless	92.19	92.39	90.20	94.77	94.35	89.76	95.17	96.18
Average	90.95	91.15	89.30	97.46	93.08	88.85	93.89	95.12

From the analysis of the tables above it is possible to determine that the second version of the model, with the addition of shifted windows, excels the first proposed model in all metrics presented.

4.4.2 Qualitative Results

This section will present some results of the network operation, extracting the attention maps as masks, the first image being the original, the second the *Ground Thruth* mask, and the third the model prediction mask (response), illustrated by Figures 29 to 36.

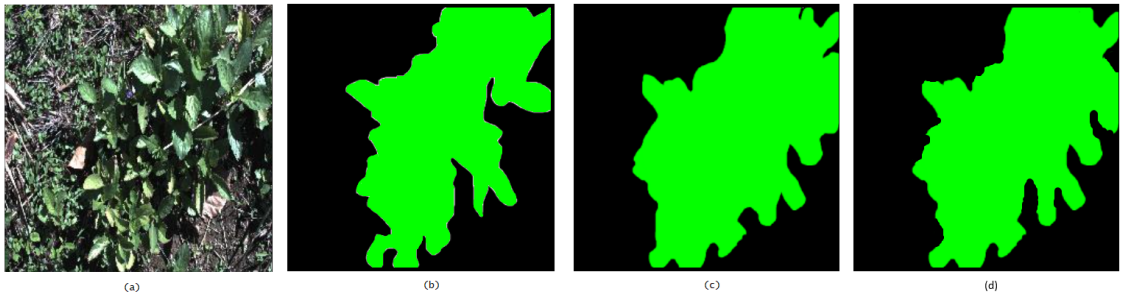

Figure 29: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Dão (Green Mask).

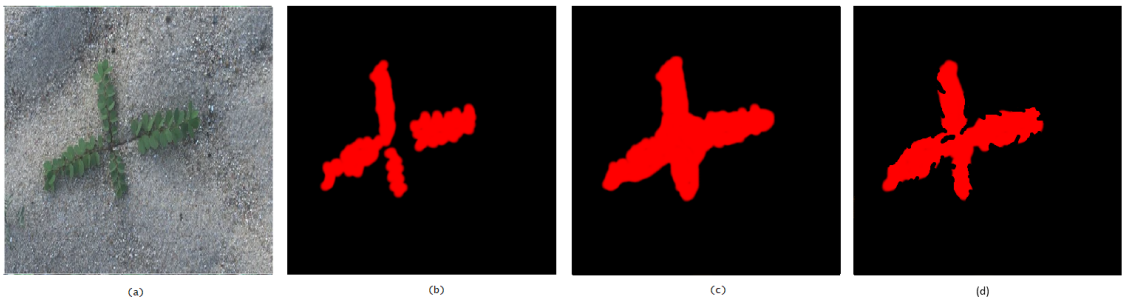

Figure 30: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Negative (Red Mask)

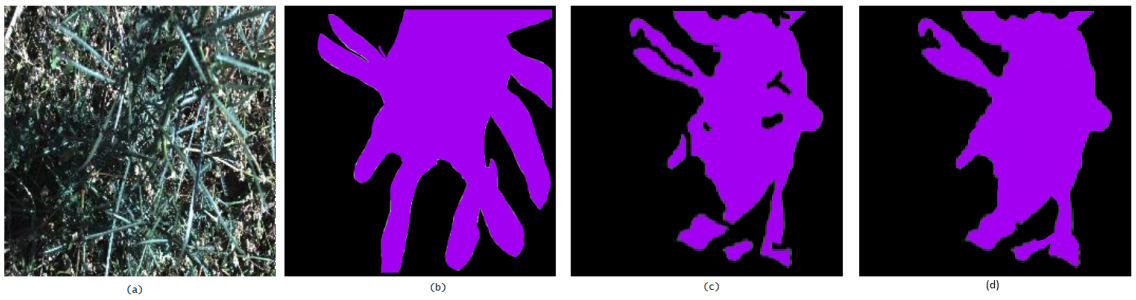

Figure 31: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Parkinsonia (Purple Mask)

