Laura Elena Cué La Rosa

Crop Recognition from Multitemporal SAR Image Sequences Using Deep Learning Techniques

DISSERTAÇÃO DE MESTRADO

DEPARTAMENTO DE ENGENHARIA ELÉTRICA
Programa de Pós-graduação em Engenharia Elétrica

Rio de Janeiro
April 2018

Pontifícia Universidade Católica
do Rio de Janeiro

Laura Elena Cué La Rosa

Crop Recognition from Multitemporal SAR Image Sequences Using Deep Learning Techniques

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–Graduação em Engenharia Elétrica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Elétrica.

Advisor: Prof. Raul Queiroz Feitosa

Rio de Janeiro
April 2018

Pontifícia Universidade Católica
do Rio de Janeiro

Laura Elena Cué La Rosa

Crop Recognition from Multitemporal SAR Image Sequences Using Deep Learning Techniques

Dissertation presented to the Programa de Pós-Graduação em Engenharia Elétrica of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Engenharia Elétrica. Approved by the undersigned Examination Committee.

Prof. Raul Queiroz Feitosa
Advisor
Departamento de Engenharia Elétrica PUC-Rio

Prof.ª Marley Maria Bernardes Rebuzzi Vellasco
Departamento de Engenharia Elétrica – PUC-Rio

Prof. Jefersson Alex dos Santos
UFMG

Prof.ª Ieda Del'Arco Sanches
INPE

Prof. Márcio da Silveira Carvalho
Vice Dean of Graduate Studies
Centro Técnico Científico PUC-Rio

Rio de Janeiro, April the 16th, 2018

Laura Elena Cué La Rosa

The author received his bachelor's degree in Biomedical Engineering from Jose Antonio Echeverria Polytechnic Institute in Havana, Cuba, 2013.

Bibliographic data

Cué La Rosa, Laura Elena

Crop Recognition from Multitemporal SAR Image Sequences Using Deep Learning Techniques / Laura Elena Cué La Rosa; advisor: Raul Queiroz Feitosa. – Rio de Janeiro: PUC-Rio, Departamento de Engenharia Elétrica, 2018.

v., 96 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica.

Inclui bibliografia

1. Engenharia Elétrica – Teses. 2. Reconhecimento de Culturas;. 3. Sensoriamento Remoto;. 4. Aprendizado Profundo;. 5. Análise multitemporal;. 6. Sentinel-1. I. Feitosa, Raul Queiroz. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. III. Título.

CDD: 621.3

Acknowledgments

My utmost appreciation to my advisor, Prof. Raul Queiroz Feitosa, for his generous support, his advices and leadership throughout the development of my dissertation. The members of staff at Computer Vision Lab at PUC-Rio for sharing their company, friendship and valuable scientific advices.

I would like to thank to Giovanny Meneses and Yadislen Acosta for their valuable contribution in the last stage of my research.

I thank PUC-Rio and CAPES for the financial support as well as to Damian Bargiel from Technische Universität Darmstadt to provide the reference to the Hanover dataset.

I want to thank to my family and friends from their love and support throughout my life, and most importantly, my appreciation to my mother, Carmen Elena who have always encouraged me and to whom I dedicate this work.

Abstract

Cué La Rosa, Laura Elena; Feitosa, Raul Queiroz (Advisor). **Crop Recognition from Multitemporal SAR Image Sequences Using Deep Learning Techniques**. Rio de Janeiro, 2018. 96p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

The present dissertation aims to evaluate a set of deep learning (DL) techniques for crop mapping from multitemporal sequences of SAR images. Three methods were considered in this study: Autoencoders (AEs), Convolutional Neural Networks (CNNs) and Fully Convolutional Networks (FCNs). The analysis was based on two databases containing image sequences generated by the Sentinel-1A. The first database covers a temperate region that presents a comparatively simpler dynamics, and second database of a tropical region that represents a scenario with complex dynamics. In all cases, a Random Forest (RF) classifier operating on texture features derived from co-occurrence matrices was used as baseline. For the temperate region, DL techniques consistently produced better results than the RF approach, with AE being the best one in almost all experiments. In the tropical region the DL approaches performed similar to RF, alternating as the best performing one for different experimental setups. By and large, CNNs achieved the best or next to the best performance in all experiments. Although the FCNs have performed well, the full potential was not fully exploited in our experiments, mainly due to the difficulty of balancing the number of training samples among the crop types. The dissertation also proposes two post-processing strategies that exploit prior knowledge about the crop dynamics in the target site. Experiments have shown that such techniques can significantly improve the recognition accuracy, in particular for less abundant crops.

Keywords

Crop Recognition; Remote Sensing; Deep Learning; Multitemporal analysis; Sentinel-1

Resumo

Cué La Rosa, Laura Elena; Feitosa, Raul Queiroz. **Reconhecimento de Culturas Agrícolas a partir de Sequencias Multitemporais de Imagens SAR utilizando Técnicas de Aprendizado Profundo**. Rio de Janeiro, 2018. 96p. Dissertação de Mestrado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

A presente dissertação tem como objetivo avaliar um conjunto de técnicas de aprendizado profundo para o reconhecimento de culturas agrícolas a partir de sequências multitemporais de imagens SAR. Três métodos foram considerados neste estudo: Autoencoders (AEs), Convolutional Neural Networks (CNNs) and Fully Convolutional Networks (FCNs). A avaliação experimental baseou-se em duas bases de dados contendo sequências de imagens geradas pelo sensor Sentinel- 1A. A primeira base cobre uma região tropical e a segunda uma região de clima temperado. Em todos os casos, utilizou-se como referência para comparação um classificador Random Forest (RF) operando sobre atributos de textura derivados de matrizes de co-ocorrência. Para a região de clima temperado que apresenta menor dinâmica agrícola as técnicas de aprendizado profundo produziram consistentemente melhores resultados do que a abordagem via RF, sendo AEs o melhor em praticamente todos os experimentos. Na região tropical, onde a dinâmica é mais complexa, as técnicas de aprendizado profundo mostraram resultados similares aos produzidos pelo método RF, embora os quatro métodos tenham se alternado como o de melhor desempenho dependendo do número e das datas das imagens utilizadas nos experimentos. De um modo geral, as RNCs se mostraram mais estáveis do que os outros métodos, atingindo o melhores resultado entre os métodos avaliados ou estando muito próximos destes em praticamente todos os experimentos. Embora tenha apresentado bons resultados, não foi possível explorar todo o potencial das RTCs neste estudo, sobretudo, devido à dificuldade de se balancear o número de amostras de treinamento entre as classes de culturas agrícolas presentes na área de estudo. A dissertação propõe ainda duas estratégias de pós-processamento que exploram o conhecimento prévio sobre a dinâmica das culturas agrícolas presentes na área alvo. Experimentos demonstraram que tais técnicas podem produzir um aumento significativo da acurácia da classificação, especialmente para culturas menos abundantes.

Palavras-chave

Reconhecimento de Culturas; Sensoriamento Remoto; Aprendizado Profundo; Análise multitemporal; Sentinel-1

Table of contents

List of figures

List of tables

List of Symbols and Abreviations

AA	Average Class Accuracy
Acc	Class Accuracy
AEs	Autoencoders
AE-PB	AE pixel-based framework
BN	Batch Normalization
CNNs	Convolutional Neural Networks
CNN-PC	CNN patch-classification framework
CRFs	Conditional Random Fields
dB	decibel
DBNs	Deep Belief Networks
DEM	Digital Elevation Model
DL	Deep Learning
F1	F1 score
FCNs	Fully Convolutional Networks
FCN-PL	FCN patch-wise labeling framework
GLCM	grey-level co-occurrence matrix
GRD	Ground Range Detected
GT	ground truth
HMM	Hidden Markov Model
MLC	Maximum Likelihood Classifier
MLCS-CR	Most Likely Class Sequence with Crop Rotation
MLCS-SL	Most Likely Class Sequence with Crop Sequence Length
MRF	Markov Random Fields
NCC	non-commercial crops
NN	Neural Networks
OA	Overall Accuracy
OBIA	object-based image analysis
PA	Producer's Accuracies
PCA	Principal Component Analysis
ReLU	Rectified Linear Unit
RF	Random Forest classifier
RF-PB	RF pixel-based framework
RS	Remote Sensing

SAE	stacked autoencoder
SAR	Synthetic Aperture Radar
SEM	Stochastic Expectation-Maximization
Seq	Sequence
SeqLen	Sequence Length
SGD	Stochastic Gradient Descent method
SNAP	Sentinel Application Platform
SOM	Self-Organizing Maps
SRTM	Shuttle Radar Topography Mission
SVM	Support Vector Machine
UA	User's Accuracies
UTM	Universal Transverse Mercator
WGS84	World Geodetic System 1984
β	Biases
Θ_z	independent identically distributed random vectors
λ	wavelength
σ^0	normalized backscatter coefficient
$a_t^{(i)}$	output of hidden unit
A	Area
e_i	probability mass functions in the cross-entropy loss function
f	encoder function
g	decoder function
G	antena gain
h	hidden represetation of the input data
I_n	sequence of multitemporal RS data
$J(\theta)$	cost function
k	number of features/kernels in the hidden unit
km^2	square kilometer
L_1,L_2	penalty terms
$L(x, \hat{x})$	reconstruction error
m	patch dimension
N	training set size
p	probability
P_r	received energy
P_t	transmitted energy
R	range
s	stride value
S	sparsity constraint

w	weights
W	set of weights
x	input feature
$\hat{x}$	reconstruction of the input x
x^{l+1}	output feature map of layer l
y	crop class
z	cross-entropy loss

1 INTRODUCTION

1.1 Motivation

Accurate crop recognition can achieve a good estimation of crop productivity, crop health and crop management. Timely and accurate estimation of crop areas can provide valuable information for governmental and private agencies to develop strategies to the agricultural market.

The use of Remote Sensing (RS) images in natural resources mapping has been popular in recent years, becoming the source data for several environmental processes modeling. During the last decade, several Optical and Synthetic Aperture Radar (SAR) satellites were launched with high spatial resolution and low revisit time. Optical remote sensing has been widely used for crop recognition, however, due to cloud cover and shadows the resultant data sets can have missing values. On the other hand, the all-weather, all-time, acquisitions provided by SAR, make multitemporal SAR image sequences a good option for crop type mapping. A key issue in RS image classification relates to capturing contextual information. In crop mapping, the temporal context is even more critical than the spatial one, because most crop types can only be discriminated by exploiting their characteristic temporal behavior.

Neural Networks, Support Vector Machines and Random Forest have been applied to crop classification in pixel-wise approaches [1, 2, 3]. Object-based classification extracting quantitative attributes from segments (mean, area, shape) has been also employed, but this approach relies on segments whose delineation ignores semantics [4, 5]. Nevertheless, the performance of these approaches strongly depends on the features selected for classification. In addition, these approaches generally do not model temporal dependencies in an explicit way.

To cope with the problem of pixel-wise and object-based approaches, Probabilistic Graphical Models (PGMs), such as Markov Random Fields (MRFs) [6] and Conditional Random Fields (CRFs) [7, 8], have successfully exploited both spatial and temporal contexts for the classification of RS imagery. Hidden Markov Model (HMM) has been used too in crop classification based on the time-series analysis of phenological states [9, 10].

Deep Learning (DL) techniques have recently gained broad interest in the RS community. Such techniques contain specific supervised and unsupervised representation-learning algorithms, which learn features from labeled and non-labeled data. In fact, state-of-the-art performance in RS image classification has been achieved by DL techniques, such as Autoencoders (AEs) [11, 12, 13], and Convolutional Neural Networks (CNNs) [14, 15], which integrate both spatial and temporal context in an unsupervised and/or supervised way. Recent works showed that Fully Convolutional Networks (FCNs) outperform CNNs for semantic image segmentation in terms of spatial accuracy and computational load [16].

Other aspect worth mentioning is that most publications about crop recognition from multitemporal RS images rely on datasets from temperate regions, where crop dynamics is comparatively simple because there is usually just a single crop per parcel during the whole season [17, 18, 19, 20, 21, 22]. Crop dynamics in tropical areas is more complex due to multiple agricultural practices such as irrigation, non-tillage, crop rotation and multiple harvests per year, which make the traditional methods not suitable for the aforementioned approaches [23].

The study performed in this work comprehends the evaluation and comparison of three DL algorithms that represent the current state-of-the-art in RS image classification, specifically AE, CNN and FCN. This study is to our knowledge the first attempts to apply FCN approach to crop recognition task. Two different datasets have been used for evaluation, one from a temperate region and a second one from a tropical region. Additionally, a postprocessing algorithm is proposed to incorporate prior knowledge about crop dynamics into the classification model.

1.2 Objectives and contributions

The general objective of this dissertation is to evaluate and compare three different DL algorithms for crop type recognition using multitemporal SAR images sequences. A secondary objective of this work is to include a priori knowledge to model inter-class and intra-class relationships within the SAR images sequence.

The contributions of this work are threefold:

1. Three DL based strategies for crop type classification from multitemporal satellite images.

2. A prior-knowledge based method to model high temporal dynamics typical of agriculture in tropical regions.

3. A performance analysis of the proposed model on datasets representative of different crop dynamics, specifically from a tropical and from a temperate region.

1.3 Organization of the reminder parts

The following parts of this work are structured as follows:

1. Chapter 2 presents an overview of the state-of-the-art in crop mapping from remote sensing image sequences with emphasis on deep learning based approaches.

2. Chapter 3 details the theoretical background of the algorithms tested in this work.

3. Chapter 4 details the methodology followed this study.

4. Chapter 5 presents the experimental protocol, describes the datasets used, the accuracy metrics used to asses the tested classification approaches, the algorithms' set up and the experimental results. The results are discussed in the last part of the chapter.

5. Chapter 6 presents the final conclusions of this work and discusses the future directions of this research.

2
RELATED WORKS

This chapter introduces some important concepts in remote sensing image classification with focus on crop type recognition. In addition, examples of the most relevant works in this field are presented, with emphasis on those related to multitemporal analysis and deep learning techniques.

2.1
Traditional Remote Sensing classification techniques

Traditional classification techniques for Remote Sensing (RS) takes the image pixel as the unit of analysis, with which each pixel is labeled as a single land use/cover class. This technique uses unsupervised (e.g., k-means) or supervised (e.g., maximum likelihood, neural network, support vector machine, random forests) methods to perform pixel-wise classification [24, 25, 26, 27, 28]. This approach uses the spectral variables of the pixels and their transformations (e.g., principal components, vegetation indices, etc.) as input to per-pixel classifiers for unsupervised and supervised classification. However, these methods have a major limitation, because they ignore spatial and temporal context.

Generally, pixel-wise classification algorithms can be divided into two groups: unsupervised and supervised classification. In unsupervised classification, the image is split into a number of classes based on the image values, without the help of prior knowledge [29, 30]. Some unsupervised classification algorithms are k-means and Self-Organizing Maps (SOM) [31, 32]. In contrast, supervised classifiers use representative examples with known class types (i.e., training samples) to learn the relationships among the spectral properties and corresponding labels, then to assign the pixel to the class type according to a mapping learned in the training phase [33]. Traditional supervised classification methods include Maximum Likelihood Classifier (MLC) [1] and K-Nearest Neighbors [34].

Spatio-contextual techniques such as texture extraction [35, 36] have also been used. Texture feature extraction is the quantification of the variability of pixels in a neighborhood and can improve the classification accuracy through smoothing spectral confusion. Statistical methods include mean and standard deviation. Features derived from grey-level co-occurrence matrix (GLCM) [37, 38] are probably the most widely used texture feature extraction strategy used nowadays [39, 5, 40]. Nevertheless, the discriminative ability of these low-level features is limited.

With the launch of more satellites, object-based image classification (OBIA) methods have been developed to partially capture spatial context by classifying segments [41, 4, 42, 43, 44]. The object-based approach generates image objects through image segmentation and then performs the classification on objects rather than pixels. The image objects are formed using spectral, spatial, and textural information.

As the spectral appearance representing the same area changes over time, the temporal context is the relationship of an image site (pixel or segment) with respect to different acquisition times. Therefore, the incorporation of temporal context in the classification model allows for significant improvements in classification of crops and vegetation [45]. Spatio-temporal Markov Random Fields (MRFs) and Conditional Random Fields (CRFs) have successfully integrated both spatial and temporal information [6, 46, 47, 21]. Hidden Markov Model (HMM) has also been used in crop classification based on the time-series analysis of phenological states [9, 48]. These approaches achieve higher accuracies than other methods at the cost of a higher computational effort and more labeled samples for training. They also require expert knowledge about the problem.

2.2
Deep Learning in Remote Sensing

Deep Learning (DL) has become a hotspot in the RS area due to its capability to learn features automatically from data with a better performance than handcrafted features that are manually designed based on domain-specific knowledge. DL architectures are end-to-end approaches that learn high-level feature representations and classify the image sites in an single unified structure. This section focus on a few supervised and unsupervised DL methods that represent the state-of-the-art in DL for RS image classification. A detailed explanation of the DL-based algorithms used on this research is given in the next chapter.

Unsupervised feature learning In crop recognition applications, obtaining reference data with labeled samples is expensive due to the costs and time consuming of field visits and/or visual interpretation by an human expert. In this context, unsupervised techniques might be an attractive alternative.

The unsupervised DL techniques learn features from the input data without knowing the labels. These features are learned from image patches with no knowledge about the semantic. Deep Belief Networks (DBNs) and Autencoders (AEs) are the most widely used unsupervised DL methods in RS area [49, 50]. However, these models can not learn discriminative representations by themselves. This task is usually left for a second stage, where a classifier is trained by using the learned feature representations.

The DBNs has been applied to hyperspectral data classification and achieved competitive accuracies compared to Principal Component Analysis (PCA), and better performance compared to SVM [49, 51]. In recent years, it has also been successfully proposed for land use and land cover, outperforming Support Vector Machine (SVM), Neural Networks (NN), stochastic Expectation-Maximization (SEM) and MRFs [50, 52].

On the other hand, Romero et al. [11] proposed a deep convolutional sparse autoencoder for learning spectral-spatial features relying on a specific sparsity criterion. A set of features is extracted from image patches and classification is then performed to assign a label to each learned feature vector. Romero and co-authors adopt a greedy (layerwise) training strategy so as to simplify the feature learning procedure. Similarly, Tao et al. [53] use sparse stacked autoencoder to learn an effective feature representation from unlabeled data, and then the learned features are fed into a linear SVM for hyperspectral data classification. In [54], Chen et al. used a stacked autoencoders (SAE) to learn deep features of hyperspectral signatures in an unsupervised fashion followed by logistic regression.

Supervised classification The main supervised approaches in DL are variants of Convolutional Neural Networks (CNNs) [55], which have outperformed most algorithms in visual recognition since 2012. While DBNs and AEs train one layer at a time in an unsupervised manner, CNNs learn features and classify the input image or image sites in a single pass.

Recently, CNN have been used for crop type classification. Kussul et al. [14] proposed 1-D and 2-D CNN architectures to explore spectral and spatial features, respectively. They integrate the spatial and temporal contexts in a supervised way, and concluded that ensemble of 1-D and 2-D CNNs outperformed the Random Forest (RF) classifier for crop recognition. Similary, Makantasis et al. [56] exploited a 2-D CNN to encode spectral and spatial information, followed by a multi-layer perceptron to perform classification. In these approaches the trained network computes a descriptor for a given image patch and predicts a single semantic label for the entire patch (independently from the others). This patch label is then assumed to be the label of the central pixel. They then apply the classifier in a sliding-window manner.

Recently, some studies have utilized 3-D CNNs for learning spatio-temporal features from videos [57] or learning spatio-spectral presentations from hyperspectral images [58]. Different from the aforementioned 2-D CNN architecture, where temporal information is exploited by the stacking the mltitemporal data, the 3-D CNN architecture use 3D kernels for the 3D convolution operation to extract spatial and spectral features simultaneously. In [59] authors use a 3-D CNN based method to automatically classify crops from multitemporal RS images [59]. They shown that 3-D CNN is especially suitable in characterizing the dynamics of crop growth and outperformed the 2-D CNN method.

CNNs were originally conceived for image categorization, i.e., to assign a single class label to the whole input image. CNNs can be easily adapted for semantic labeling (also know as semantic segmentation). In this approach, the conventional CNN architecture is applied to the patch centered at each pixel being classified, whereby the single label delivered by the CNN is assigned to the central pixel. More recent approaches predict all labels in an image patch instead of a single label to be assigned to the central pixel. In this scenario the so called Fully Convolutional Network (FCNs) came into play, which outperformed CNNs for semantic labeling/segmentation [60, 61, 16] in terms of thematic and spatial accuracy as well as computational load.

3
THEORETICAL BACKGROUND

This chapter presents the theoretical fundamentals for understanding the frameworks proposed in Chapter 4. Only the principal aspects of each one are given. For further details, the reader is referred to the papers cited in the following sections.

3.1
Synthetic Aperture Radar

Synthetic Aperture Radar (SAR) has been widely used for Earth remote sensing for more than 30 years. It provides high-resolution images independent from daylight, cloud coverage and weather conditions [62, 63]. SAR systems have a side-looking imaging geometry and are based on a pulsed radar. The systems transmits electromagnetic pulses and receives the echoes of the backscattered signal. The amplitude and phase of the backscattered signal depends on the physical and electrical properties of the imaged object. The systems stores the backscatter information corresponding to the cell area on ground scene. The images are recorded parallel to sensor motion (azimuth) and also orthogonal to its motion (range).

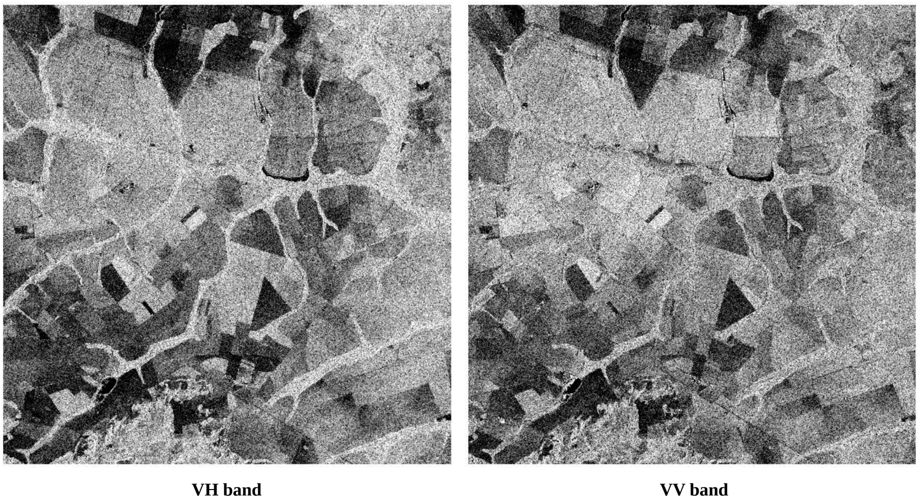
The radar signals are either transmitted with electric field plane parallel (horizontal polarization) or perpendicular (vertical polarization) to the Earth surface. The antenna can transmit and receive in either horizontal (H) or vertical (V) single polarizations (HH or VV) or cross-polarization (HV or VH).

Since SAR systems Earth surface, the backscattered information comes from a portion of a area of a pixel. Thus, backscatter measured from a target area in SAR is usually normalized per unit geometric cell area known as normalized backscatter coefficient σ^0 as shown in the following equation:

$$\sigma^0 = P_r \frac{(4\pi)^3 * R^3}{A * P_t * G^2 * \lambda^2} \tag{3-1}$$

where P_r refers to the received energy, G is the antenna gain, λ is the wavelength, P_t is the transmitted energy, R corresponds to the range and A is the area over which the measurement is made. This is the so-called the SAR equation.

The wavelength affects crop backscatter magnitude because of differences in dielectric constant and relationship between wavelength and leave size. Polarization also influences crop discrimination in SAR images since VV polarized signals interact more with crop structure, HH polarization penetrates crops and captures underlying soil roughness and moisture content and cross-polarized images have also been found to improve crop separability [64, 65, 66]. Usually VV–VH is the preferred dual-polarization for crop classification. Figure 1 presents an example of a SAR image with this type of polarization from an agricultural region in Campo Verde, Brazil.



VH band

VV band

Figure 1: Example of VH and VV bands of a SAR image dual-polarized.

3.2 Random Forest

First proposed by Tin Kam Ho of Bell Labs in 1995 [67, 68], Random Decision Forest is a classifier that consists of growing an ensemble of decision trees and letting them vote for the most popular class. Tin Kam Ho used the random subspace method, where the trees are constructed in randomly selected subspaces and established that this technique can achieve high accuracy for both training and unseen data.

The Random Forest algorithm (RF) in the current form was introduced by Leo Beimman in 2001 [69], who defined it as a classifier consisting of a collection of tree-structured classifiers $\{R(x, \Theta_z), z = 1, ...\}$, where $\{\Theta_z\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x [69]. Beirman combines Bagging [70] with a random variable selection at each node [71] in one the most effective methods in machine learning, working very well for a wide range of problems. The mathematical fundamentals behind the algorithm can be found in [69].

3.3 Autoencoders

An Autoencoder (AE) [72, 73] is a Neural Network (NN) [74] that is trained with backpropagation and mini-batch gradient descent method to reproduce at its output the pattern presented at its input. The basic architecture of an AE involves an encoder function f whose outcome h is used as a representation of the input data x, and a decoder function g, that maps back from representation to the input space (see Figure 2). Since they are restricted to only reproduce the input at the output, it often learns useful properties of the data.

The AE automatically learns features from unlabeled data. The learning process searches the parameter space for the set of values that minimizes the reconstruction error, a measure of the average discrepancy between the input and the corresponding output of the AE. Once the parameters have been learned, the encoder is used to generate for any input the corresponding internal representation, which is expected to be more discriminative than the original one.

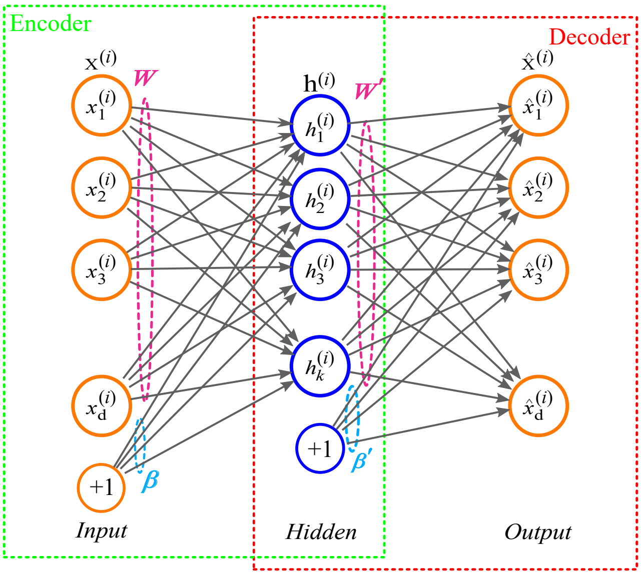


Figure 2: Autoencoder architecture, example case for input data $x^{(i)}$.

Encoder and Decoder functions: Given a set of unlabeled training samples $x = \{x^{(1)}, x^{(2)}, ..., x^{(I)}\}$ where $x^{(i)} \in \mathbb{R}^d$, the encoder function allows the straightforward computation of a new feature representation $h^{(i)} \in \mathbb{R}^k$ (see Figure 2(green dotted block)). Each sample vector $x^{(i)}$ is processed by applying a linear mapping followed by a nonlinear activation function:

$$h^{(i)} = f(Wx^{(i)} + \beta) \tag{3-2}$$

where $W \in \mathbb{R}^{k \times d}$ is a weight matrix for k feature, $\beta \in \mathbb{R}^{k}$ is a bias vector and $f(.)$ is the encoder activation function (typically the element-wise sigmoid or hyperbolic tangent non-linearity). The decoder function maps from feature space $h^{(i)}$ back into input space (see Figure 2(red dotted block)), producing the reconstruction $x^{\widehat{(i)}}$ as follows:

$$\hat{x}^{(i)} = g(W^{'} h^{(i)} + \beta^{'}) \tag{3-3}$$

where $W^{'}$ is usually constrained to be equal to W^{T}, $\beta^{'} \in \mathbb{R}^{d}$ and $g(.)$ is the decoder activation function. The set of parameters $\theta = (W, W^{'}, \beta, \beta^{'})$ are learned simultaneously attempting to get the lowest possible reconstruction error $L(x, \hat{x})$, typically mean square error $\frac{1}{2}\|x - \hat{x}\|^{2}$. The cost function of autoencoder is shown below:

$$J(\theta) = \frac{1}{N} \sum_{i} L(x^{(i)}, g(f(x^{(i)}))) \tag{3-4}$$

where N denotes the training set size and L the reconstruction error for each training sample.

Regularization and Sparsity: To capture useful properties it is important to prevent the autoencoder from learning the identity function. This is achieved through regularized terms. One particular form of regularization consists in constraining the dimension of the latent representation h being smaller than that of input x, i.e., $k < d$ (undercomplete). However, when $k > d$ (overcomplete), is still possible extract meaningful features by enforcing a sparsity constraint on the hidden units.

Sparsity in the representation can be achieved by penalizing the neurons to be inactive most of the time. A neuron is considering “inactive” if its output value is close to 0 (for sigmoid) or -1 (for hyperbolic tangent) [75]. The cost function comprising sparsity and regularization takes the form

$$\begin{aligned} J(\theta) = \frac{1}{N} \sum_{i} L(x^{(i)}, g(f(x^{(i)}))) + \alpha \sum_{l}^{n_l} \sum_{j}^{m_{l-1}} \sum_{r}^{m_l} (W_{rj}^{l})^{2} \\ + \lambda \sum_{t}^{k} S(a_{t}^{(i)}) \end{aligned} \tag{3-5}$$

where the first term corresponds to the reconstruction error for each training sample, the second term is the regularization term (the weight-decay term), which favors the low magnitude of the weight vectors, and helps to prevent overfitting. Here, n_l is the number of layers, m_l is the number of neurons in layer l, $(W_{rj}^{l})^{2}$ represents the connection between the $j-th$ neuron in layer

$l-1$ and the $r-th$ neuron in layer l. The parameter α controls the relative importance of the two terms [76]. In last term, $S(.)$ is the sparsity constraint that penalizes the hidden units for being far from zero and $a_t^{(i)}$ represents the output of $t-th$ hidden unit of $i-th$ input sample. There are many forms of penalty terms, such as the L_2 or L_1 norm, Student-t and Kullback-Liebler divergence. In practice, one common choice for the sparsity cost unit $S(.)$ corresponds to the L_1 penalty $S(a_t) = \|a_t\|_1$ [77, 78].

3.4 Convolutional Networks

In this section, Convolutional Neural Networks (CNNs) and Fully Convolutional Neural Networks (FCNs) are described. Both neural networks architectures are oriented to handling data with some spatial and/or temporal relationship (e.g. images, videos, speech processing, etc.). In image analysis, each layer of data in a Convolutional Networks (convnet) is a three-dimensional array where the first layer is the image, with pixel size $m \times n$, and d channels. The region of the input space that are path-connected to a particular unit of the network is called the receptive field. These type of networks are also trained with backpropagation and stochastic gradient descent method (SGD).

3.4.1 CNNs principles

CNNs [79, 80, 55] rely on local linear operations followed by a non-linear transformation, forming a sequential hierarchy of processing layers. The main objective is assigning a single class label to an entire image/scene. A general description of the CNN forward pass is given by

$$\begin{aligned} x^1 \rightarrow u_1(x^1; W^1) \rightarrow x^2 \rightarrow u_2(x^2; W^2) \rightarrow ... \\ \rightarrow x^L \rightarrow u_L(x^L; W^L) \rightarrow z \end{aligned} \tag{3-6}$$

where x^1 is the input (an order 3 tensor image in this work) and it goes through the processing in the first layer u_1 all the way to the final layer u_L. The functions $u_l(.)$ are usually linear functions, subsequently passed through nonlinearities, while w^l are the parameters at the $l-th$ layer. The input of each layer u_l is the output of the preceding layer x^l. One additional layer z is added, which delivers the final network result.

In the following, are described the five basic building blocks of the CNN architectures that01 1his work deals with: convolution, nonlinear activation, spatial pooling, classification, and the loss function. More details can be found in [81].

Convolutional layer: Convolutional layer consists of several filters using on local receptive fields on the features maps of the previous layer or input, in order to extract interesting features. The input to a the layer is a 3-dimensional array, which is convolved with a set of k trainable filters. Each filter is a set of weights W of size $w \times w \times d$ (usually squared), which maps the d-dimensional feature map at the input to a k-dimensional feature map. Given an input tensor of dimension $m \times n \times d$ and a kernel window size of $w \times w \times d$ centered at location i, j, the convolution response for the r-th filter can be expressed as

$$x^{l+1}_{i,j,r} = \sum_{c=1}^{d} \sum_{q=0}^{w-1} \sum_{v=0}^{w-1} (W^{l}_{v,q,c,r} \times x^{l}_{v,q,c})_{i,j} + \beta \tag{3-7}$$

where β is a learneable bias term and l is the layer number. Eq. (3-7) is repeated for all k filters and for any spatial location i, j. The final output of the convolutional layer is $\left(\frac{(m-w+2zer)}{s} + 1\right) \times \left(\frac{(n-w+2zer)}{s} + 1\right) \times k$-dimensional feature map, where s is the stride and zer is the number of *zero padding*, if applied and k is the number of filters. Stride indicates that the convolution is performed once every s pixels both horizontally and vertically, and *zero padding* is a padding trick used to control the size after convolution (e.g. to ensure output with the same height and width of input image). Elements of the padded rows and columns are usually set to 0. One of the many benefits of the convolution layer over the fully connected counterpart is that all spatial locations share the same filters and each filter is applied by sliding it over the input which greatly reduces the number of parameters to be learned. Figure 3 illustrates this procedure.

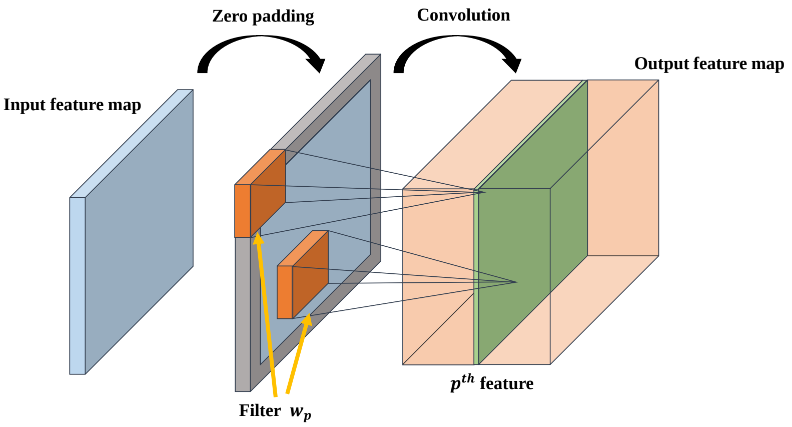


Figure 3: Principle of a convolutional layer.

Nonlinearity layer: Many transformations have been used in the neural network community to produce nonlinearity. However, during the training procedure such nonlinearities cause the magnitude of the gradient to reduce significantly, and after several layers, the gradient will be close to 0 (the *vanishing gradient* problem). The use of nonsaturated activation function improve the gradient propagation and accelerate the learning speed. In this context, the most commonly chosen one is the Rectified Linear Unit (ReLU) [82], formulated as $ReLU(x) = max(0, x)$, which replaces all negative values in the feature map (in the l-th layer) by zero.

Pooling layer: Most convolutional networks involve down sampling layers. The objective is twofold: to provide some shift invariance and to summarize spatial information while preserving discrimination, both at a low computational cost. A commonly used pooling strategy is the so-called max pooling. It consists of mapping each non-overlapping subregion (typically 2×2) to a single number, the maximum within the group. Given a small window size of $H \times H$ centered at location i, j, the maximum value in H_{ij} is given by $x_{ij}^{l+1} = \max_{a \in H_{ij}} x_a^l$.

Fully connected, classification and loss: A fully connected layer is commonly used at the end of a CNN model and implies that every neuron in the previous layer l is connected to every neuron on the next layer $l + 1$. It can be seen as a Multi Layer Perceptron [75] that uses a Multinomial Logistic Regression in the penultimate layer (other classifiers like SVM can also be used). For an image classification problem with C classes, the scores (class-conditional probabilities) are usually given by the softmax activation function

$$p(y = i|x^L) = \frac{\exp(x^L \times W_i^L)}{\sum_c^C \exp(x^L \times W_c^L)} \tag{3-8}$$

where $p(y = i|x^L)$ is a C-dimensional output vector whose i-th entry encodes the posterior probability of class i. The filters W_c^L can be interpreted as the weight vector of the classifier.

The last layer is a loss layer, that defines the objective function, which is minimized in the training procedure in a supervised way using back-propagation. In a classification problem, the cross-entropy loss is often used to measure the discrepancy between the CNN prediction y and the target e, and its formulated as:

$$z[p(y = i|x^L), e] = -\sum_i p(y = i|x^L) \log (e_i) \tag{3-9}$$

where $[e_1, ..., e_C]$ is a categorical C-dimensional vector, whereby $e_i = 1$ if i is the target class, and $e_i = 0$ otherwise. Once both e and p are probability mass functions, the cross-entropy loss measures the distance between them.

Figure 4 shows a regular CNN architecture composed by a convolutional layer, followed by three downsampling stages, a fully connected layer and a softmax layer.

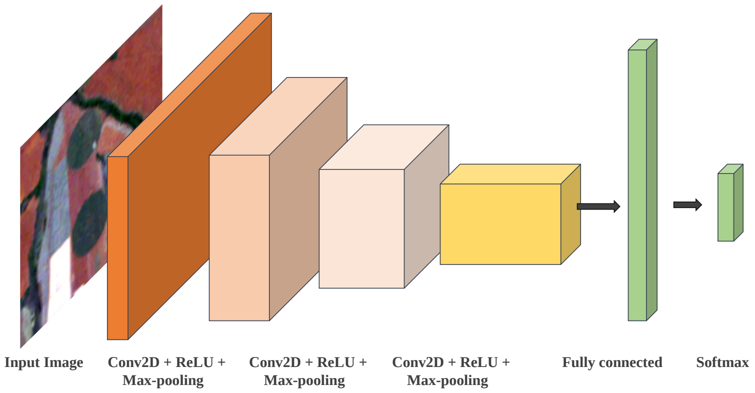

Input Image
Conv2D + ReLU + Max-pooling
Conv2D + ReLU + Max-pooling
Conv2D + ReLU + Max-pooling
Fully connected
Softmax

Figure 4: Convolutional Neural Networks architecture.

3.4.2 FCNs principles

The FCNs [83] were introduced in the literature as an extension of CNNs specifically designed for semantic segmentation, i.e., assigning a semantic label to every pixel in the input image. Basically, the FCN replaces the fully connected layer of a standard CNN by upsampling layers to recover the spatial resolution of the input at the output layer (see Figure 5).

FCN performs an end-to-end learning in two phases: first downsamples (consisting of successive convolution, activation and pooling layers) and then upsamples (deconvolution) it again, allowing to predict dense output labels for an arbitrary-sized input. Often, the output of an upsampling layer is connected with the output of the corresponding layer (in terms of size) in the downsampling stage. These connections, so called skip connections, aim to recover fine details that might have vanished due to successive downsamplings. Several layers of deconvolution and activation functions can learn a nonlinear upsampling.

In FCNs, both learning and inference are performed for the whole image at once in order to get a probability map of semantic labels, without loss in terms of spatial resolution.

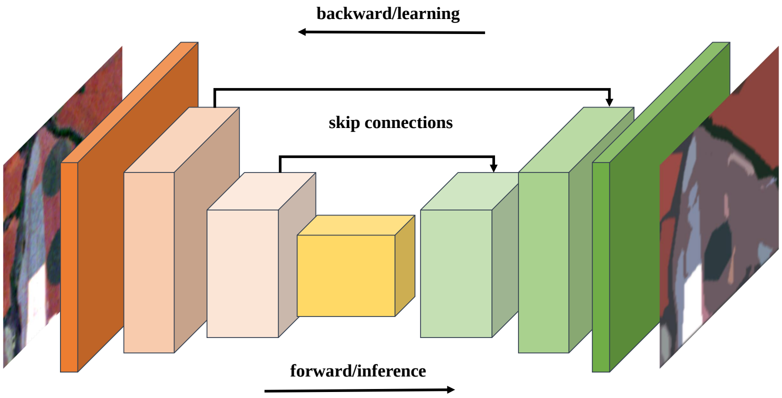


Figure 5: Fully convolutional architecture for semantic labeling.

From fully connected to upsampling: The fully connected layer can be interpreted as a convolutional layer with filters that cover their entire input regions. For example, a fully connected layer with $k = 1028$ that is connected with an input volume of size $16 \times 16 \times 256$, can be viewed as a convolutional layer with dimension of $16 \times 16 \times 1028$ performed with stride 1 and no zero padding. That is, the filter size is setting to be the size of the input volume, under this conditions only a single depth column fits across the input volume and the output will be $1 \times 1 \times 1028$. Thus, converting the fully connected layers to convolutional layers (with a specific filter size), the network can take an input of any size and output a classification map.

Deconvolutional Layer: As the data goes through convolutional and max-pooling layers, the size of the feature maps becomes smaller. Thus, this process has to be inverted somehow to obtain at the output a semantic label matrix with the same resolution as the input data. To this purpose FCNs use so called deconvolutional layers [84]. In practice, deconvolution is implemented as the transposed convolution operator and can be seen as a convolutional layer with backward and forward passes inverted [85, 81].

The transpose convolution relocates the activations of the previous layer in the upsampled grid and performs a convolution for end-to-end learning by backpropagation from the pixelwise loss. Figure 6 shows the transpose of convolving a 3×3 kernel over a 5×5 input padded with a 1×1 border of zeros using 2×2 strides, which it is equivalent to convolving a 3×3 kernel over a 3×3 input (with one zero inserted between inputs) padded with a 1×1 border of zeros using unit strides.

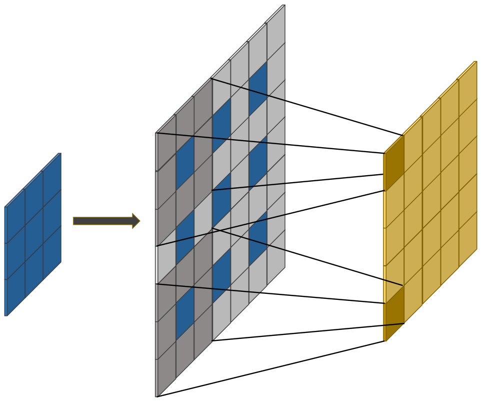


Figure 6: Principle of the transpose convolution (deconvolution). Adapted from [81].

Classification and loss layer: As in CNNs the classification layer uses Multi Layer Perceptron (or any other classifier). In this case, the final layer in the network is a $1 \times 1 \times C$ convolution (where C is the number of classes) typically followed by a softmax non-linearity to provide the per class distribution at each pixel. Finally, the model is trained by minimizing the pixel-wise cross-entropy loss. The loss is not computed over a single prediction as for the CNN, but over the grid of spatial predictions.