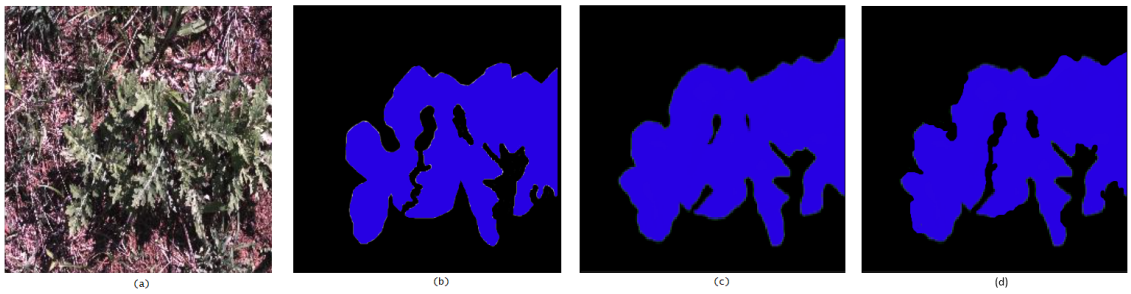

Figure 32: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Parthenium (Blue Mask)

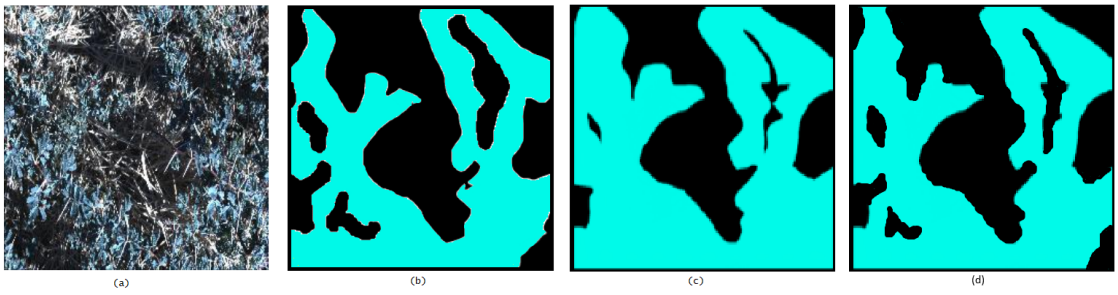

Figure 33: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Prickly Acacia (CYan Mask)

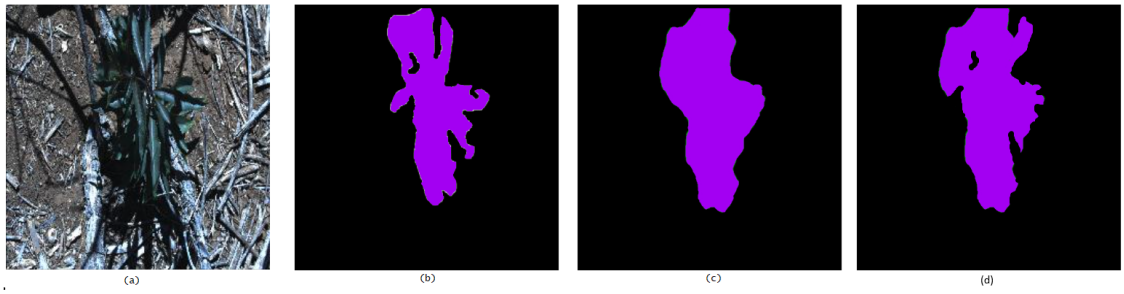

Figure 34: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Parkinsonia (Purple Mask)

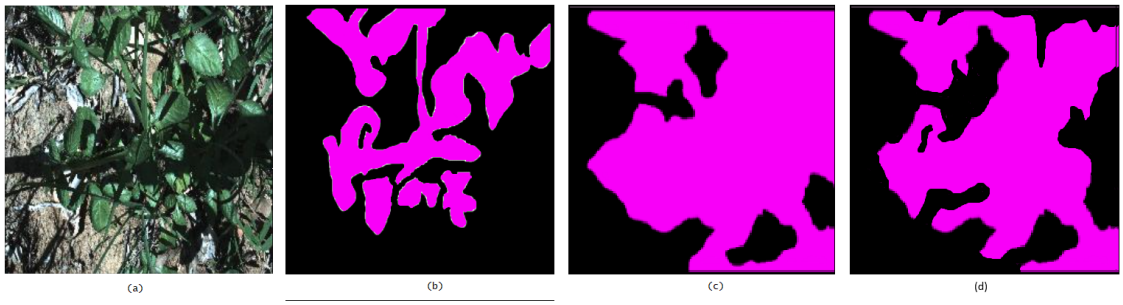

Figure 35: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Lantana (Pink Mask)

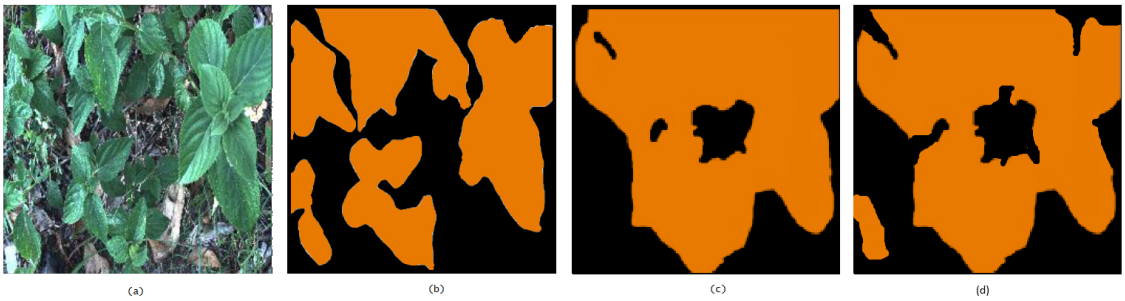

Figure 36: (a) Original image. (b) Segmentation mask ground-truth. (c) VT-Net Segmentation. (d) VT-Net-V2 Segmentation. Class: Zion's Wort (Orange Mask)

5 CONCLUSION

This work achieved the main proposed objective, which is to build a Vision Transformer segmentation model to perform class segmentation of weeds in soybeans crops through the adoption of a hybrid model, which uses convolution as an adaptation for the original ViT model. Furthermore, the model was improved by implementing the shifted window methodology presented by the Swin Transformer model (LIU et al., 2022), creating the final model that is close to the state-of-the-art solution (BEiT), and uses both convolution and shifted windows as additional resources.

It can also be concluded from this work that Vision Transformers networks are an interesting alternative to convolutional networks. Through the presented studies, we performed complete segmentation tasks using a hybrid model. It can also be said that the fusion between a ViT and a common network can generate interesting results, combining the self-attention methodology of Transformers networks with the robustness of a convolutional network.

5.1 Future Works

For future work, it would be interesting to focus on improving the dataset's quality by capturing more accurate images. In the current version, many images contain weeds, but in conditions that were not very evident, such as shadows, occlusions, and reduced leaf size. We can also mention feature extraction improvement to make the most of the nature of Transformers networks.

REFERENCES

AHMED, F. et al. Classification of crops and weeds from digital images: A support vector machine approach. *Crop Protection*, Elsevier, v. 40, p. 98–104, 2012.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. p. 1–15, 2015.

BAO, H. et al. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

BELLO, I. et al. Attention augmented convolutional networks. *International Conference on Computer Vision (ICCV)*, p. 1–13, 2020.

BENEDETTI, D. et al. DNA damage and epigenetic alteration in soybean farmers exposed to complex mixture of pesticides. *Mutagenesis*, Oxford University Press UK, v. 33, n. 1, p. 87–95, 2018.

BIANCHI, M. et al. Interferência de raphanus sativus na produtividade de cultivares de soja. *Planta Daninha*, SciELO Brasil, v. 29, p. 783–792, 2011.

CARION, N. et al. End-to-end object detection with transformers. In: SPRINGER. *European conference on computer vision*. [S.l.], 2020. p. 213–229.

CHEN, K.; HUANG, J. Research on the design and application of house price prediction algorithms and model based on machine learning. In: *2023 International Conference on Internet of Things, Robotics and Distributed Computing (ICIRDC)*. [S.l.: s.n.], 2023. p. 815–821.

CHEN, L.-C. et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 40, n. 4, p. 834–848, 2017.

CHENG, B. et al. Masked-attention mask transformer for universal image segmentation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2022. p. 1290–1299.

CHO, K. et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

CORDTS, M. et al. The cityscapes dataset for semantic urban scene understanding. In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016.

CORREIA, N. M.; DURIGAN, J. C. Controle de plantas daninhas na cultura de soja resistente ao glyphosate. *Bragantia*, SciELO Brasil, v. 69, p. 319–327, 2010.