If the loss function is a sum over the spatial dimensions of the output layer, its gradient will be a sum over the gradients of each of its spatial components. Thus SGD computed on the whole image will be the same as SGD taking all of the output layer receptive fields as a mini-batch. In this scenario10 both feedforward and backpropagation are much more efficient when computed over an entire image instead of independently patches [83].

3.4.3 Regularization layers

There are several ways of controlling the capacity of neural networks to prevent overfitting. Some of them are described in the following.

Batch Normalization: Batch Normalization (BN) [86] properly initializes neural networks by forcing the set of activations throughout a network to have zero mean and unit variance for each training mini-batch. This normalization address the problem of internal covariate shift [87] (when the distribution of each layer's inputs changes during training). BN makes network training less sensitive to layer initialization and improves convergence.

Dropout: This technique has been proposed to reduce data overfitting in the feature-learning procedure [88, 89]. During training, each neuron is kept active with some probability p (dropout rate hyperparameter) or it is set to zero otherwise. This can be seen as training a different model at every epoch, similar to bagging. At inference time all activations are used but with scaled-down weights. Mathematically, this approximates ensemble averaging, i.e., the geometric mean. The ensemble predictions formed by voting tend to generalize better than the individual predictions. Dropout could slow down training but is particularly effective when the number of parameters to learn is large (e.g., in the fully connected layers).

Weight decay: Weight decay is a L_2 regularizer and the most common form of regularization. It can be implemented by adding a penalty term to weight updates during backpropagation. For every weight w in the network is added the term $\frac{1}{2}\lambda w^2$ to the objective, where λ is the hyperparameter, which controls the penalization.

3.4.4 DenseNets

DenseNets [90] is a CNN where earlier feature maps are concatenated with the last convolution output forming a data cube, which is then submitted to a convolution operation. Thus, the output of the l-th layer is defined as $x_l = bn^l(\text{conct}(x^{l-1}, x^{l-2}, ..., x^1))$, where conct(...) represents the concatenation operation and bn is a composite function of three consecutive operations: BN, followed by ReLU and a 3×3 convolution. The purpose is to reuse the information contained of the previous feature maps (with the same resolution, no pooling is performed) generated up to that network layer. Let's suppose that the depth of the feature map at the output of each intermediate layer l has G-dimensional feature maps, where G is called growth rate parameter. Thus, at the hidden layer L the DensNet will have $L \times G$ feature maps (see Figure 7).

After L layers a transition layer is introduced to reduce the spatial dimensionality of the feature maps. Such transformation is composed of a 1×1 convolution followed by a 2×2 pooling layer. A deep DenseNet architecture is composed of one or more dense connections and downsamplig layers, followed by a fully connected layer and a softmax layer (see Figure 7).

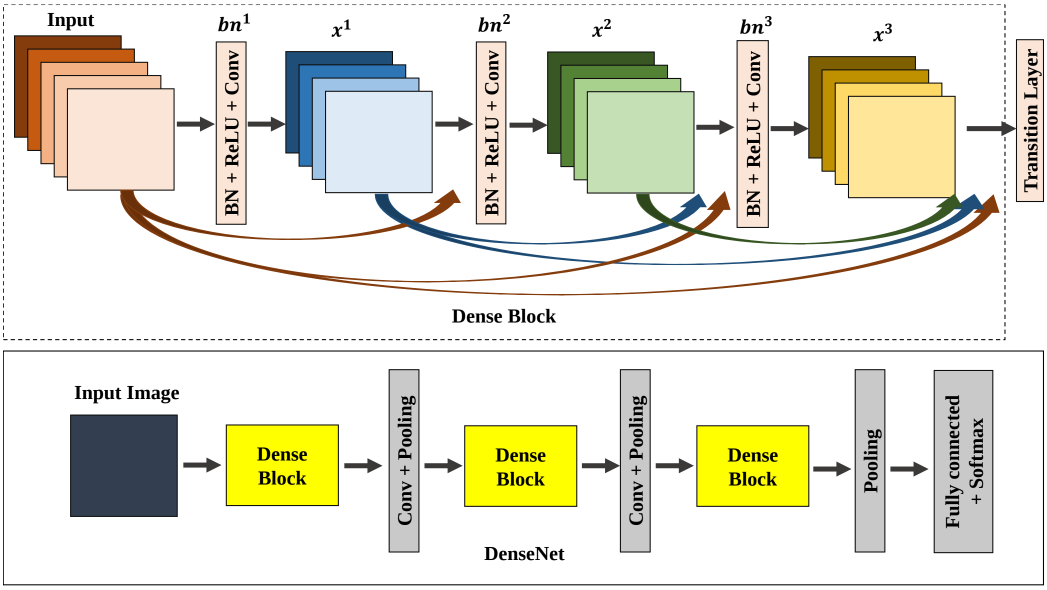


Figure 7: A 4-layer dense block with a growth rate of G = 4 (dotted block) and a deep DenseNet with three dense blocks (solid block).

4
METHODOLOGY

This chapter describes the methodology followed in the present dissertation to accomplish the three goals of this work. Firstly, the single class and multiclass approaches are described, followed by a detailed explanation of each classification framework. Finally, the prior-knowledge post-processing algorithm is described.

4.1
Single class and Multiclass analysis

Single class: Given a sequence $(I_1, I_2, ..., I_T)$ of multitemporal RS data acquired at epochs 1 to T respectively, the pair data-reference is defined as follows:

$$\begin{aligned} H = &((I_{11}, I_{12}, ..., I_{1T}), t_1)... \\ &((I_{N1}, I_{N2}, ..., I_{NT}), t_N) \end{aligned} \tag{4-1}$$

where I_{ij} is the data from site i in epoch j, t_i is the ground truth of $i - th$ image site, T is the sequence length and N is the number of image sites. We assume that there is a single crop per image site, i.e., a unique reference map for the whole sequence. Notice that in this approach a crop label is assigned to a pixel even during the prepared soil and after harvest stages, although no crop actually exists at these stages.

Multiclass: The multiclass approach relaxes the hypothesis that there is a single crop per image site throughout the sequence. For a given sequence of multitemporal RS data from time 1 to time T, the pair data-reference is defined as follows

$$\begin{aligned} H = &((I_{11}, I_{12}, ..., I_{1T}), (t_{11}, t_{12}, ..., t_{1T}))... \\ &((I_{N1}, I_{N2}, ..., I_{NT}), (t_{N1}, t_{N2}, ..., t_{NT})) \end{aligned} \tag{4-2}$$

where t_{ij} corresponds to the ground truth in epoch i at site j. The multiclass approach considers also the soil class that corresponds to the period prior to seeding and after harvest.

4.2 Crop type classification frameworks

The general framework (Figure 8) is fed by a sequence of multitemporal images $(I_1, ..., I_T)$ covering a certain geographical area. Each image I_t corresponds to a different date within a selected period of time and is a tensor $I_i \in \mathbb{R}^{k \times l \times d}$, where k and l refers to spatial coordinates, and d is the number of bands per pixel. The crop type classification models to be tested in this work are illustrated in Figure 8 (dotted blocks). Three DL methods are used to evaluate three different approaches: 1) unsupervised feature learning using AEs for pixel-based classification (AE-PB), 2) CNNs for patch classification with spatially independent predictions (CNN-PC) and 3) FCNs for full patch labeling with structured predictions (FCN-PL). A RF classifier used for pixel-based classification (RF-PB) was chosen as baseline.

For all frameworks, the temporal context was exploited using the feature stacking technique. Specifically, spatially correspondent pixels in all epochs are concatenated along the 3rd dimension; the resulting tensor serves as input to the classification in all epochs of the sequence.

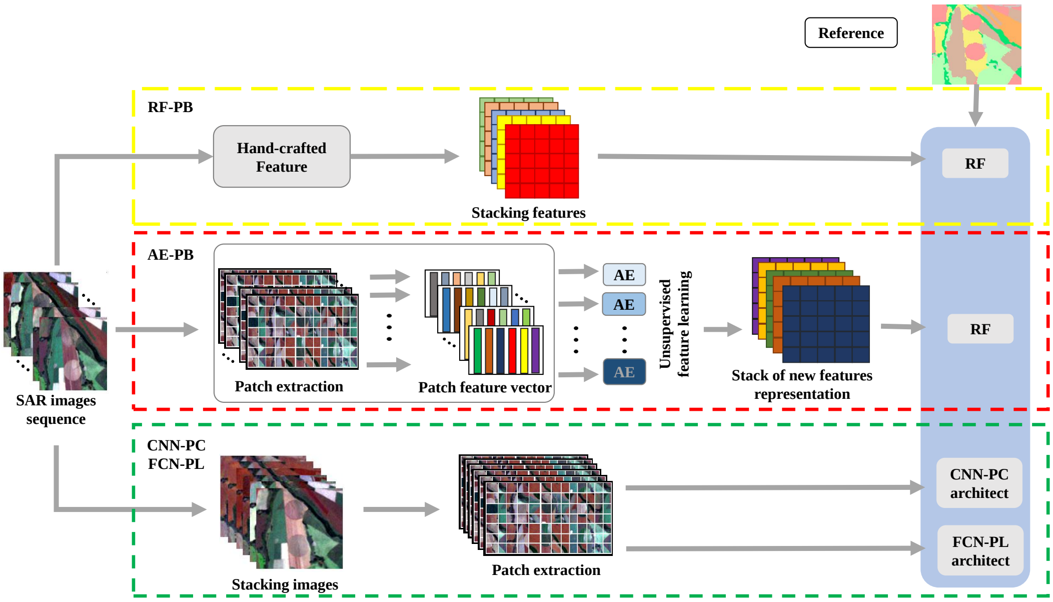


Figure 8: General crop type recognition framework: RF-PB (yellow dotted block), AE-PB (red dotted block), CNN-PC and FCN-PL (green dotted block).

4.2.1 Random Forest pixel-based framework

The RF-PB approach is highlighted in Figure 8 (yellow dotted block). It consists of applying a RF classifier to the feature vector of each pixel. Recall that there is a single feature space, where all pixels in the sequence are represented. Pixels at the same spatial coordinates share the same representation in this space for all epochs. The procedure consists of three main steps: 1) extract the texture features for each image in the sequence separately, 2) stack these features over all images and 3) apply a Random Forest classifier to map this feature space to a crop type in a given epoch. The classification result is arranged in two dimensional matrix, in order to produce a label image.

4.2.2 Autoencoder pixel-based framework

Contrary to RF-PB the AE-PB exploits the spatial context by taking as representation of each pixel the feature of all pixels in a patch of size $m \times m$ (the size of window) around that [91]. These are straightened into a one-dimensional vector to be fed into an AE network. As depicted in Figure 8 (red dotted block), the framework consists of four main steps: 1) patch-based feature extraction, 2) unsupervised feature learning, 3) stacking new feature representation, and 4) classification.

Patch extraction: The patch-wise descriptor of each image site was built by arranging in a vector the pixel-wise descriptors within its $m \times m$ neighborhood. Thus, each pixel is represented by a $m \times m \times d$ low level feature vector, where d is the number of image bands.

Unsupervised feature learning: Part of the resulting patches are extracted for each image using random sampling. This set of patches is fed into an AE of one hidden layer architecture (see Figure 2 in chapter 3) for an unsupervised learning of the feature extractor h (i.e., encoder). It is worth pointing out that an AE is trained for each set of patches, i.e., an AE is trained for each image in the sequence separately.

Stacking feature representation: After the unsupervised feature learning, the learned feature extractor h of the corresponding AE is used to get the new feature representation for each image. Specifically, a new feature representation in $\mathbb{R}^k$ is computed for each $m \times m \times d$ patch of the input image, i.e., a k-dimensional feature is extracted from each location i, j. Next, the concatenation of those representations over the whole sequence is taken as the final pixel descriptor.

Classification: A RF classifier is applied to map points in this feature space to a crop type in a given epoch. Next, each pixel is spatially arranged to obtain a crop-map at the same resolution of the input image.

4.2.3 CNN patch-classification framework

As in [16] the CNN-PC captures the spatial context of a certain pixel by a CNN that takes as input an image patch (extracted from the original image) and predicts a single label, which is assigned to the central pixel. As illustrated in Figure 8 (green dotted block), the framework consists of three principal steps: 1) image stacking, 2) patch extraction, and 3) classification.

Image stacking: The idea is stacking the pixel wise features, in this case the raw data, over the whole sequence, similar to the procedure adopted in the RF-PB approach. So, there is a single feature space of $k \times l$ dimensions with $d \times b$ bands, which represents all images in that sequence, where b is the number of images.

Patch extraction: The stacking feature space is then cropped in densely overlapping images patches with a sliding window technique with a 1-pixel step, so as to preserve the spatial resolution of predictions. Each image patch has a dimension of $m \times m \times d \times b$, where $d \times b$ is the depth of the patch. These images patches must accomplish the condition of m«k and m«l.

Classification: In this step, a CNN architecture is used to perform both training and inference. This architecture takes as input an image patch and computes the patch class probabilities, which will be assigned to the pixel at the center of the patch. Next, each query is spatially concatenated to obtain a crop-map at the same resolution of the input image.

Architecture details As shown in Figure 9 the architecture consists of a convolutional layer followed by a pooling layer. Then, they are followed by a fully connected layer and an activation layer, which assigns class scores to the central pixel of the input patch.

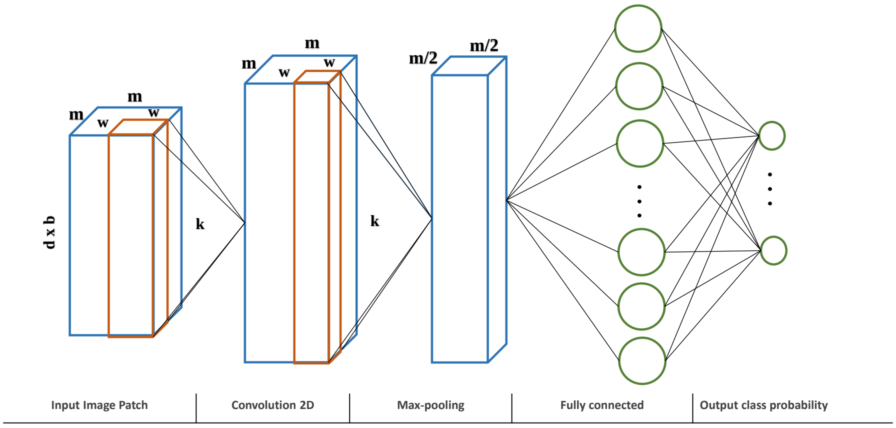


Figure 9: CNN-PC architecture.

Limitations: The patch-classification framework presents some limitations for crop mapping task:

- Patches close to each other in space are likely to represent the same class. However, the network is designed to predict a single label from a patch, independently on the labels of its surrounding. Often, this leads to a salt-and-pepper-like result.
- A post-processing stage is often applied to performs structured prediction on the probabilities given by the classifier, e.g., Conditional Random Fields [92].
- To obtain a prediction map at the same resolution of the input image, it is necessary to predict a considerable amount of patches corresponding to the total number of pixels. This can be extremely inefficient for large-scale image recognition.

4.2.4 FCN patch-wise labeling framework

The FCN-PL framework considers spatial structures by training an FCN architecture to predict all labels in the patch instead of a single label. Similar to CNN-PC, the framework consists of three main stages: 1) image stacking, 2) patch extraction, and 3) classification.

Image stacking: As in the previous framework, given a SAR image sequence, pixel-wise features are stacked over all images to create a single feature space which represents the entire sequence.

Patch extraction: The stacking feature space is decomposed into a series non-overlapping images patches. In this order, was applied a sliding window technique with a m-pixel step to get adjacent images patches. Each image patch has a dimension of $m \times m \times d \times b$, where $d \times b$ is the depth of the patch.

Classification: In the classification step, the image patches are submitted to the FCN-PL network to obtain a class score map at the same resolution as the input patches. This way, the time can be roughly reduced at inference compared with CNN-PC approach. After mosaicking the predicted patches, it is obtained a crop-map at the same resolution of the input image.

Architecture details: As illustrated in Figure 10, the FCN-PL consist of a downsampling (blue dotted block) and a upsampling (red dotted block) path. Each downsampling step is implemented as a dense block followed by a convolution and a max-pooling layer. The Dense block is referred to the concatenation of the new feature maps created at a given resolution (see chapter 3 subsection 3.4.4). The dense block architecture used in the downsampling path is composed of two convolutional steps, whereby the input of a dense block is concatenated with its output. From then on, two upsampling layers restore the original resolution and a final convolution layer computes the class scores.

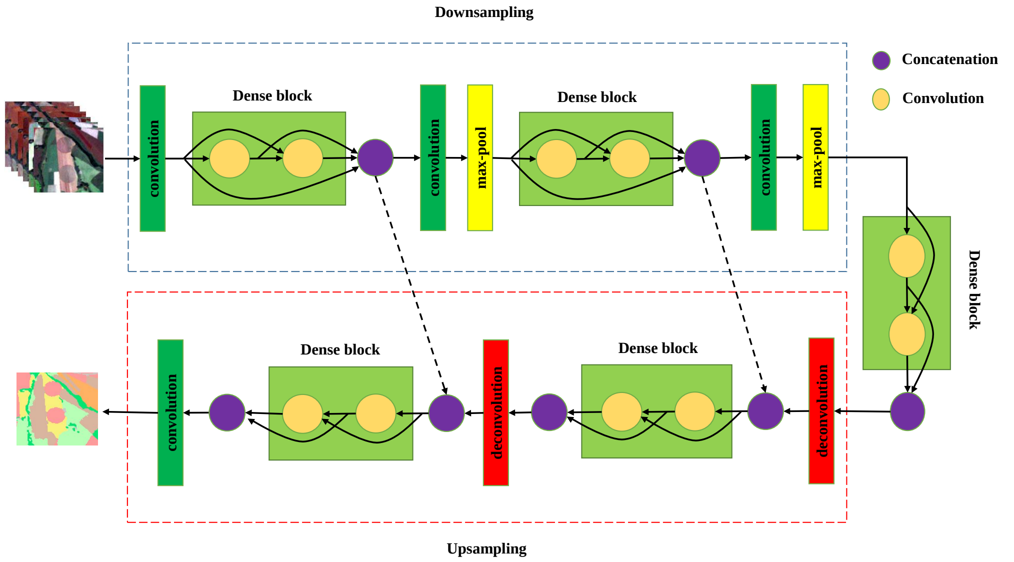


Figure 10: FCN-PL architecture.

Following [93], each upsampling stage is designed with a deconvolution followed by a dense block. Each of these dense blocks comprises two convolutional steps, but unlike to the downsampling ones, their input is not concatenated with their output. It is also employed skip connections that concatenate feature maps from the upsampling stage with the corresponding feature maps produced in the downsampling path. The upsamplings are learned so that the deconvolutions reconstruct learn spatial and geometrical arrangements of activations at larger scale but acting locally, conditioned on the receptive field of the previous layers.

The output of this architecture can be interpret as structured, since every predicted label is learned to be interdependent with its neighbors, conditioned on the receptive field of the previous layers.

Pros and Cons: As compared to a CNN-PC approach, the FCN-PL exhibits the following pros and cons:

- The feature reuse by dense block and skip connections enforces connectivity between downsampling and upsampling stages.
- Learning class-relationships and co-occurrences represented in the input patch (i.e., the prediction is locally structured) can minimize salt-and-pepper effect ensuring that sites are labeled in regard to neighboring labels and data.
- The image patch can be feedforwarded to the trained network obtaining a dense score map which implies a lower execution time at inference.
- Need of a more sophisticated balanced strategy to deal with highly unbalanced data bases.

4.3 Incorporating prior knowledge

In this section we propose two post-processing approaches that enforce prior knowledge about the dynamics of the different crop types in the target site.

MLCS with crop rotation: The first approach, called most likely class sequence with crop rotation (MLCS-CR), is illustrated in Figure 11. It is inspired in a earlier work on crop recognition that rely on directed graphical models to represent crop dynamics [10]. In this model y_i stands for the crop class in epoch i, for $i = \{1, ..., T\}$, and x denotes the observed feature vector.

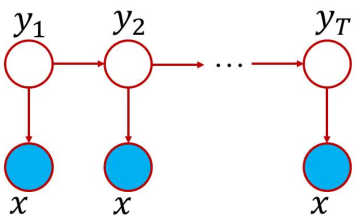


Figure 11: MLCS-CR.

Recall that in all classification methods introduced in the previous section the observation related to an image site is the same for every epoch, i.e., the stacked feature vector over all sequence. So, the probability that a particular sequence of crop classes $(y_1, y_2, ..., y_T)$ occurs in a given image site over T epochs given the observation x is given by

$$\begin{aligned} p(y_1, y_2, ..., y_T|x) = & p(y_1|x)p(y_2|y_1)p(y_2|x)... \\ & p(y_{T-1}|x)p(y_{T-1}|y_T)p(y_T|x) \end{aligned} \tag{4-3}$$

where $p(y_i|x)$ is the crop posterior probability in epoch i given the observation x, and $p(y_{i+1}|y_i)$ is the crop transition probability from epoch i to epoch $i+1$.

Assuming that the terms on the right-hand side of Equation $(4-3)$ can be properly estimated, it is possible to compute the probability of every sequence of crop classes. We propose to take as the final result the sequence $(\hat{y}_1, \hat{y}_2, ..., \hat{y}_T)$ corresponding to the highest probability, formally

$$(\hat{y}_1, \hat{y}_2, ..., \hat{y}_T) = \arg\max_{y_i}[p(y_1|x)p(y_2|y_1)...p(y_{T-1}|y_T)p(y_T|x)] \tag{4-4}$$

The posterior probabilities $p(y_i|x)$ can be calculated by any of the classification methods introduced in previous section. As for the transition probabilities $p(y_{i+1}|y_i)$ we rely on prior knowledge.

Human experts on crop dynamics in the target site may inform the crop class transitions that are less probable to occur in each pair of consecutive epochs. For instance, under a proper/high temporal resolution, a change from *maize* to *soybean* must necessarily go first through the class *soil*. So, the transition *maize* → *soybean* can not occur in consecutive epochs. In view of equation (4-4), every sequence containing at least one improbable transition will have zero probability, and will therefore be discarded as a potential solution.

Estimating the probabilities of possible transitions is not an easy task. Even experienced experts may find it difficult to choose a real value between

0 and 1, which represents the probability of each possible transition with adequate accuracy. Given this difficulty, we propose an alternative solution, which consists in taking for each class sequence its maximum probability value. In other words, we assume that the probability of all admissible class transitions in the sequence being evaluated are equal to 1. Thus, the solution is given by

$$(\hat{y}_1, \hat{y}_2, ..., \hat{y}_T) = \arg\max_{y_i} \max_{p(y_i|y_{i-1})} [p(y_1|x)p(y_2|y_1)...p(y_{T-1}|y_T)p(y_T|x)] \quad (4\text{-}5)$$

By assuming that every possible transition has probability equal to 1, and impossible class transitions have zero probability, Eq.(4-5) can be further simplified as

$$(\hat{y}_1, \hat{y}_2, ..., \hat{y}_T) = \arg\max_{y_i} [p(y_1|x)...p(y_T|x)] \quad (4\text{-}6)$$

whereby only sequences with no impossible class transitions are considered.

Figure 12 describes this solution for $C = 4$ classes and $T = 4$ epochs. Columns correspond to epochs, and rows to crop classes. So, the nodes represent a crop class in each epoch. Arcs identify possible class transitions in adjacent epochs. From this graph we can infer the set of sequences to be evaluated in the computation of Eq.(4-6).

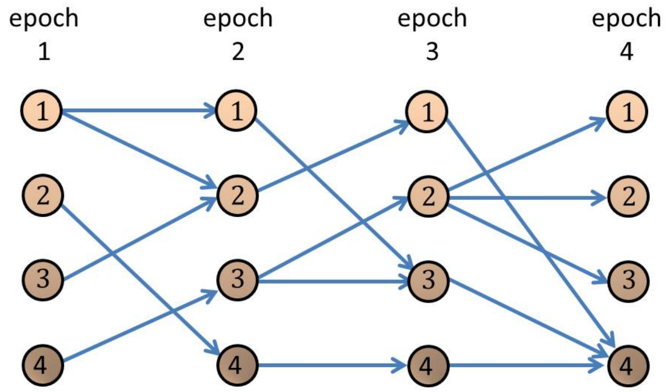


Figure 12: Example of possible transitions.

The elimination of sequences inconsistent with the prior knowledge has two benefits: the number of sequences to be evaluated is reduced, and the accuracy increases.

Notice that the algorithm is not applied in a new train-inference procedure. It just combines the classification results obtained by the methods introduced in the previous section with prior expert knowledge to eliminate solutions that involve impossible class transitions.

The graph of Figure 12 can be represented by a set of $T-1$ $C \times C$ binary transition matrices. Each matrix refers to a pair of adjacent epochs. The rows correspond to the crop type in the earlier epoch and columns to the crop type in the later epoch. Its element ij will be 1 ("true") or 0 ("false") if the transition $y_i \rightarrow y_j$ is possible or impossible, respectively.

MLCS crop sequence length: The MLCS-CR post processing improves classification accuracy, as will be demonstrated experimentally in the next chapter. However, it still admits some wrong solutions that could be avoided with the use of prior knowledge. Figure 13 shows an example of a wrong solution not detected by MLCS-CR. It shows the crop evolution along six epochs. Let's assume that the only two possible class sequences in the target site are the ones shown in the upper part of Figure 13. Both sequences consist of class C occurring in three consecutive epochs but shifted in time in relation to one other.

Notice that transition $S \rightarrow C$ between epochs 2 and 3 and the transition $C \rightarrow C$ between epochs 3 and 4 would be permitted by the MLCS-CR approach. This would allow solutions other than the ones enrolled as admissible. Starting either from class C or class S, the sequences can choose a wrong path after epoch 3, allowing a sequence consisting of class C along epochs 1 to 5, as well as a sequence with class C only in epoch 3 preceded and followed by class S. Both sequences are inconsistent with the prior knowledge. In fact, the MLCS-CR approach only enforced consistency for 2 consecutive epochs.

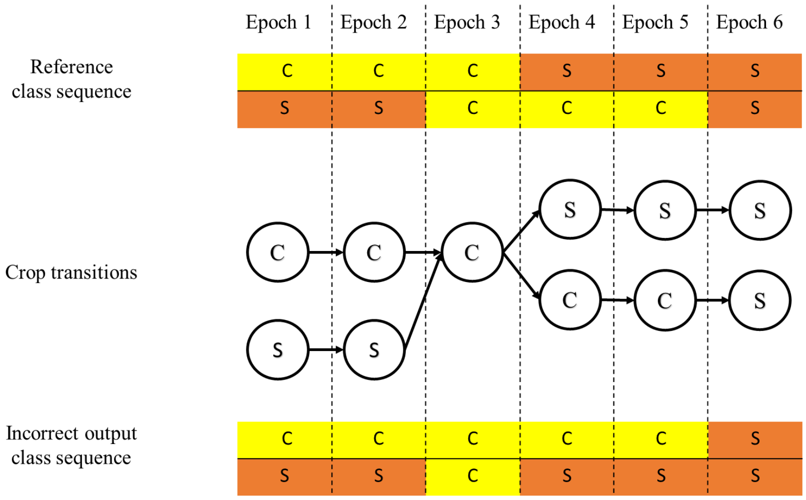


Figure 13: MLCS-CR transition matrix approach. C and S stands for two different crop types.

In order to avoid this kind of errors we propose the MLCS crop sequence length (MLCS-SL) approach (see Figure 14), which takes into account the knowledge about the crops' sequence lengths. Given the same reference class sequences, each crop type was split up into a set of classes depending on their sequence lengths (see Figure 14 top). With this refinement, the transition matrix between epoch 3 and epoch 4 only accept as possible transitions $C3 \rightarrow S1$ or $C1 \rightarrow C2$ (see Figure 14 middle) circumventing the aforementioned incorrect paths. After applying the algorithm, each set of classes are grouped back to the original crop type, obtaining this way the output class sequence (see Figure 14 bottom). It is worthing point out that the posterior probabilities of $C1$,$C2$,$C3$ is the probability of class C and the posterior probabilities of $S1$,$S2$,$S3$ is the probability of class S.

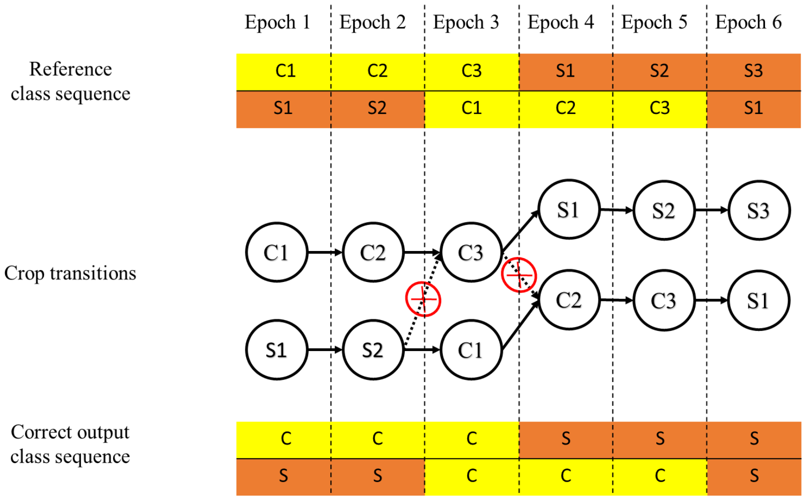


Figure 14: MLCS-SL transition matrix approach.

5
EXPERIMENTAL ANALYSIS

In this chapter we describe the experiments to evaluate the multitemporal crop classification frameworks introduced in Chapter 4. Section 5.1 describes the target sites and the training-testing sample selection strategy. Section 5.2 describes the metrics used for accuracy assessment. Section 5.3 presents the implementation details and parameters' setup. Finally, Section 5.4 describes the experimental protocols and discusses the results.

5.1
Study area

Since agricultural areas can vary strongly in different regions, two highly differentiated study areas have been used to assess the performance of the proposed methods: Hanover in Germany and Campo Verde in Brazil. The weather conditions in these sites are very different, which leads to distinct crop dynamics.

Hanover dataset: The first site used in our experiments is in the surroundings of the city of Hanover, in Northern Germany (52°22'N, 9°43'E) (see Figure 15). The average annual precipitation is 656 mm and the average annual temperature is 8.9 °C. Class found in this area are *barley*, *rye*, *wheat*, *canola*, *grassland*, *maize*, *potato* and *sugar beets*. Typical of temperate regions, in this dataset each parcel belongs to the same class over the whole season. The site covers an extension of 1728 km^2.

The dataset consists of 24 dual polarized (VV & VH) Sentinel-1 images acquired in the Interferometric Wide Swath Level-1 Mode with a 250 km swath at 5 meters by 20 meters spatial resolution, captured from October 2014 to September 2015 (see Table 1). The images were downloaded from the Sentinels Scientific Data Hub in Level-1 Ground Range Detected (GRD) and preprocessed using the Sentinel Application Platform (SNAP) with Sentinel-1 Toolbox.

First, precise orbit information was applied, which is available days after the generation of the product. The orbit file provides accurate satellite position and based on this information the orbit state vectors in the metadata

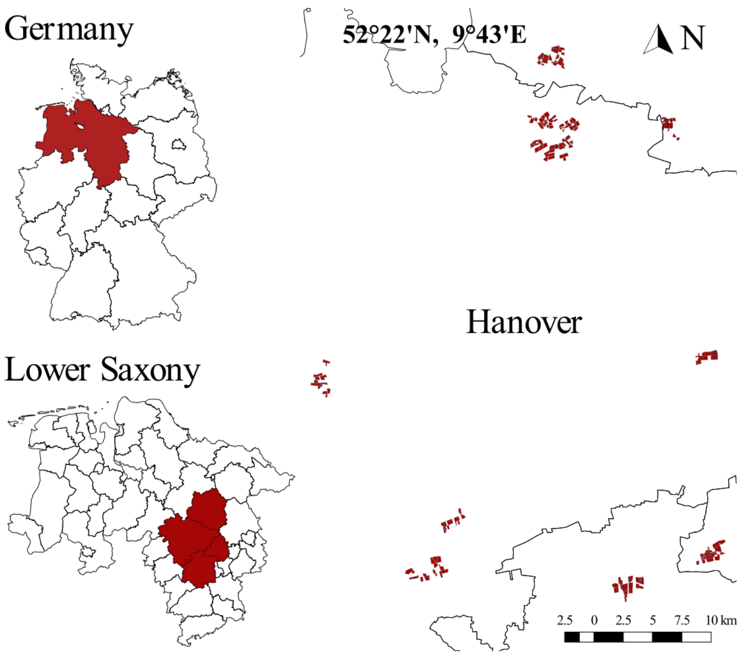


Figure 15: Location of study area Hanover, Germany.

were updated. Second, the products were geometrically corrected using a Range Doppler terrain correction with a Digital Elevation Model (DEM) from Shuttle Radar Topography Mission (SRTM) and radiometrically calibrated to a backscatter coefficient σ^0. This step also involved georeferencing to the World Geodetic System 1984 (WGS84) system and resampling of the DEM and the images to 10 m resolution. Next, the VV and VH bands in a linear scale were converted to dB. Two images per date at different times were necessary to cover the entire region, and as last step the data of these dates was merged and clipped according to the area of interest.

The reference in situ data contains 256 fields (∼120000 pixels) [94]. Two disjoint sets of polygons were randomly selected, one for training and the other for testing, using stratified random sampling from Quantum GIS. To ensure that there were no pixels from the same field in the training and the testing sets, the selection was performed at the polygon level. The experiments were conducted taken approximately 50% for training and 50% for testing. Table 2 illustrates the field distribution for all crops in the study area.

In Hanover, crop year stretches from October to October with one planting period. The crop life cycle extends from 4 months (*barley* and *potatoes*) to 10 months (*rye,wheat* and *canola*). Figure 16 shows the crop calendar.

Table 1: Sentinel-1 acquisition dates over Hanover.

Year	**Month**	**Date**
2014	October	13, 22
	November	15,27
	December	09, 21
2015	January	14, 29
	February	10, 22
	March	15, 27
	April	11, 23
	May	14, 26
	June	10, 22
	July	13, 25
	August	18, 30
	September	14, 26

Table 2: Distribution of training and test data from Hanover.

	Train	testing
	# fields	# fields
Maize	24	23
Potato	8	8
Canola	8	7
Sugar beet	23	23
Barley	4	3
Wheat	19	19
Rye	11	11
Grassland	17	16

Campo Verde dataset: The second site is situated in Campo Verde, a municipality in the state of Mato Grosso in the central west region of Brazil (15°32’48"S, 55°10’08"W) (see Figure 17) [23]. The average annual precipitation is 1726 mm and the average annual temperature is 22.3 °C. The main crops found in this area are *soybean*, *maize* and *cotton*. Some minor crops, such as *beans* and *sorghum*, are also present. In the class non-commercial crops (*NCC*) we joined *millet*, *brachiaria* and *crotalaria*. Other classes present in the dataset are *pasture*, *eucalyptus*, uncultivated *soil* (e.g., bare soil, soil with weeds, soil

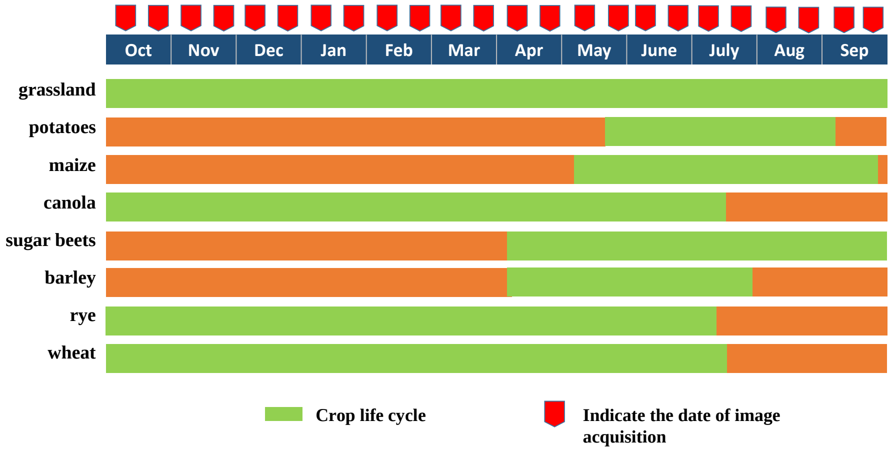


Figure 16: Crop calendar for Hanover. Adapted from [94].

with crop residues), *turfgrass* and *cerrado* (Brazilian savanna). Figure 18 shows the class occurrences per month in the dataset. The area used in our analysis has an extension of 4,782 km^2.

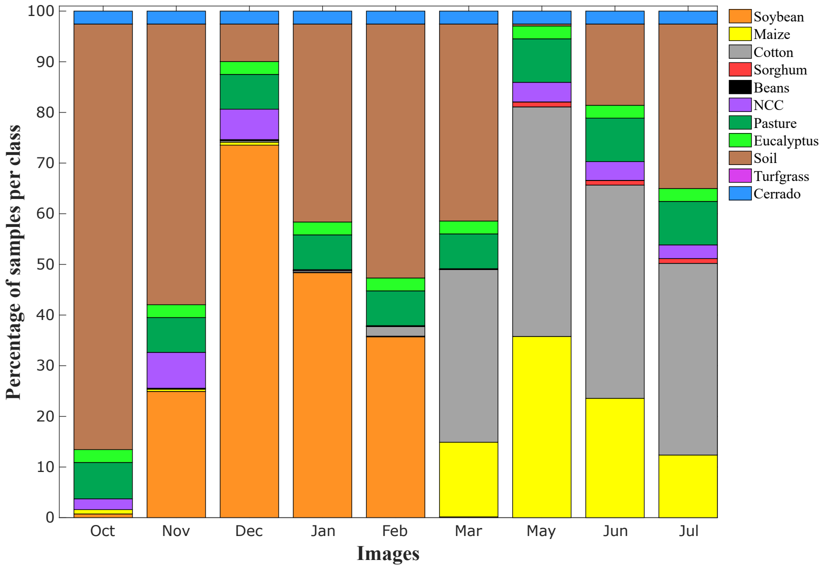


Figure 18: Class occurrences per month in Campo Verde.

The available database consists of a set of 14 pre-processed SAR Sentinel-1 and 15 Landsat-8/OLI mosaic images [23]. Only the SAR images were used in this work. The 14 dual polarized (VV & VH) images were acquired in the Interferometric Wide Swath Level-1 Mode with a 250 km swath at 5 meters by 20 meters spatial resolution, captured from October 2015 to July 2016 (see Table 3). The images were acquired from the Sentinels Scientific Data Hub in Level-1 GRD and preprocessed using SNAP with Sentinel-1 Toolbox.

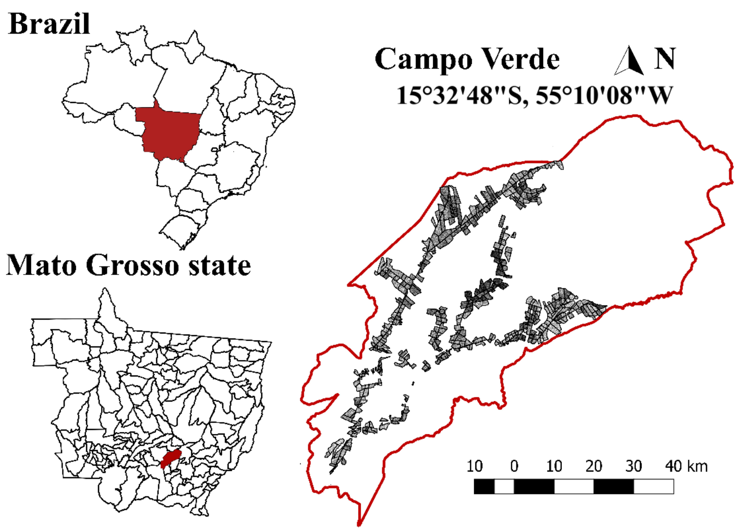


Figure 17: Location of study area in Campo Verde municipality, Mato Grosso state, Brazil. Taken from [95].

First, the images were radiometrically calibrated to a backscatter coefficient σ^0. Second, the images were geometrically corrected using a Range Doppler terrain correction with a Digital Elevation Model (DEM). In this step the images were georeferenced to the WGS84 system and resampled for 10 m resolution. Next, the images were converted to dB, co-registered using a RapidEye mosaic (5 m spatial resolution) and georeferenced to UTM projection Zone 21S and Datum WGS84.

Table 3: Sentinel-1 Acquisition dates over Campo Verde.

Year	**Month**	**Date**
2014	October	29
	November	10,22
	December	04, 16
2015	January	21
	February	14
	March	09, 21
	May	08, 20
	June	13
	July	07, 21

The available reference data (ground truth)[23] comprises a total of 513 fields (~6 millions pixels). In order to select training and testing sets some polygons was then split up using Quantum GIS getting a total of 608 fields.

For a random selection of the sets we applied the same procedure adopted for the Hanover dataset. For all ground truths the procedure ensured that all classes were represented in both training and testing set. Table 4 illustrates the global fields distribution for training/testing set.

Table 4: Distribution of training and testing data for Campo Verde.

	Train	testing
Number fields	312	296

The crop year stretches from late August to July with two seeding periods. The main crops are annual crops; their phenological cycles can extend to 3 or 4 months (*soybeans* and *maize*) and to 4 up to 6 months (*cotton*). Figure 19 shows the crop calendar for principal crops: *soybeans*, *cotton* and *maize*. The types of crop rotation present in the dataset are *soybeans-maize*, *soybeans-cotton*, *soybeans-sorghum*, *soybeans-pasture*, *soybeans-beans*, *soybeans-non-commercial crops* (NCC), *beans-cotton*, *maize-cotton*, *NCC-cotton*. In addition, some areas were cultivated with soybean in the first period and later used as *pasture* (*soybean-pasture* rotation) [23].

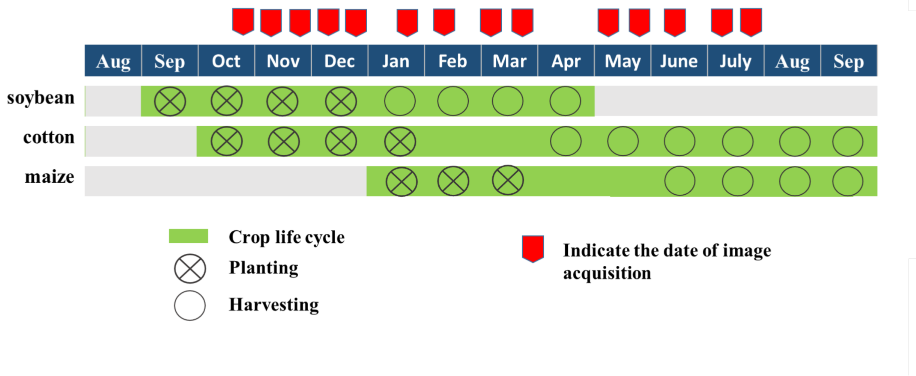


Figure 19: Crop calendar for principal crops in Campo Verde.