DIFFGRAM. *Open Source Data Labeling Platform*. 2022. Available at: https://diffgram.com/main/. Accessed: 10-10-2021.

DOSOVITSKIY, A. et al. An image is worth 16x16 words: Transformers for image recognition at scale. In: *ICLR*. [S.l.]: OpenReview.net, 2021.

EVERINGHAM, M. et al. The pascal visual object classes challenge: A retrospective. In: . [S.l.: s.n.], 2015. v. 111, n. 1, p. 98–136.

FERREIRA, A. dos S. et al. Weed detection in soybean crops using convnets. *Computers and Electronics in Agriculture*, Elsevier, v. 143, p. 314–324, 2017.

FLECK, N. G.; CANDEMIL, C. R. G. Interferência de plantas daninhas na cultura da soja (glycine max (l.) merrill). *Ciência Rural*, SciELO Brasil, v. 25, p. 27–32, 1995.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016.

HE, K. et al. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 770–778.

HERRERA, P. J.; DORADO, J.; RIBEIRO, Á. A novel approach for weed type classification based on shape descriptors and a fuzzy decision-making method. *Sensors*, MDPI, v. 14, n. 8, p. 15304–15324, 2014.

IBRAHIM, A. M. et al. Real-time collision warning system based on computer vision using mono camera. In: *2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES)*. [S.l.: s.n.], 2020. p. 60–64.

JAIN, R.; KASTURI, R.; SCHUNCK, B. G. *Machine vision*. [S.l.]: McGraw-hill New York, 1995.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LIU, Z. et al. Video swin transformer. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2022. p. 3202–3211.

LOKANAN, M.; LIU, S. Predicting fraud victimization using classical machine learning. *Entropy*, v. 23, n. 3, 2021. ISSN 1099-4300. Available at: ⟨https://www.mdpi.com/1099-4300/23/3/300⟩.

LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 3431–3440.

NASCIMENTO, G. do et al. A perception system for an autonomous pesticide boom sprayer. In: IEEE. *Latin American Robotics Symposium (LARS)*. [S.l.], 2019. p. 86–91.

NICOLOPOULOU-STAMATI, P. et al. Chemical pesticides and human health: the urgent need for a new concept in agriculture. *Frontiers in public health*, Frontiers Media SA, v. 4, p. 148, 2016.

NITIN; GUPTA, S. B. Artificial intelligence in smart agriculture: Applications and challenges. *CURRENT APPLIED SCIENCE AND TECHNOLOGY*, v. 24, n. 2, p. 1–24, 2024.

OLSEN, A. et al. DeepWeeds: A multiclass weed species image dataset for deep learning. *Scientific reports*, Nature Publishing Group, v. 9, n. 1, p. 1–12, 2019.

PANDA, S. P. Automated speech recognition system in advancement of human-computer interaction. In: *2017 International Conference on Computing Methodologies and Communication (ICCMC)*. [S.l.: s.n.], 2017. p. 302–306.

PAVLOV, V. et al. Application for video analysis based on machine learning and computer vision algorithms. In: *14th Conference of Open Innovation Association FRUCT*. [S.l.: s.n.], 2013. p. 90–100.

PRINCE, S. J. *Computer vision: models, learning, and inference*. [S.l.]: Cambridge University Press, 2012.

PRIYA, L.; G, I. R.; RAJAMANI, V. Crop disease detection and monitoring system. *International Journal of Recent Technology and Engineering (IJRTE)*, v. 8, p. 3050–3053, 06 2020.

PU, Q.; HU, B. Intelligent movie recommendation system based on hybrid recommendation algorithms. In: *2023 International Conference on Ambient Intelligence, Knowledge Informatics and Industrial Electronics (AIKIIE)*. [S.l.: s.n.], 2023. p. 1–5.

RAKSHITHA, C.; SELVAN, C. Improving cctv footage in computer vision using deep learning techniques. In: *2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS)*. [S.l.: s.n.], 2023. p. 1–5.

RAMACHANDRAN, P. et al. Stand-alone self-attention in vision models. *Advances in Neural Information Processing Systems*, v. 32, 2019.

SHAFAIT, F.; KEYSERS, D.; BREUEL, T. Pixel-accurate representation and evaluation of page segmentation in document images. In: . [S.l.: s.n.], 2006. v. 1, p. 872–875.