5.2 Accuracy Assessment

The performance of tested methods were expressed in terms of overall accuracy, class accuracy, average class accuracy, F1 score per class, class-averaged F1 score and kappa index. A description of each metric calculated in this work is detailed bellow (more details can be found in [96]).

The Confusion matrix records correctly and incorrectly recognized examples for each class. Table 5 presents the matrix in mathematical terms. The true classes are noted C_i $(1 \leq i \leq h)$, whereas the estimated classes, as defined by the considered classifier,are noted $\hat{C}_j$ $(1 \leq j \leq h)$.

Table 5: Mathematical example of confusion matrix.

	C_1	C_2	...	C_h
$\hat{C}_1$	cm_{11}	cm_{12}	...	cm_{1h}
$\hat{C}_2$	cm_{21}	cm_{22}	...	cm_{2h}
...	...	...	...	...
$\hat{C}_h$	cm_{h1}	cm_{h2}	...	cm_{hh}

The terms cm_{ij} $(1 \leq i, j \leq h)$ denote the number of samples recognized as category i in the classification map, when they actually belong to category j in the reference data. Consequently, diagonal terms $(i = j)$ correspond to correctly classified instances and the off-diagonal $(i \neq j)$ terms represent incorrectly classified ones. The proportion is calculated by $p_{ij} = cm_{ij}/cm$, when cm is the total number of samples. The sums of the confusion matrix elements over row i and column j are noted cm_{i+} and cm_{+j},respectively.

The Overall Accuracy (OA) represents the proportion of correctly classified pixels with respect to reference data. Thus, the most used empirical measure, OA is a global measure accuracy, so it is depending of larger classes. This measure ranges from 0 (perfect misclassification) to 1 (perfect classification) and can be stated as the trace of the confusion matrix divided by the total number cm of classified instances:

$$OA = \frac{\sum_{i=1}^{h} cm_{ij}}{cm} \tag{5-1}$$

The producer's accuracies value represents the probability that a certain class on the reference is correctly classified. The PA for the class C_j and can be computed by:

$$PA_{C_j} = \frac{cm_{jj}}{cm_{+}j} \tag{5-2}$$

The user's accuracies represents the probability that a pixel classified into a given class actually represents that class on the reference. The UA for the class C_i and can be computed by:

$$UA_{C_i} = \frac{cm_{ii}}{cm_{i}+} \tag{5-3}$$

F1 score (F1) is the harmonic mean of UA and PA. F1 is usually more useful than accuracy, especially if uneven class distribution. The F1 measure for the class C_i can be computed by:

$$F1_{C_i} = 2 \times \frac{PA_{C_i} \times UA_{C_i}}{PA_{C_i} + UA_{C_i}} \tag{5-4}$$

The Average Class Accuracy (AA) is the proportion of correctly classified pixels per class. So, it is insensitive to the number of samples of the reference classes.

Kappa index it is a measure of the magnitude of agreement between the predicted and reference class relationship. The calculation is based on the difference between the actual agreement compared to the agreement that would be expected by chance. It has value between 0 and 1, when a value of 0 point to a total random classification and a value of 1 pint to a perfect agreement between the reference and classification pixels. It is an approach to measure agreement over and above chance. The Kappa index is also a global measure accuracy.

5.3 Parameters setup

The hyperparameters of each tested method were tuned based on experiments. In order to balance the number of training samples for all classes, we replicated samples of less abundant classes. For CNN-PC it was taken into account the class of the central pixel, whereas for FCN-PL the balancing was done patch-wise. For the Hanover dataset we selected $30,000$ samples per class and for Campo Verde $130,000$ samples per class.

For the DL frameworks batch sizes were selected experimentally and fixed to 128 for AE-PB and CNN-PC, and 32 for the FCN-PL. For the optimization we used AdaGrad [97] with a learning rate of 0.01 and AdaDelta [98] with a learning rate of 1.0, for Campo Verde and Hanover datasets respectively.

We developed a program to configure the experiments in a simple and user friendly way. It was implemented using the Sklearn module of Python for RF-PB experiments and Keras (with TensorFlow backend) for the DL based techniques. The models were trained on a desktop workstation with an Intel Core i7-4790 3.6GHz CPU, 32GB of main memory and an NVIDIA GeForce GTX1080 graphics processor with 12GB of memory. All experiments run under Linux (Ubunutu 16.04 distribution).

RF implementation details: For the RF-PB approach hand-crafted features were used. Following [20], we computed texture features (correlation, homogeneity, mean and variance) from Gray-Level Co-occurrence Matrices (GLCM) in four directions (0, 45, 90 and 135 degrees) using 7×7 windows per polarization (VV and VH in this case). We tested 3 window sizes (3, 5, 7 pixels) and decided to use 7×7 regions. This approach yielded 32 dimensional feature vectors for each pixel in each epoch. After some tests the RF classifier was foxed to 250 random trees with a maximum depth equal to 25.

AE-PB implementation details: Patches from the original images were selected as input features. After some tests we decided to take patches with 7 × 7 pixels as input to the AE, thus, the final vector comprehend a $7 \times 7 \times 2bands = 98$ features. The patches were flattened into vectors, which were standardized zero mean and unit variance. It was employed 100 neurons at the hidden layer with $tanh$ activation function and an L_1 norm fixed to 0.001. The feature maps obtained this way were the inputs to a random forest classifier setup as described before.

CNN-PC implementation details: Patches from the original images were selected as input features. After having tested square patches of width/height equal to 5, 7, 9 and 16, we decided to work with 7 × 7 patches. The downsampling stage were built with 3×3 convolution using ReLU as activation function, followed by a 2 × 2 max pooling. The convolution stride was fixed to 1 pixel. For the convolution spatial padding was applied in order to preserve the spatial dimension after the convolution. In the end, a fully connected with dropout of 20% and a softmax layer to perform classification were added. The input patches were standardized by subtracting the mean. Table 6 summarizes all layers for inputs consisting of a stack of 14 images (i.e., 28 channels) and 11 classes.

Training was carried out by SGD applied to patches randomly selected from training set. At each iteration, the patches were grouped in mini-batches to estimate the gradient of the loss function with respect to the network's parameters.

Table 6: Architecture details of CNN-PC model.

CNN-PC Architecture	
Layers	Output shape
Input	7 × 7 × 28
3 × 3 Conv	7 × 7 × 100
Max Pooling	3 × 3 × 100
Fully connected	200
Softmax, 11 classes	
Total params: 207.711	
Trainable params: 207.711	
Non-trainable params: 0	

FCN-PL implementation details: Patches from the original images were selected as input features. In order to exploit the advantage of the FCN architecture, large patches were selected, specifically of size 16, 32, 64, and 128 pixels. Experiments showed that 32 × 32 pixel patches delivered the best results. Dense block layers were composed of BN, followed by ReLU, a 3 × 3 convolution (with stride 1, i.e., no resolution loss) and dropout with 20% rate. The growth rate of the dense block was set to $G = 16$. An initial convolution was applied with zero padding. The downsampling stages were built with BN, followed by ReLU, a 1 × 1 convolution, a dropout with 20% rate and a 2 × 2 max pooling. Upsampling was carried out by applying a 3 × 3 transposed convolution with stride 2. Table 7 summarizes all layers (DB stands for the downsampling dense blocks and DB' stands for the upsampling dense block) for a stack of 14 images and 11 classes. As in CNN-PC, training was carried out by SGD.

Table 7: Architecture details of FCN-PL model.

FCN-PL Architecture		
Layers	Output shape	Feature maps
Input	32×32	28
3×3 Conv	32×32	48
DB (2 layers)	32×32	$48 + 16 + 16 = 80$
Downsampling	16×16	80
DB (2 layers)	16×16	$80 + 16 + 16 = 112$
Downsampling	8×8	112
DB' (2 layers)	8×8	$16 + 16 = 32$
Upsampling	16×16	$32 + 112 = 144$
DB' (2 layers)	16×16	$16 + 16 = 32$
Upsampling	32×32	$32 + 80 = 112$
1×1 Conv	32×32	11
Softmax, 11 classes		
Total params: 174.672		
Trainable params: 172.624		
Non-trainable params: 2.048		

5.4
Experiments

This section describes the experimental procedures and the results for each method/protocol tested in this work.

5.4.1
Single class evaluation protocol

In the Campo Verde dataset there are more than one crop per parcel along the sequence. Therefore, the single class protocol can not be applied to the entire sequence. In order to test the single class approach in Campo Verde dataset we split the sequence into two sub-sequences, within the single class condition holds, as shown in Figure 20. The first one is from October 2015 to February 2016 (hereafter called Seq-1) containing mostly soybean. The second sub-sequence extends from March to July (hereafter called Seq-2), with *cotton* and *maize* being the major crops. These sequences contain mainly uncultivated soil and a unique crop type per parcel, and can therefore be used to test the single class approach. In spite of the occurrence of soil in some epochs, in these experiments we assigned a single crop label to each pixel along the whole sub-sequence in order to identify the crop type cultivate in that planting period. In both sub-sequences, we had to discard a small number of parcels, where this condition does not hold.

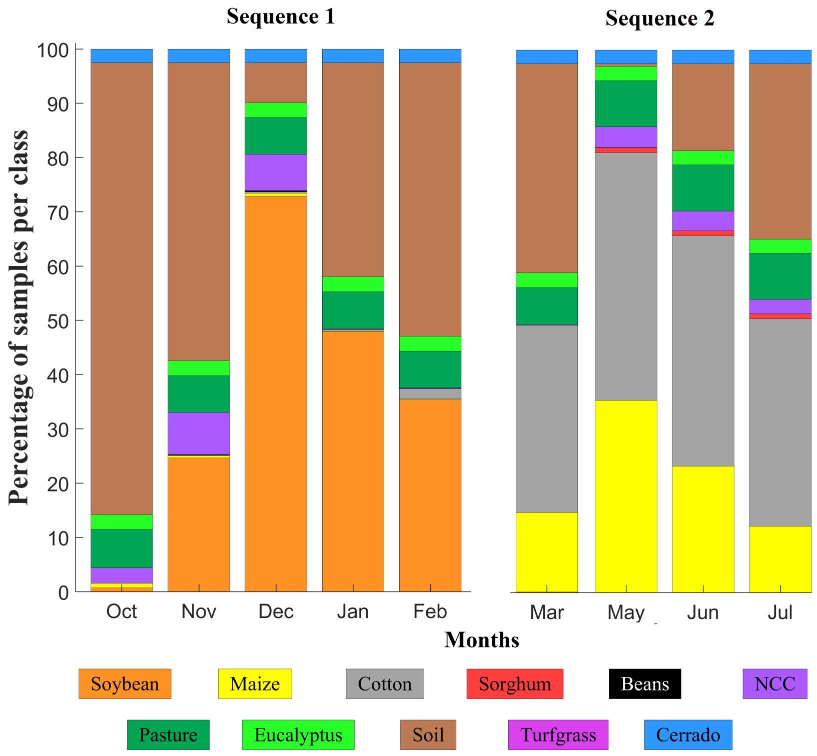


Figure 20: Campo Verde sub-sequences for single class analysis.

All accuracy values reported in the next subsections refer to the last epoch of the whole sequence. Notice that many sequences ending in a given epoch can be built by appending images of earlier epochs.

Results for Hanover: Figures 21 and 22 summarize the measured performances in terms of average F1 and AA, respectively. The bars within each group correspond to RF-PB, AE-PB, CNN-PC and FCN-PL frameworks, respectively. The leftmost bar of a graph refers to a monotemporal classification, i.e., for a sequence comprising a single image. The next bars to the right indicate the performance measured upon sequences of increasing lengths, formed by adding earlier images consecutively. As more images were considered, average F1 improved from 25% to 82% (RF-PB), from 34% to 92% (AE-PB), from 31% to 90% (CNN-PC) and from 23% to 92% (FCN-PL). In terms of AA a similar behavior was observed: from 27% to 80% (RF-PB), from 36% to 90% (AE-PB), from 31% to 90% (CNN-PC) and from 23% to 93% (FCN-PL). Table 8 also shows that the OA and Kappa index increased as prior images were added to the sequence.

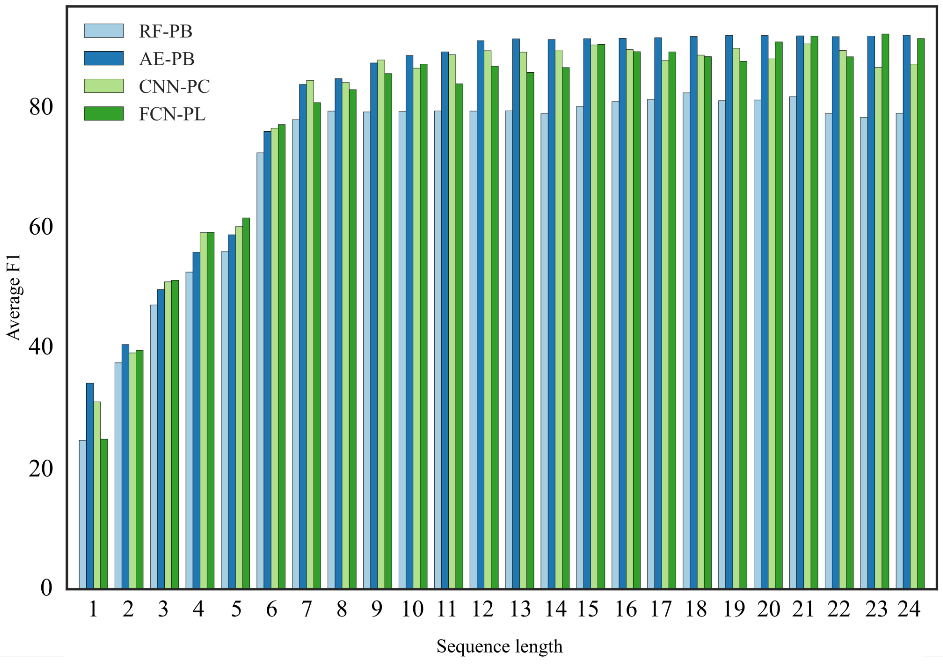


Figure 21: Average F1 for different sequences, taking the last image in the database and adding earlier images. Hanover dataset.

For up to 11 images per sequence (which includes images from May to September) performance improved considerably as more images were added to the sequence. For sequences between 12 and 24 images, the performance remained nearly constant for all frameworks. This behavior can be explained

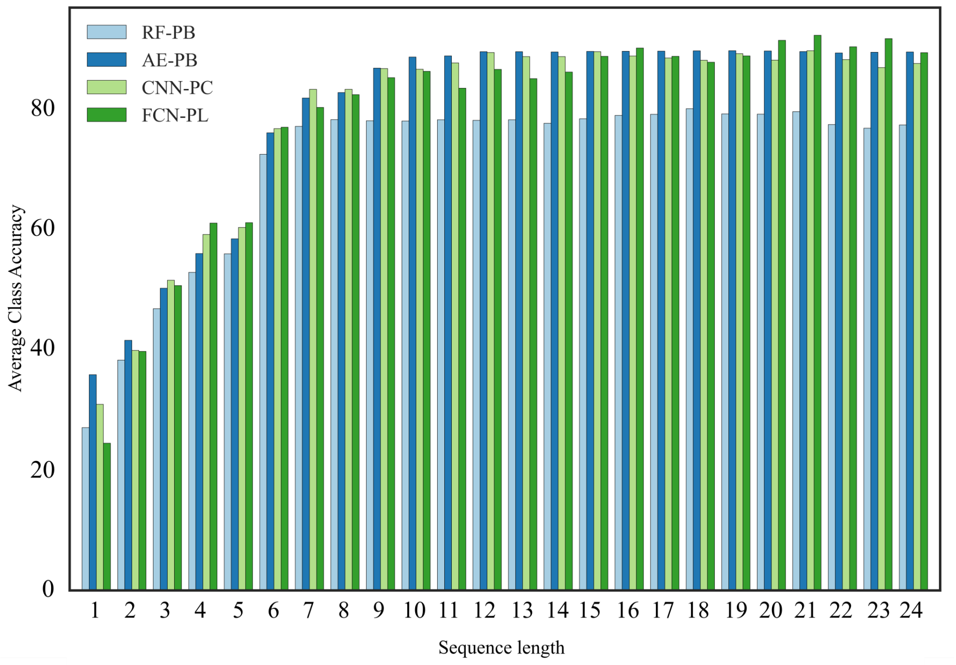


Figure 22: AA for different sequences, taking the last image in the database and adding earlier images. Hanover dataset.

by the crop calendar shown in Figure 16. Up to May, many crops are still in their prepared soil stages and cannot be easily discriminated. From May on the crops sprout and assume characteristic appearances in the SAR data, which allows better discrimination.

Table 8: Hanover: OA and Kappa index for different sequence lengths.

Sequence Length	RF-PB		AE-PB		CNN-PC		FCN-PL	
	OA	Kappa	OA	Kappa	OA	Kappa	OA	Kappa
1	34.11	21.65	**46.46**	**35.29**	41.33	29.13	33.17	19.24
3	58.17	50.01	**63.66**	**56.04**	61.00	53.38	59.38	51.47
5	69.46	63.27	**74.56**	**69.13**	72.54	67.10	68.63	62.67
7	86.79	83.97	91.77	90.00	**92.05**	**90.38**	89.38	87.15
9	86.96	84.18	92.62	91.08	**93.59**	**92.25**	91.98	90.30
11	86.97	84.19	93.64	92.32	**94.00**	**92.75**	90.70	88.75
13	87.41	84.72	**94.57**	**93.43**	93.19	91.79	92.27	90.64
15	87.85	85.23	**94.67**	**93.55**	94.22	93.01	93.84	92.54
17	88.08	85.50	**94.73**	**93.62**	93.19	91.79	93.08	91.66
19	87.53	84.83	**94.81**	**93.72**	93.66	92.34	92.42	90.85
21	87.65	84.98	**94.73**	**93.62**	94.12	92.89	93.56	92.24
24	87.50	84.76	**94.66**	**93.53**	92.08	90.44	93.99	92.74

The DL based techniques outperformed the RF-PB approach in almost all experiments, being AE-PB generally the best performing one. In terms of F1 and class accuracy (Acc) all classes reached higher values for DL frameworks.

Table 9 show the results for *barley* and *rye*, which are cereals crops and differentiate between them is a more challenging task.

Table 9: Hanover: F1 and Accuracy (Acc) per class for different sequence lengths. Crops type: *barley* and *rye*. SeqLen stands for the sequence length.

		RF-PB		**AE-PB**		**CNN-PC**		**FCN-PL**	
SeqLen		**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**
barley	7	14.95	9.83	26.21	15.82	**29.58**	**23.73**	24.88	20.12
	12	20.32	13.21	**67.77**	**61.52**	54.61	63.06	45.97	52.31
	16	31.43	20.12	**70.54**	60.68	60.87	63.67	55.58	**74.73**
	21	42.74	30.41	73.63	60.06	67.23	64.44	**86.53**	**95.71**
rye	7	74.88	73.66	**86.81**	83.66	85.92	**85.21**	81.21	82.93
	12	75.11	74.17	**92.72**	**92.82**	91.35	87.74	83.23	80.04
	16	77.13	75.22	**93.66**	**94.15**	92.04	86.95	88.38	83.44
	21	75.42	71.42	93.69	94.44	**94.42**	**96.62**	90.01	92.72

We did not assess quantitatively the spatial accuracy of the methods. However, Figure 23 provides some visual perception on how the four methods perform under this point of view. It shows some clippings of the predictions maps produced by the four methods. For conciseness we show for each method (four groups of two images) the results for sequence lengths 5 (left image within the group) and 22 (right image within the group). Results show that for all frameworks temporal information improves the accuracy of the output crop map. Generally, FCN-PL and CNN-PC produced smoother maps, whereas the salt-and-pepper effect is more apparent in the RF-PB and AE-PB outcomes.

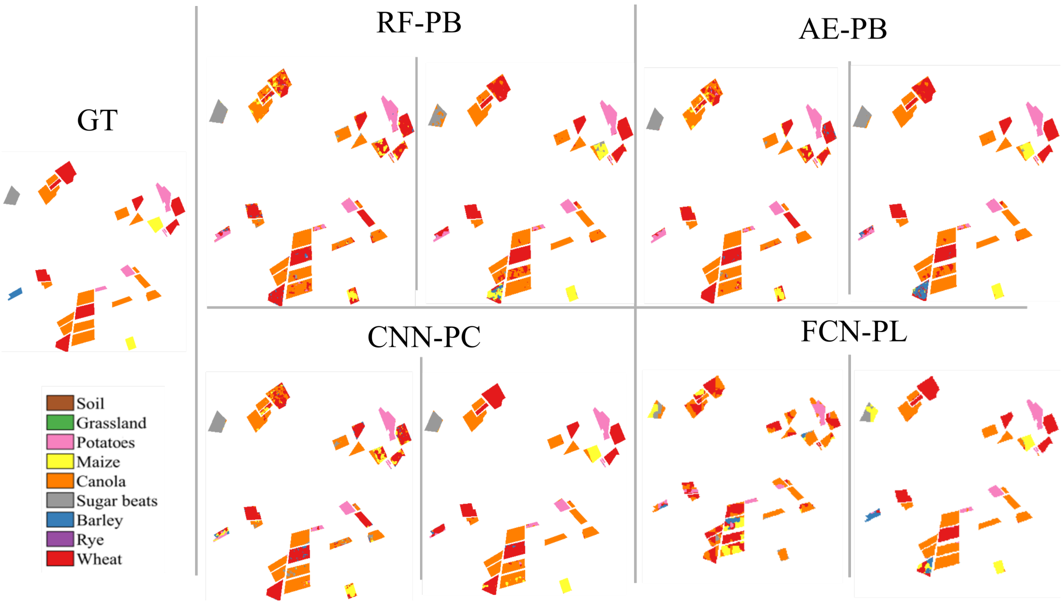


Figure 23: Hanover single class. Example of predictions for sequence lengths 5 and 22. GT stands for ground truth.

Results for Campo Verde: The results for Seq-1 are shown in Figure 24, Figure 25 and Table 10. Similar to the results drawn from the experiments on Hanover, temporal information helped to improve the classification performance for all evaluated frameworks. Improvements on Average F1 from 16% to 60% (RF-PB), from 16% to 56% (AE-PB), from 15% to 61% (CNN-PC) and from 20% to 59% (FCN-PL) were measured. In terms of AA, improvements from 30% to 73% (RF-PB), from 34% to 73% (AE-PB), from 35% to 76% (CNN-PC) and from 34% to 78% (FCN-PL) were attained.

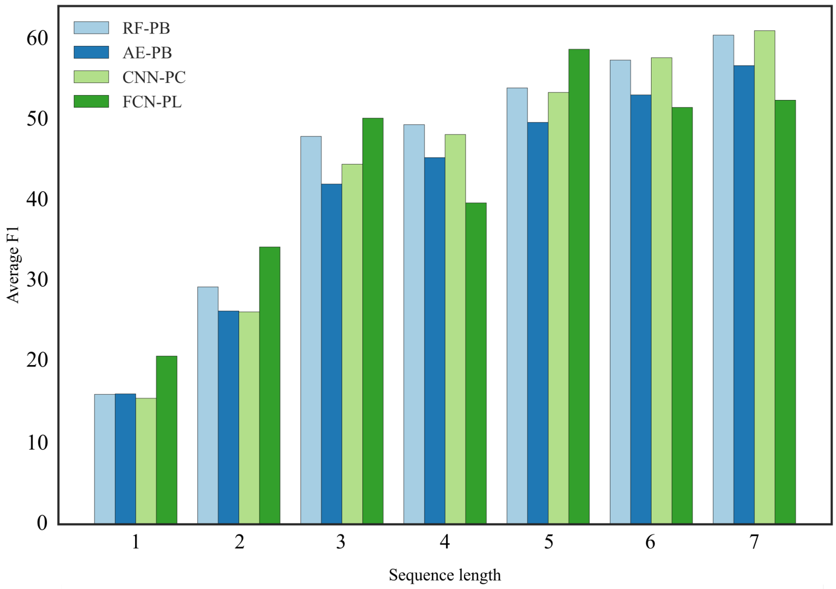


Figure 24: Average F1 for different sequences, taking the last image in the sub-sequence and adding earlier images to classify the season. Campo Verde Seq-1.

Similarly, for Seq-2 (see Figure 26) we obtained improvements on average F1 from 19% to 57% (RF-PB), from 25% to 57% (AE-PB), from 21% to 58% (CNN-PC) and from 27% to 56% (FCN-PL). In terms of AA we recorded improvements (see 27), they were observed improvements from 31% to 69% (RF-PB), from 37% to 68% (AE-PB), from 38% to 72% (CNN-PC) and from 41% to 67% (FCN-PL).

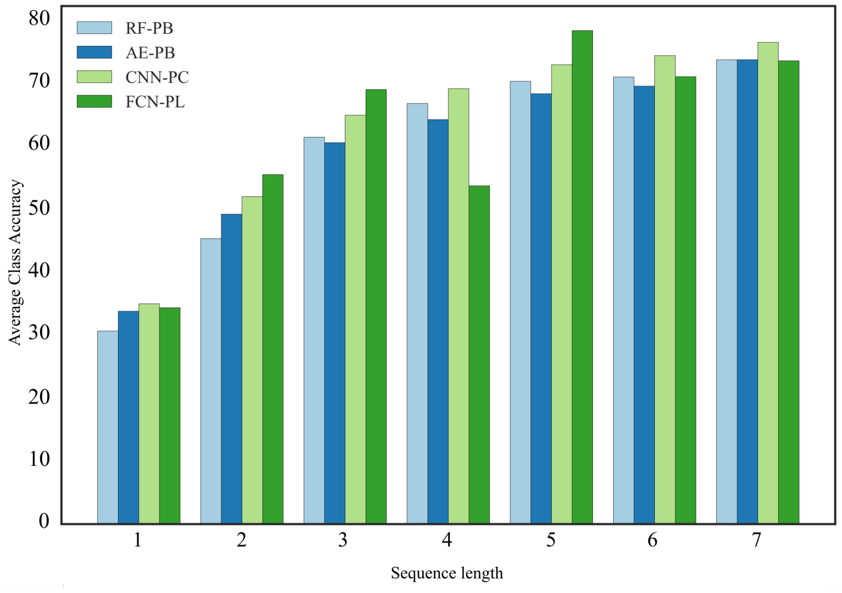


Figure 25: AA for different sequences, taking the last image in the sub-sequence and adding earlier images to classify the season. Campo Verde Seq-1.

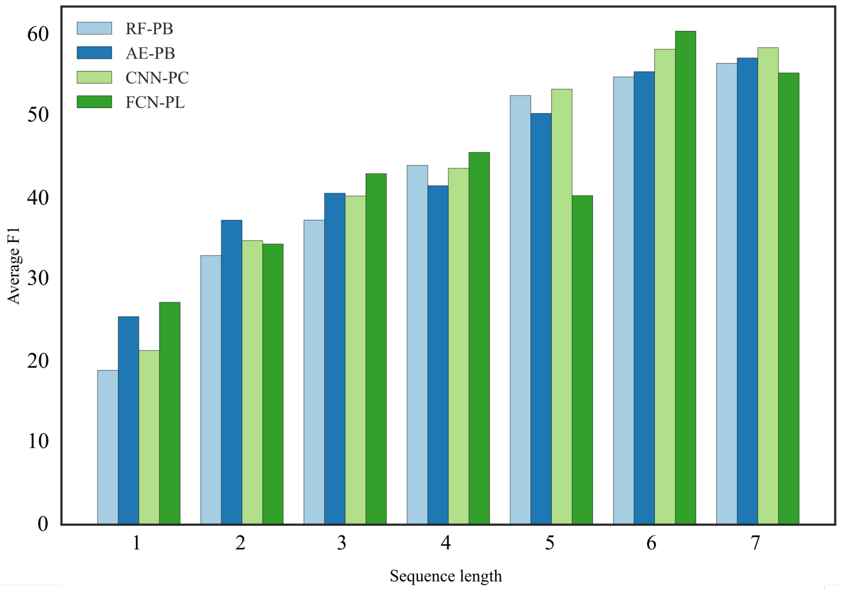


Figure 26: Average F1 for different sequences, taking the last image in the database and adding earlier images to classify the season. Campo Verde Seq-2.

The results in Table 10 and Table 11 are consistent with the results exhibited so far; by and large OA and Kappa index improved as more images were added to the sequence.

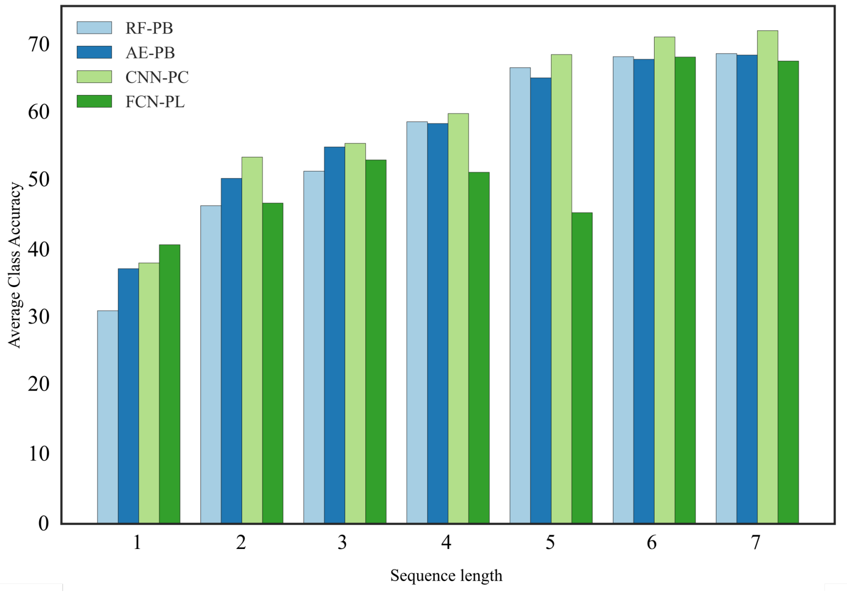


Figure 27: AA for different sequences, taking the last image in the database and adding earlier images to classify the season. Campo Verde Seq-2.

Table 10: Campo Verde Seq-1: OA and Kappa index for different sequence lengths.

	RF-PB		AE-PB		CNN-PC		FCN-PL	
Sequence Length	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**
1	40.30	17.17	40.27	17.58	38.04	16.32	**53.07**	**21.92**
2	60.97	33.24	61.56	35.74	60.02	34.09	**72.38**	**47.99**
3	77.88	56.03	75.26	52.79	77.18	55.65	**86.13**	**70.35**
4	80.87	61.15	77.97	56.97	81.49	**62.67**	**82.77**	57.08
5	85.09	68.70	83.44	65.88	86.41	71.36	**91.64**	**81.14**
6	85.86	70.10	84.65	68.03	**87.37**	**73.10**	86.98	71.62
7	86.91	72.22	85.54	69.85	87.96	74.16	**89.03**	**76.31**

The FCN-PL performance declined considerably in both sub-sequences for some sequence lengths, specifically, 4 for Seq-1 and 5 for Seq-2. One possible reason for this behavior lies in the dynamics of some crops. Take as example soybean, which is the dominant crop in Seq-1. It can be easily inferred from Figure 20 that for soybean the seeding and, consequently, the harvest epoch vary considerably within the sequence. Thus, in one epoch soybean might be in different phenological stages depending on the parcel being imaged. Under this conditions, a classifier can get confused because the training set contains samples in different phenological stages for the same crop. The climate

Table 11: Campo Verde Seq-2: OA and Kappa index for different sequence lengths.

	RF-PB		**AE-PB**		**CNN-PC**		**FCN-PL**	
Sequence Length	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**
1	33.39	20.62	52.11	36.90	41.35	28.53	**58.17**	**42.81**
2	66.14	53.14	68.03	55.77	63.47	50.90	**71.34**	**58.47**
3	71.01	59.18	71.07	59.45	69.19	57.42	**78.01**	**67.93**
4	74.87	64.33	73.70	62.85	74.53	64.07	**83.75**	**75.66**
5	78.33	68.87	75.82	65.63	**78.55**	**69.27**	73.75	60.98
6	80.99	72.55	78.96	69.82	81.70	73.30	**86.36**	**79.75**
7	82.40	74.42	80.19	71.41	83.27	75.35	**84.65**	**77.20**

conditions in Hanover dataset do not permit such a variation in seeding/harvest times. Even more important, was the difficulty to deal with unbalanced training samples among classes in FCN-PL. Recall that for the RF, AE and CNN each sample consists exactly of one pixel. For the FCN-PL architecture, a sample is an image patch that carries all the classes of the pixels in it. In such cases, some crops might have been not properly represented in the training sets and the classifier accuracy might have been impacted. Nevertheless, FCN-PL was the most accurate in terms of OA and Kappa for almost all sequence lengths in both sub-sequences. In spite of comparatively higher values for OA, the average F1 and AA were lower due to low accuracies for classes with few samples.

Table 12 (Seq-1) and Table 13(Seq-2) shows the F1 and class accuracy (Acc) for *soybeans*, *maize*, *cotton* and *beans*. For conciseness, we show in these tables only the performance for sequence lengths equal to 1, 4 and 6. For Seq-1, FCN-PL was the most accurate approach in almost all experiments, except for length 6, where performance droped for *maize*, *cotton*, and *beans*. Notice that at this point images from November were added, which, compared with images from December, contained a considerable percentage of parcels in soil stage. For Seq-2, FCN-PL was consistently the most accurate for *maize* and *cotton*, and also the worst for *beans* and *soybeans* (classes with less abundant samples) for sequence lengths 1 and 4.

Table 12: Campo Verde Seq-1: F1 and Accuracy (Acc) per class for different sequence lengths.

		RF-PB		AE-PB		CNN-PC		FCN-PL	
SeqLen	**Crop**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**
1	**Soybeans**	58.32	42.18	58.19	41.84	57.08	40.62	**70.44**	**59.10**
	Maize	9.39	29.93	9.148	32.59	8.29	38.07	**12.31**	**47.66**
	Cotton	4.63	21.64	4.17	19.64	4.86	28.2	**12.86**	**31.41**
	Beans	2.44	33.62	2.83	36.77	2.27	51.57	**4.99**	**54.69**
4	**Soybeans**	91.82	86.91	90.00	83.59	92.29	87.12	**92.76**	**96.30**
	Maize	41.97	82.88	39.44	88.82	47.56	93.46	**53.61**	**97.24**
	Cotton	**43.04**	**74.77**	28.17	60.44	31.65	76.33	23.27	63.94
	Beans	39.29	93.57	24.46	97.07	29.47	**99.27**	**46.36**	90.78
6	**Soybeans**	94.41	91.18	93.37	89.56	95.15	92.22	**95.17**	**93.50**
	Maize	55.58	88.93	43.58	91.48	**57.61**	**97.44**	41.71	96.42
	Cotton	**54.13**	79.98	40.47	74.54	49.92	**90.47**	41.68	85.61
	Beans	**41.59**	96.07	21.77	96.26	30.54	**99.16**	21.07	98.62

Table 13: Campo Verde Seq-2: F1 and Accuracy (Acc) per class for different sequence lengths.

		RF-PB		AE-PB		CNN-PC		FCN-PL	
SeqLen	**Crop**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**
1	**Soybeans**	3.41	6.60	**3.86**	**23.06**	3.58	13.02	0	0
	Maize	26.90	16.51	72.14	68.79	62.73	50.15	**73.16**	**70.96**
	Cotton	59.54	47.08	59.64	46.52	52.96	38.35	**68.38**	**58.39**
	Beans	0.09	6.37	0.44	0.26	**0.87**	**10.59**	0	0
4	**Soybeans**	9.88	27.60	8.49	**30.67**	**9.97**	11.71	0	0
	Maize	79.95	73.50	80.16	74.07	79.73	72.64	**87.13**	**89.86**
	Cotton	89.93	86.35	89.21	84.70	89.40	85.29	**93.77**	**93.19**
	Beans	7.33	49.78	8.00	50.90	**8.87**	**63.91**	0	0
6	**Soybeans**	44.45	67.32	56.38	73.32	55.61	**79.59**	**59.36**	61.27
	Maize	85.68	81.93	83.91	80.26	84.99	81.22	**90.08**	**88.84**
	Cotton	91.71	88.74	90.15	86.04	91.66	90.65	**93.98**	**94.63**
	Beans	10.28	54.95	13.75	57.81	16.13	**74.76**	**30.79**	36.35

Figure 28(Seq-1) and Figure 29(Seq-2) shows some clippings of the predictions maps produced by the four methods for both sub-sequence. For conciseness we show for each method (four groups of two images) the results for sequence lengths 1 (left image within the group) and 6 (right image within the group). Results show that for all frameworks temporal information improves considerably the accuracy of the output crop map. As in Hanover dataset, FCN-PL produced smoother maps. The salt-and-pepper effect is even more apparent for the other three methods in comparison with the results from Hanover. In addition, the prediction maps were more accurate for more abundant classes (*soybean* and *maize*) and less accurate for classes that are less present (e.g., *pasture*).

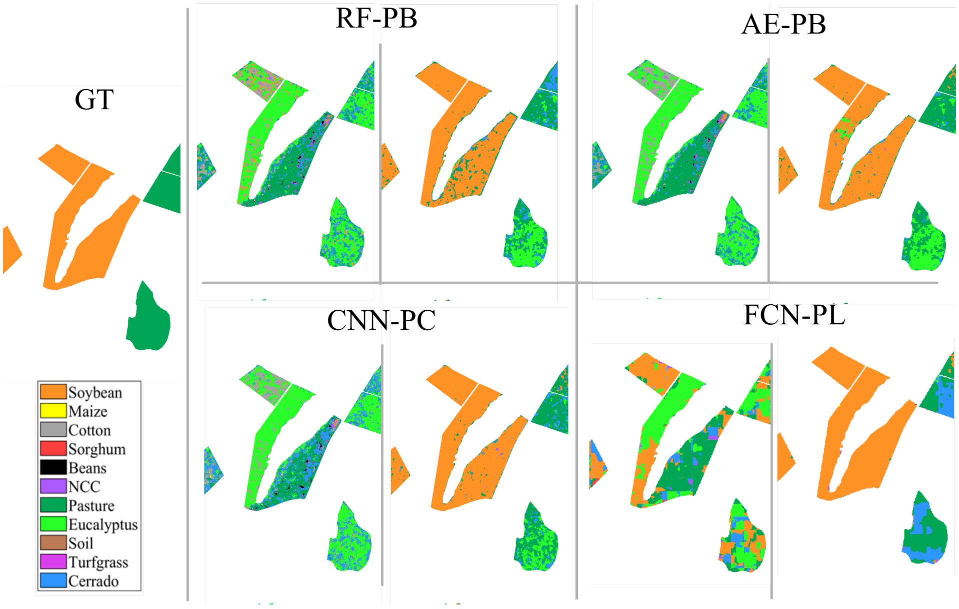


Figure 28: Seq-1, example of predictions for sequence lengths 1 and 6. GT stands for ground truth.

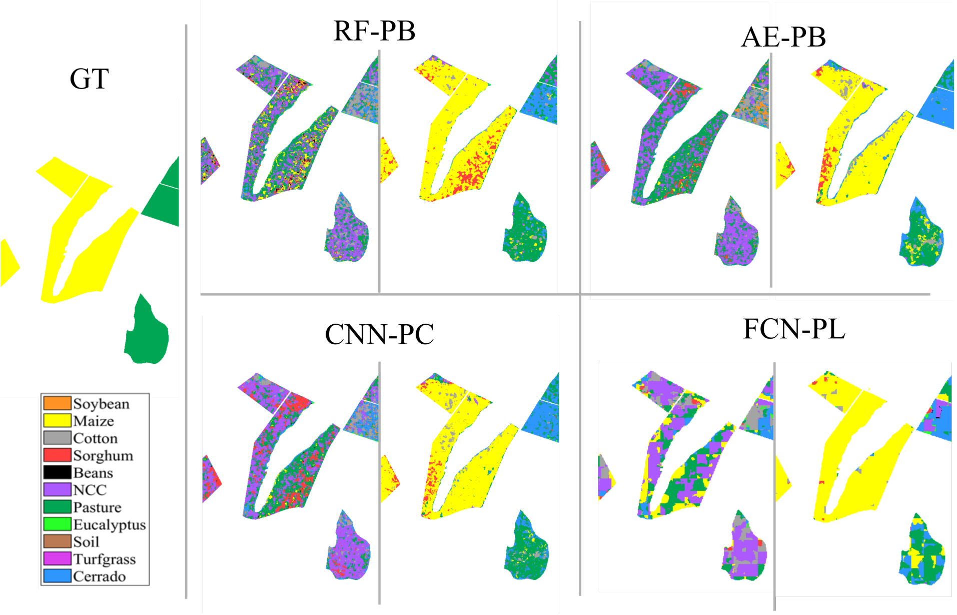


Figure 29: Seq-2, example of predictions sequence lengths 1 and 6. GT stands for ground truth.

5.4.2 Multiclass evaluation protocol

The Hanover dataset in its original form has a single crop per pixel along the whole sequence. In order to test the Multiclass protocol on this dataset the class uncultivated soil was inferred from [94]. We assigned the class label *soil* to the periods outside of the crop life cycle (i.e., soil preparation and post-harvesting) and it splits the sequence in 7 different periods/ground truth (see Figure 30 yellow block). The new class distribution after this modification is shown in Figure 31. Notice that the ground truth from the second date in May (`May_2`) to first date of July (`Jul_1`) corresponds to the original ground truth. Two protocols were adopted in this case:

1. Protocol I: Similar to the experiments dealing with a single class per pixel along the sequence, we classified the most recent image of a sequence adding earlier images successively. Different from the single class analysis, in this case there might be more than one reference map along the sequence. The main objective of this protocol is evaluate the performance of the different approaches in each epoch when information from past is exploited. For conciseness, in the next subsection we only show the results of this protocol in graphic mode (not tables).

2. Protocol II: In this protocol we classified all images within the sequence using the whole set of images. The main objective of this protocol was to evaluate the performance of the different approaches in each epoch when the information from past, present and future is exploited. For Campo Verde dataset images were grouped into 9 different ground truth (see Figure 18) and for Hanover dataset images are grouped into 7 ground truth data (see Figure 31).