SHARMA, S. Precision agriculture: Reviewing the advancements, technologies, and applications in precision agriculture for improved crop productivity and resource management. v. 4, p. 41–45, 07 2023.

SHIT, S.; DAS, D. K.; RAY, D. N. Real-time object detection in deep foggy conditions using transformers. In: *2023 3rd International conference on Artificial Intelligence and Signal Processing (AISP)*. [S.l.: s.n.], 2023. p. 1–5.

SILVA, A. et al. Densidades de plantas daninhas e épocas de controle sobre os componentes de produção da soja. *Planta Daninha*, SciELO Brasil, v. 26, p. 65–71, 2008.

STRUDEL, R. et al. Segmenter: Transformer for semantic segmentation. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2021. p. 7262–7272.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, v. 27, 2014.

SZELISKI, R. *Computer vision: algorithms and applications*. [S.l.]: Springer Science & Business Media, 2010.

TARAN, V. et al. Performance evaluation of deep learning networks for semantic segmentation of traffic stereo-pair images. In: *Proceedings of the 19th International Conference on Computer Systems and Technologies*. ACM, 2018. (CompSysTech'18). Available at: ⟨http://dx.doi.org/10.1145/3274005.3274032⟩.

TERRA, F. et al. Autonomous agricultural sprayer using machine vision and nozzle control. *Journal of Intelligent & Robotic Systems*, Springer, v. 102, n. 2, p. 1–18, 2021.

TERRA, F.; ROSA, G. da; DREWS, P. Evaluation of the pressure-flow relationship in a boom of an autonomous robotic agricultural sprayer. In: IEEE. *Latin American Robotics Symposium (LARS)*. [S.l.], 2019. p. 228–233.

TERRA, F. P. et al. A low-cost prototype to automate agricultural sprayers. *IFAC-PapersOnLine*, Elsevier, v. 53, n. 2, p. 15835–15840, 2020.

UGALE, V.; GUPTA, D. A comprehensive survey on agricultural image processing. *International Journal of Science and Research, v5 (1), p133-135*, 2016.

VARGAS, L.; ROMAN, E. S. Manejo e controle de plantas daninhas na cultura de soja (in portuguese). Passo Fundo: Embrapa Trigo, 2006., 2006.

VASWANI, A. et al. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017.

VOLL, E. Controle de ervas daninhas em plantio convencional de soja, no norte do paraná. In: *In: SEMINARIO BRASILEIRO DE HERBICIDAS E ERVAS DANINHAS*. [S.l.: s.n.], 1978.

WANG, J. et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 43, n. 10, p. 3349–3364, 2020.

WEBER, F. et al. A low cost system to optimize pesticide application based on mobile technologies and computer vision. In: IEEE. *Latin American Robotics Symposium (LARS)*. [S.l.], 2018. p. 345–350.

WU, H. et al. Cvt: Introducing convolutions to vision transformers. In: *Proceedings of the IEEE/CVF international conference on computer vision*. [S.l.: s.n.], 2021. p. 22–31.

WU, K. et al. *TinyViT: Fast Pretraining Distillation for Small Vision Transformers*. 2022. Available at: ⟨https://arxiv.org/abs/2207.10666⟩.

XIE, E. et al. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, v. 34, p. 12077–12090, 2021.

ZHAO, H.; JIA, J.; KOLTUN, V. Exploring self-attention for image recognition. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2020. p. 10076–10085.

ŽUNIĆ, J.; HIROTA, K.; ROSIN, P. L. A hu moment invariant as a shape circularity measure. *Pattern Recognition*, Elsevier, v. 43, n. 1, p. 47–57, 2010.

A ADDITIONAL TECHNICAL INFORMATION

***mIoU* - Mean Intersections per Class**: The mIoU (TARAN et al., 2018) can be described as an expansion of the IoU (*Intersection over Union*), translated as Intersection by Union, in short the IoU is a way of checking the similarity between the *Ground Truth* (annotated image) and the *Predicted* which represents the image resulting from the segmentation algorithm. However, the use of IoU is common to classification and segmentation problems of two classes (positive and negative), for our task, which is a multi-class task, an expansion of this calculation was used to deal with multi-classes.