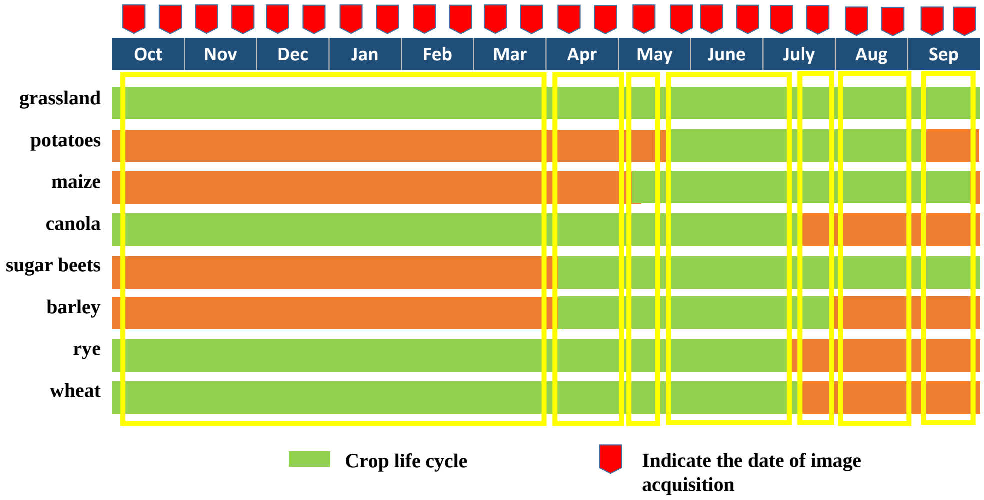


Figure 30: Hanover crop life cycle. Adapted from [94].

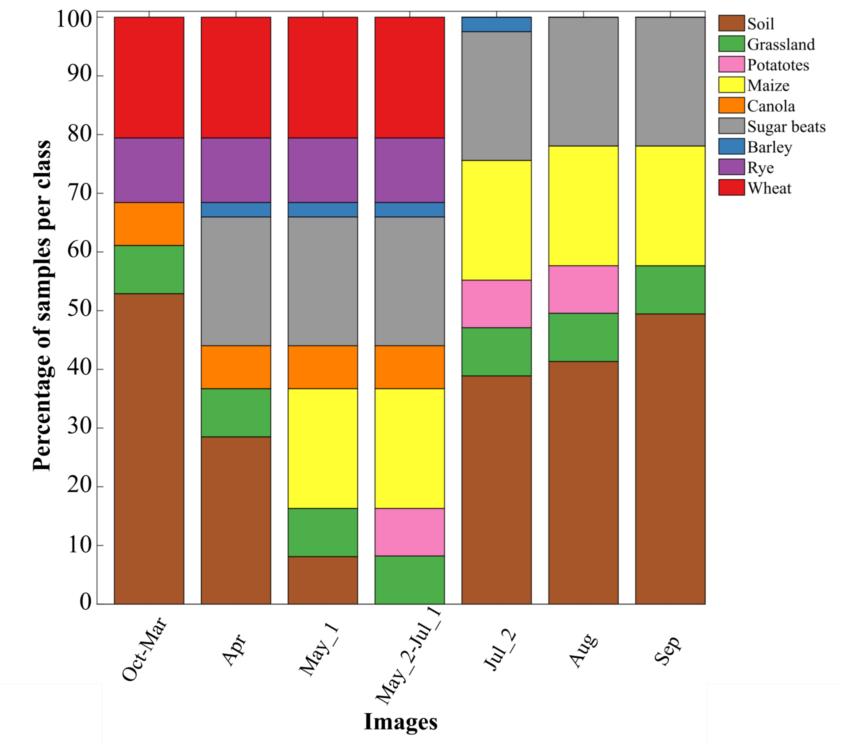


Figure 31: Hanover new class distribution.

Results for Hanover - Protocol I: Figures 32, 34, 36 and 38 show the results for RF-PB, AE-PB, CNN-PC and FCN-PL, respectively, in terms of average F1 (grayish bars), AA (orangish bars) and OA (blueish bars). For conciseness we show the results for the second image acquired at each month (i.e., `Oct_2`,`Nov_2`,..). Each group of bars presents the performance corresponding to the acquisition date indicated on the horizontal axis. The number of bars within a group corresponds to the different sequence lengths. Thus, the leftmost bar of each group corresponds to a single image, the one being classified (i.e., monotemporal classification). Bars to the right indicate the classification performance of the same target image using data of earlier images consecutively. Notice that the leftmost group has the classification for both image of October, the two earliest images in the dataset. The rightmost group has 24 bars corresponding to the maximum number of images in the database. In addition, Figures 33, 35, 37 and 39 shown the Kappa index for RF-PB, AE-PB, CNN-PC and FCN-PL, respectively.

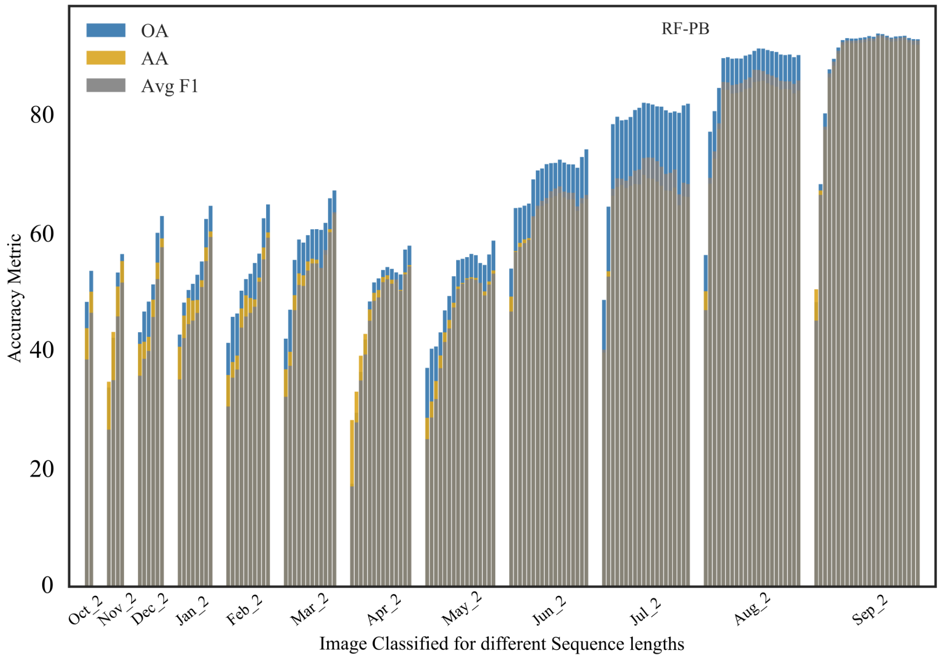


Figure 32: OA (bluish bars), AA (orangish bars) and average F1 (grayish bars). RF-PB performance for different sequences (bar groups), formed by adding earlier images. Hanover.

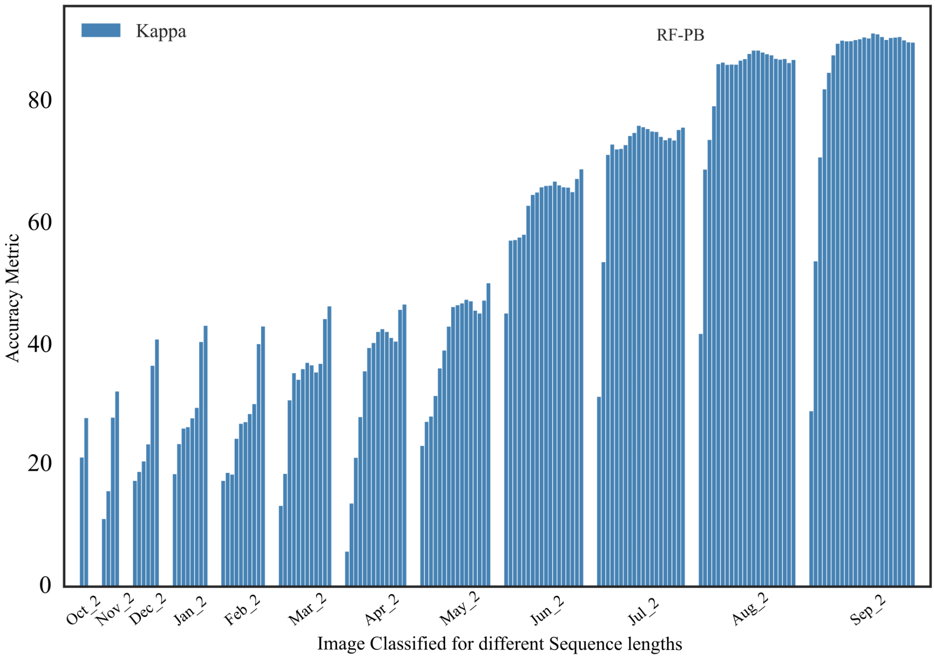


Figure 33: Kappa index. RF-PB performance for different sequences (bar groups), formed by adding earlier images. Hanover.

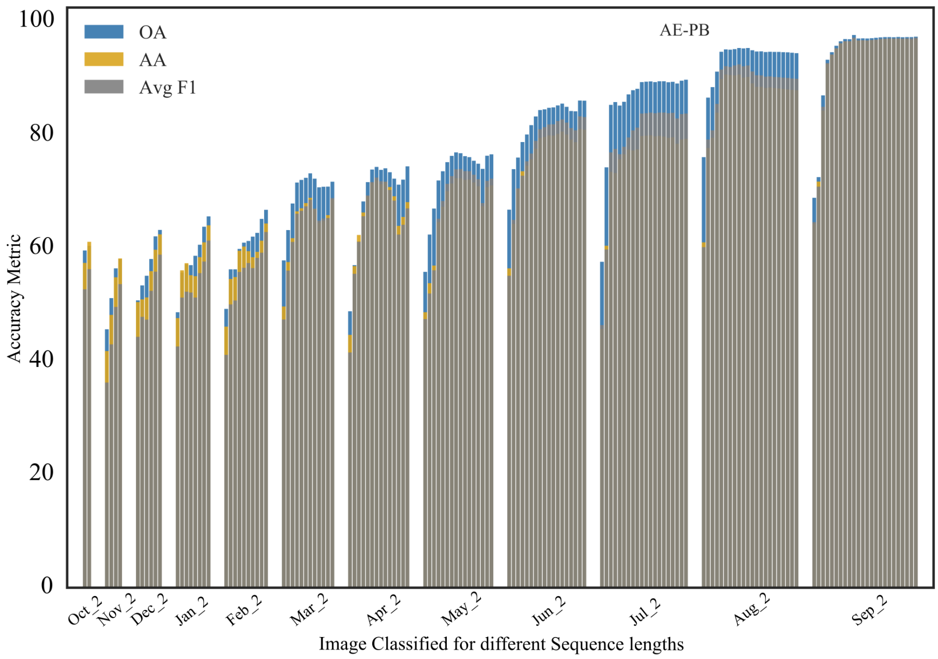


Figure 34: OA (blueish bars), AA (orangish bars) and average F1 (grayish bars). AE-PB performance for different sequences (bar groups), formed by adding earlier images. Hanover.

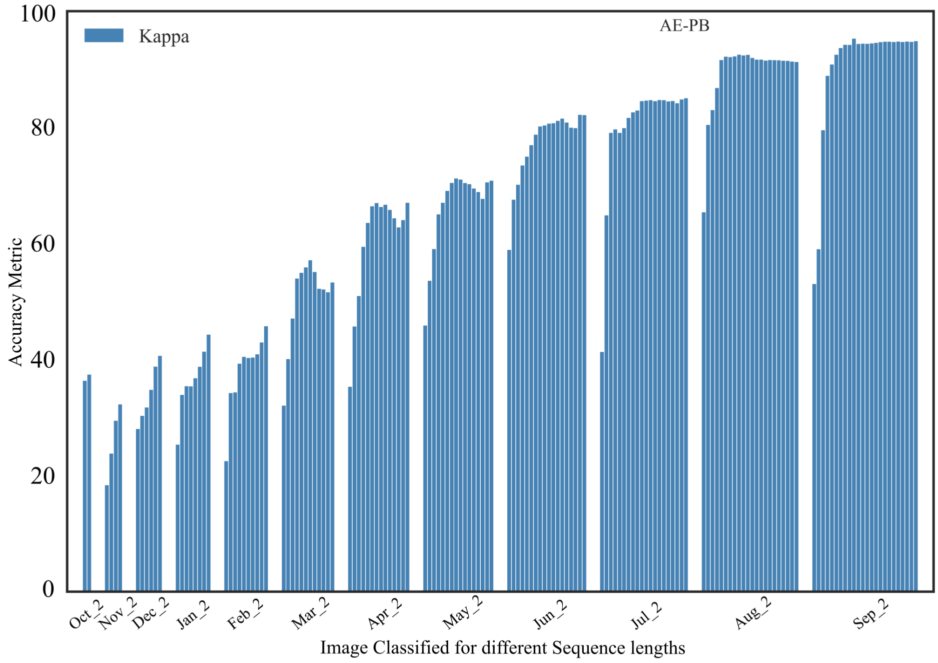


Figure 35: Kappa index. AE-PB performance for different sequences (bar groups), formed by adding earlier images. Hanover.

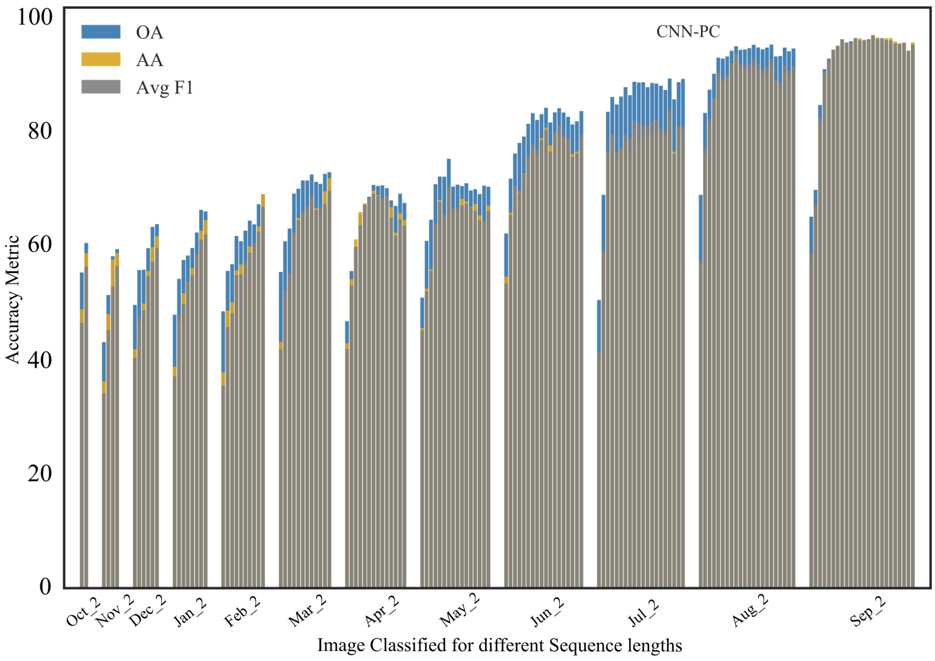


Figure 36: OA (blueish bars), AA (orangish bars) and average F1 (grayish bars). CNN-PC performance for different sequences (bar groups), formed by adding earlier images. Hanover.

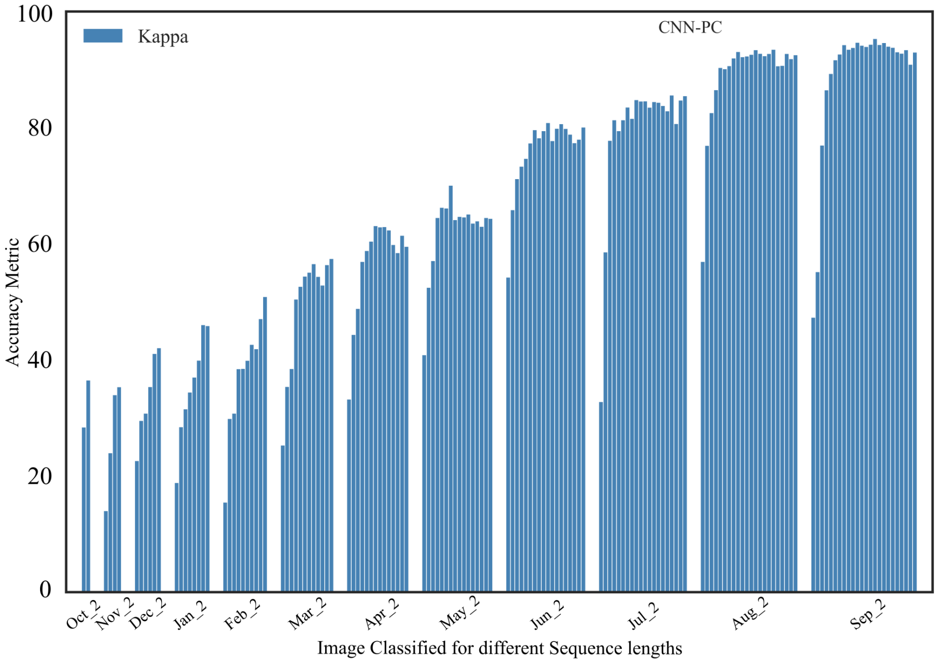


Figure 37: Kappa index. CNN-PC performance for different sequences (bar groups), formed by adding earlier images. Hanover.

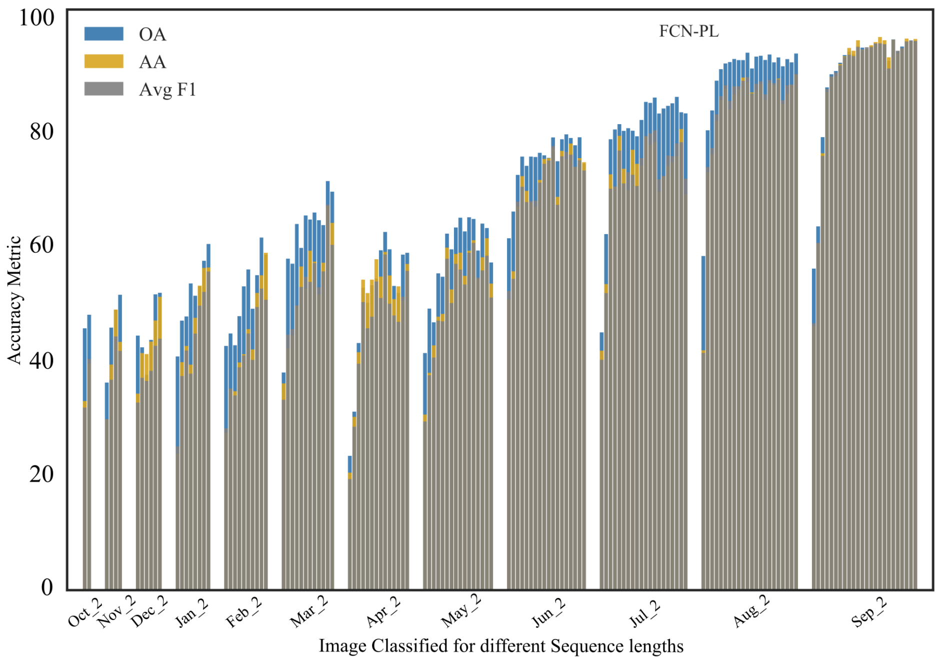


Figure 38: OA (blueish bars), AA (orangish bars) and average F1 (grayish bars). FCN-PL performance for different sequences (bar groups), formed by adding earlier images. Hanover.

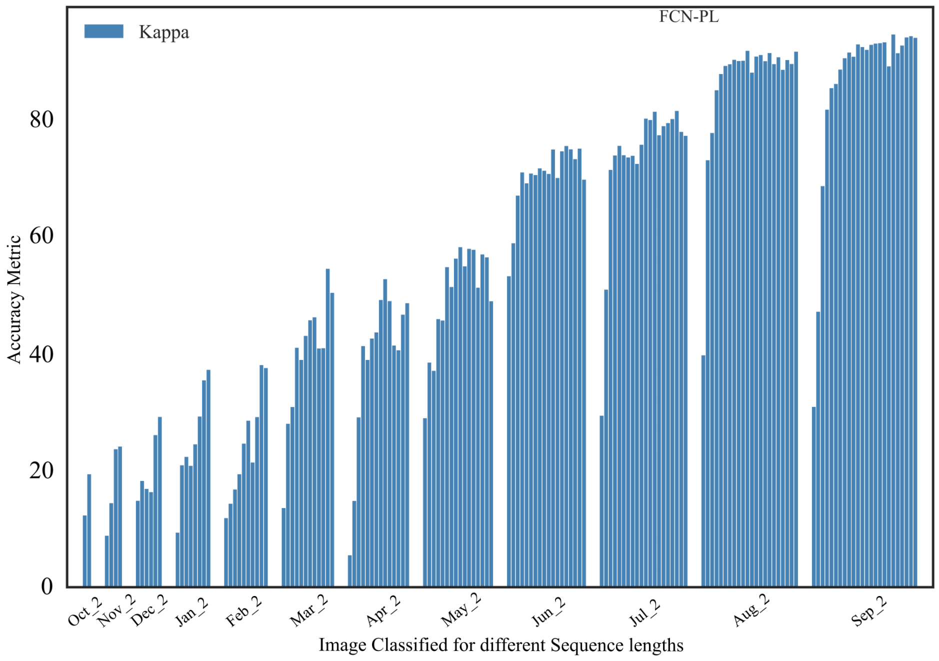


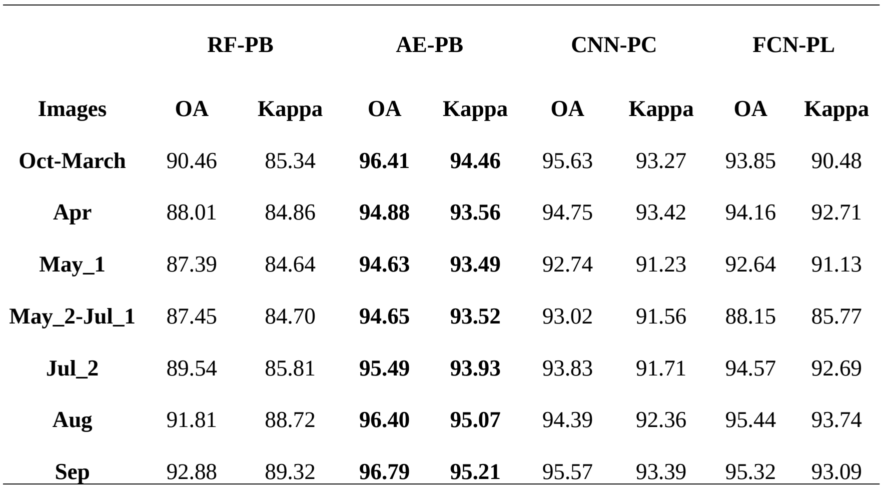
Figure 39: Kappa index. FCN-PL performance for different sequences (bar groups), formed by adding earlier images. Hanover.