***Pixel Accuracy* - Accuracy per pixel**: Pixel accuracy (SHAFAIT; KEYSERS; BREUEL, 2006) is commonly related to each class separately, so I presented the metrics per class. When considering pixel accuracy per class, we are essentially evaluating one binary mask at a time. A true positive represents a pixel that is correctly predicted to belong to a given class, while a true negative represents a pixel that is correctly identified as not belonging to a given class, the calculation was done directly by *callbacks*, having the following equation 8:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

Where TP is true positives (*true positive*), TN is true negatives (*true negatives*), FP is false positives (*false positives*) and FN is false negatives (*false negatives*).

B IMPLEMENTATION

During the implementation of the work, the publicly accessible *Framework* Keras was used. The codes were all developed without the need for local (physical) operation, only being operated via Kaggle and Google Colab cloud services

Training and network configuration parameters:

```
# Operating Parameters

learning_rate = 0.001
weight_decay = 0.0001
NUM_EPOCHS = 15
MAX_EPOCH = 20
RAW_IMG_SIZE = (256, 256)
IMG_SIZE = (224, 224)
INPUT_SHAPE = (IMG_SIZE[0], IMG_SIZE[1], 3)
BATCH_SIZE = 32
FOLDS = 5
STOPPING_PATIENCE = 32
LR_PATIENCE = 16
INITIAL_LR = 0.0001

# ViT-Base Parameters

patch_size = 16 # Size of the patches to be extracted.
num_patches = (224 // patch_size) ** 2
projection_dim = 64
num_heads = 12 # Number of heads
transformer_units = [
     projection_dim * 2,
     projection_dim,
]

transformer_layers = 12 # Size of transformer layers.
mlp_head_units = [2048, 1024] # Size of heads.
```

Classification network, which is used as the backbone of the segmentation network:

```
  inputs = layers.Input(shape=INPUT_SHAPE)
  # Patch creation
  patches = Patches(patch_size)(inputs)
  # Encode of patches.
  encoded_patches = PatchEncoder(num_patches, projection_dim)(
patches)

  # Block transformer layers (being layer limit range)
  for _ in range(transformer_layers):
      # Layer normalization 1.
      x1 = layers.LayerNormalization(epsilon=1e-6)(encoded_patches)
      # MLP Layer
      attention_output = layers.MultiHeadAttention(
          num_heads=num_heads, key_dim=projection_dim, dropout=0.1
      )(x1, x1)
      # Skip connection 1.
      x2 = layers.Add())([attention_output, encoded_patches])
      # Layer normalization 2.
      x3 = layers.LayerNormalization(epsilon=1e-6)(x2)
      #MLP.
      x3 = mlp(x3, hidden_units=transformer_units, dropout_rate=0.1)
      # Skip connection 2.
      encoded_patches = layers.Add() ([x3, x2])

  # Create a Tensor [batch_size, projection_dim].
  representation = layers.LayerNormalization(epsilon=1e-6)(
encoded_patches)
  representation = layers.Flatten() (representation)
  representation = layers.Dropout(0.5)(representation)
  # Add MLP.
  features = mlp(representation, hidden_units=mlp_head_units,
dropout_rate=0.5)
  # Classification
  result = layers.Dense(NUM_CLASSES,activation="sigmoid")(features)
  # Creates the Keras model.
  model = keras.Model(inputs=inputs, outputs=result)
  return model
```

Addition of hybrid adaptation to the segmentation model:

```
class ViTSegmentation(ViT): # Receives the ViT backbone.
     def __init__(self, num_classes, path=None, **kwargs):

          features = kwargs["features"] if "features" in kwargs else 256

          kwargs["use_bn"] = True
          # Add segmentation.
          head = nn.Sequential(
              nn.Conv2d(features, features, kernel_size=3, padding=1,
    bias=False),
              nn.BatchNorm2d(features),
              nn.ReLU(True),
              nn.Dropout(0.1, False),
              nn.Conv2d(features, num_classes, kernel_size=1),
              Interpolate(scale_factor=2, mode="bilinear", align_corners
    =True),
          )

          super().__init__(head, **kwargs)
          # Additional fusion layers.
          self.auxlayer = nn.Sequential(
              nn.Conv2d(features, features, kernel_size=3, padding=1,
    bias=False),
              nn.BatchNorm2d(features),
              nn.ReLU(True),
              nn.Dropout(0.1, False),
              nn.Conv2d(features, num_classes, kernel_size=1),
          )
          # Crash prevention if it can't find the weights.
          if path is not None:
              self.load(path)
```