Clearly, all metrics tended to increase as prior images were added to the sequence in almost all experiments. This improvement was generally significant for sequence lengths from 2 to 5 images, staying nearly constant or even declining for longer sequences. A similar behaviour showed the results for average F1 and AA. The plot reveals AE-PB as the best performing approach in most cases, followed by CNN-PC, RF-PB and FCN-PL. The plot also shows that performance varied depending on the epoch, reaching the highest values in September.

For FCN-PL and RF-PB a considerable decline was observed in April and May. This can be understood by looking at the class occurrence histogram in Figure 31. From April to the second date of May there were more crops at the beginning of their life cycle (e.g., *barley*, *maize* and *potatoes*), making the classification more challenging. Thus, adding earlier images, which contain mostly class *soil*, brought moderate gains.

Results for Hanover Protocol II: Table 14, Figure 40 and Figure 41 show that the use of data from earlier and later epochs brought a considerable improvement for all images/epochs in the database. Again the DL frameworks outperformed RF, from 84% to 97% for both, average F1 score and AA. In terms of OA and Kappa index, DL approaches achieved values above 91%, being AE-PB the best performing framework in almost all experiments, followed by FCN-PL. AE-PB was once again the best approach followed by FCN-PL and CNN-PC.

Table 14: OA and Kappa index for all images within the sequence. Hanover dataset.



	RF-PB		AE-PB		CNN-PC		FCN-PL	
Images	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**
Oct-March	90.46	85.34	**96.41**	**94.46**	95.63	93.27	93.85	90.48
Apr	88.01	84.86	**94.88**	**93.56**	94.75	93.42	94.16	92.71
May_1	87.39	84.64	**94.63**	**93.49**	92.74	91.23	92.64	91.13
May_2-Jul_1	87.45	84.70	**94.65**	**93.52**	93.02	91.56	88.15	85.77
Jul_2	89.54	85.81	**95.49**	**93.93**	93.83	91.71	94.57	92.69
Aug	91.81	88.72	**96.40**	**95.07**	94.39	92.36	95.44	93.74
Sep	92.88	89.32	**96.79**	**95.21**	95.57	93.39	95.32	93.09

Results for Campo Verde - Protocol I: Figure 42, 44, 46 and 49 show the results recorded in our experiments on Campo Verde (protocol I) for RF-PB, AE-PB, CNN-PC and FCN-PL, respectively. The figures summarize the results in terms of average F1 (grayish bars), AA (orangish bars) and OA (blueish bars) for each image, whereby `Month_#`, stands for first and second

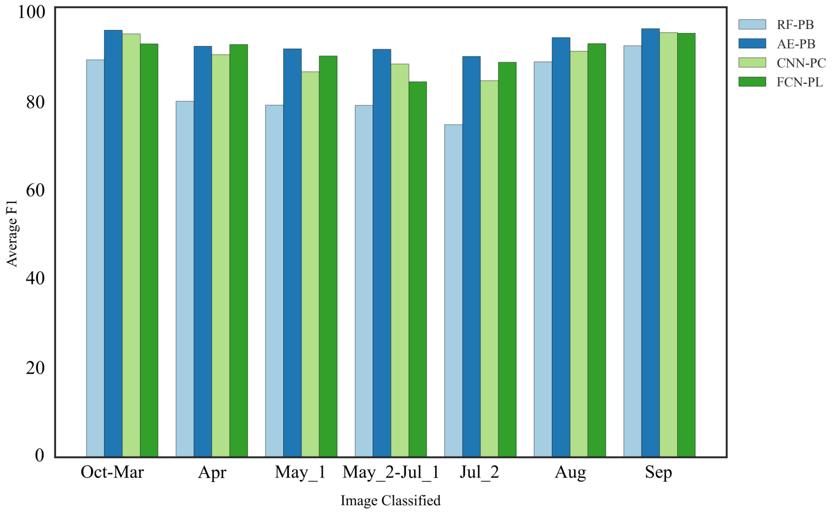


Figure 40: Average F1 for all images within the sequence. Hanover dataset.

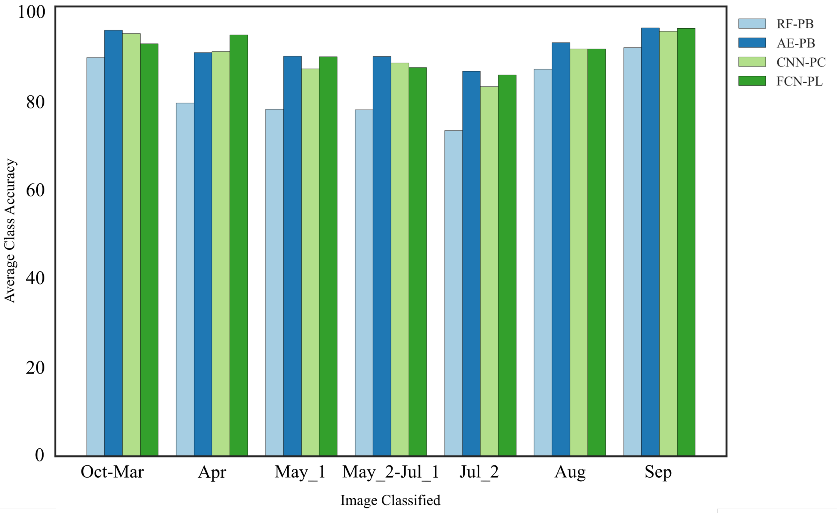


Figure 41: AA for all images within the sequence. Hanover dataset.

image for November, December, March, May and July. The bars to the right indicate the classification performance of the same target image when more earlier images were considered as data input. Notice that the leftmost group has only the classification for October, the earliest image in the dataset. The rightmost group has 14 bars corresponding to the maximum number of images in the database. In addition, Figures 43, 45, 47 and 49 show the Kappa index achieved by RF-PB, AE-PB, CNN-PC and FCN-PL, respectively.

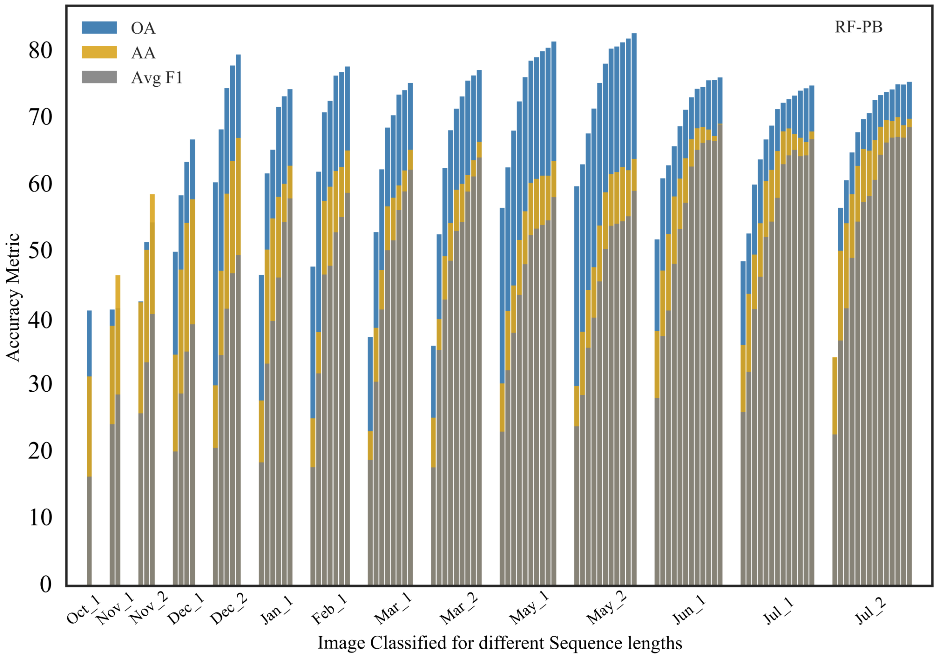


Figure 42: OA (blueish bars), AA (orangish bars) and average F1 (grayish bars). RF-PB performance for different sequences (bar groups), formed by adding earlier images. Campo Verde.

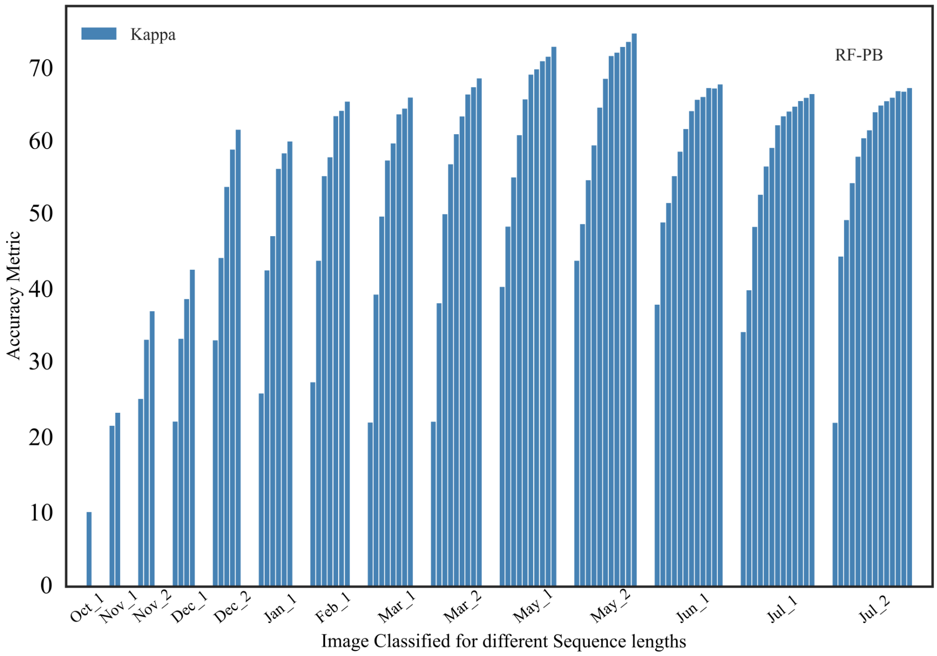


Figure 43: Kappa index. RF-PB performance for different sequences (bar groups), formed by adding earlier images. Campo Verde.

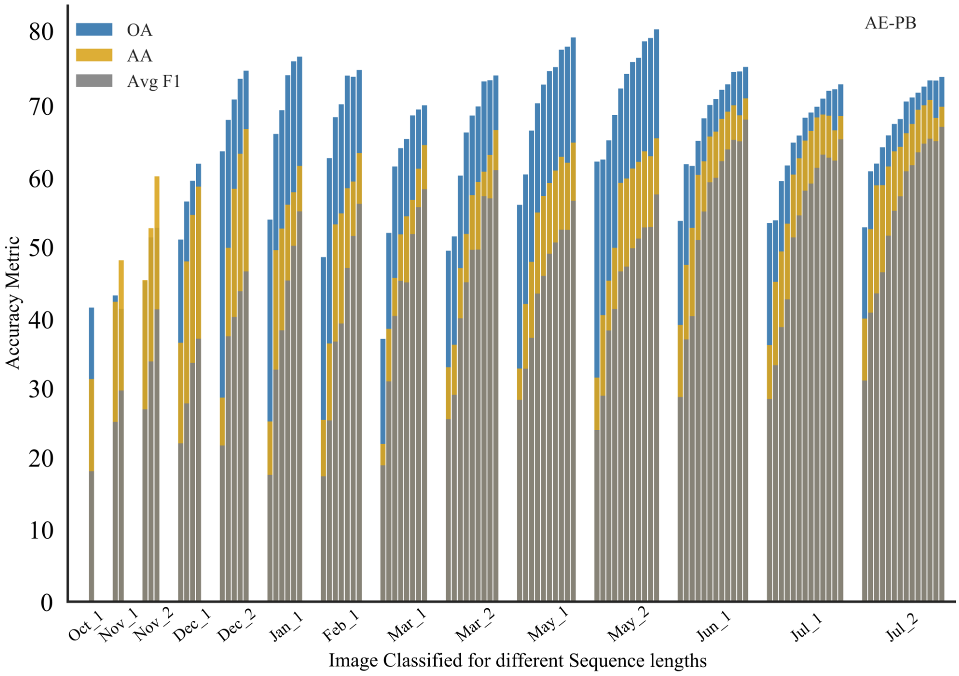


Figure 44: OA (blueih bars), AA (orangish bars) and average F1 (grayish bars). AE-PB performance for different sequences (bar groups), formed by adding earlier images. Campo Verde.

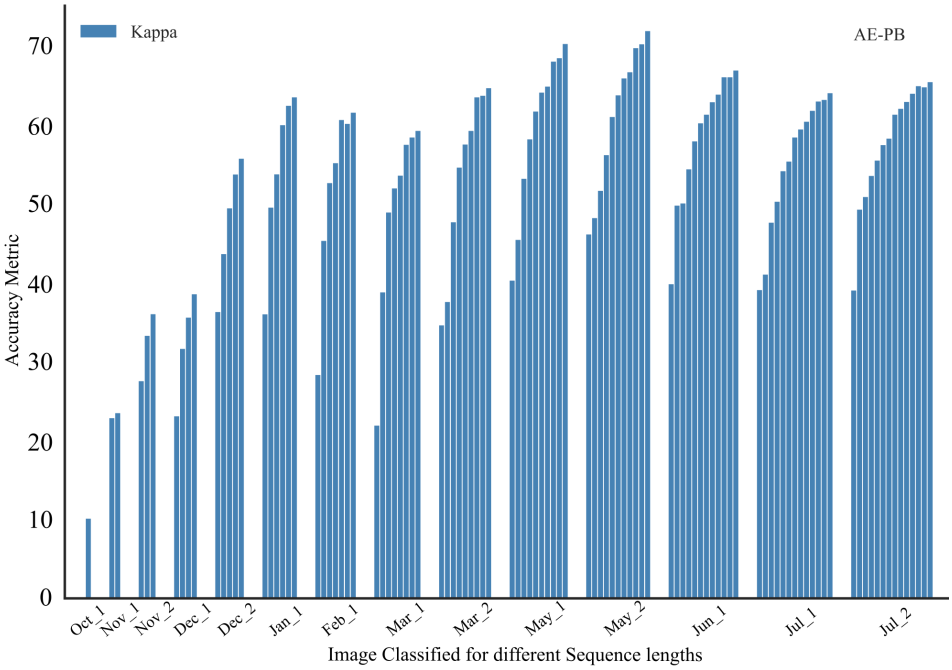


Figure 45: Kappa index. AE-PB performance for different sequences (bar groups), formed by adding earlier images. Campo Verde.

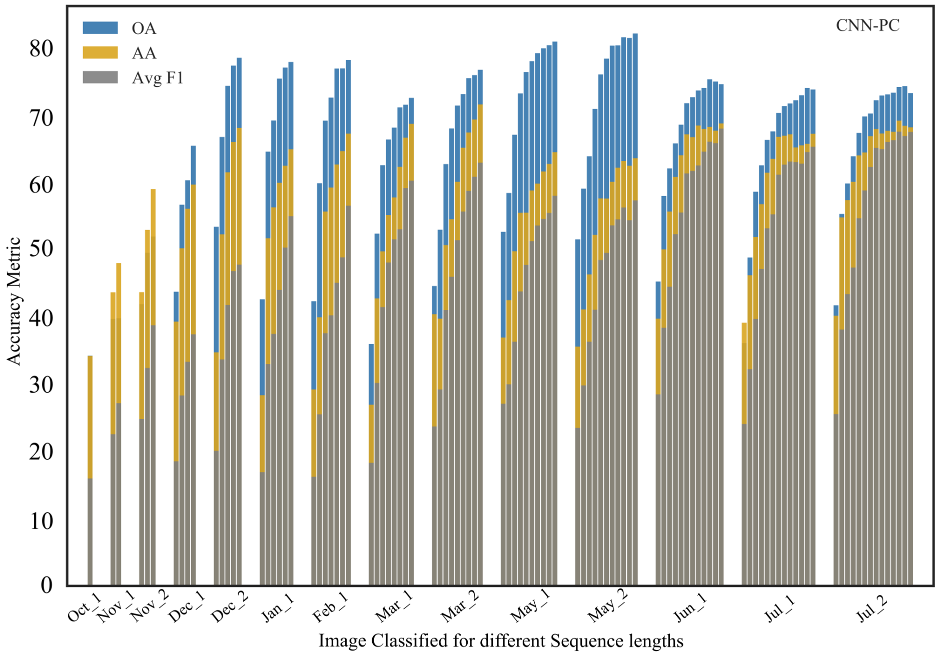


Figure 46: OA (blue bars), AA (orange bars) and average F1 (gray bars). CNN-PC performance for different sequences (bar groups), formed by adding earlier images. Campo Verde.

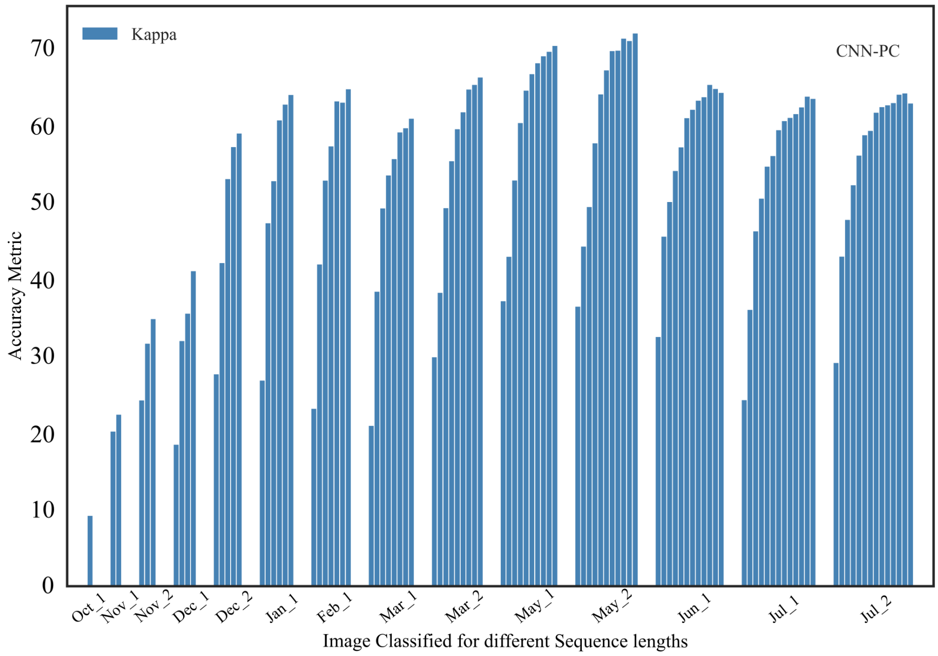


Figure 47: Kappa index. CNN-PC performance for different sequences (bar groups), formed by adding earlier images. Campo Verde.

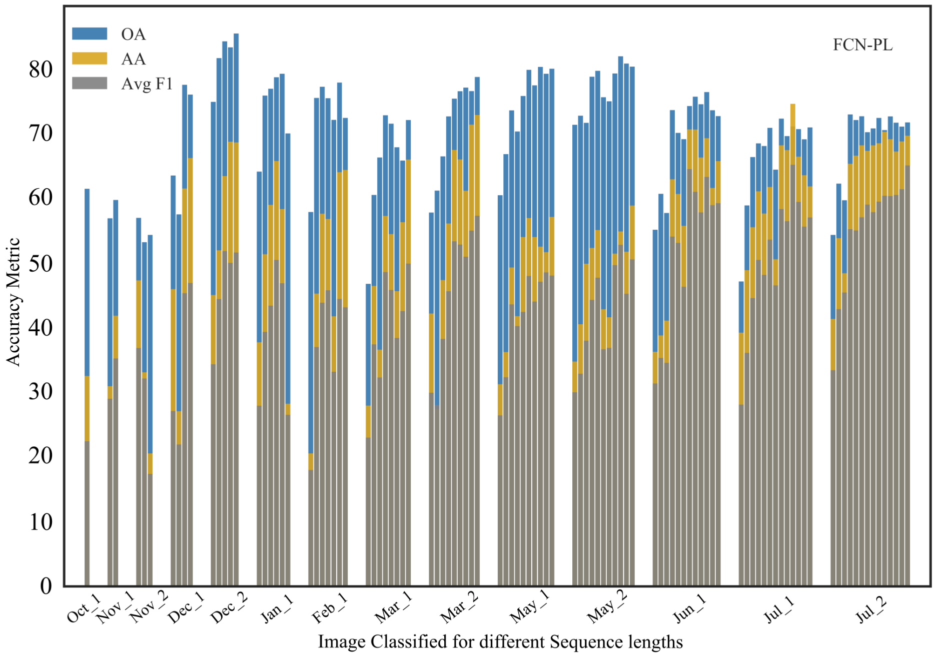


Figure 48: OA (blue bars), AA (orange bars) and average F1 (gray bars). FCN-PL performance for different sequences (bar groups), taking images in the dataset and adding former images to classify that image. Campo Verde.

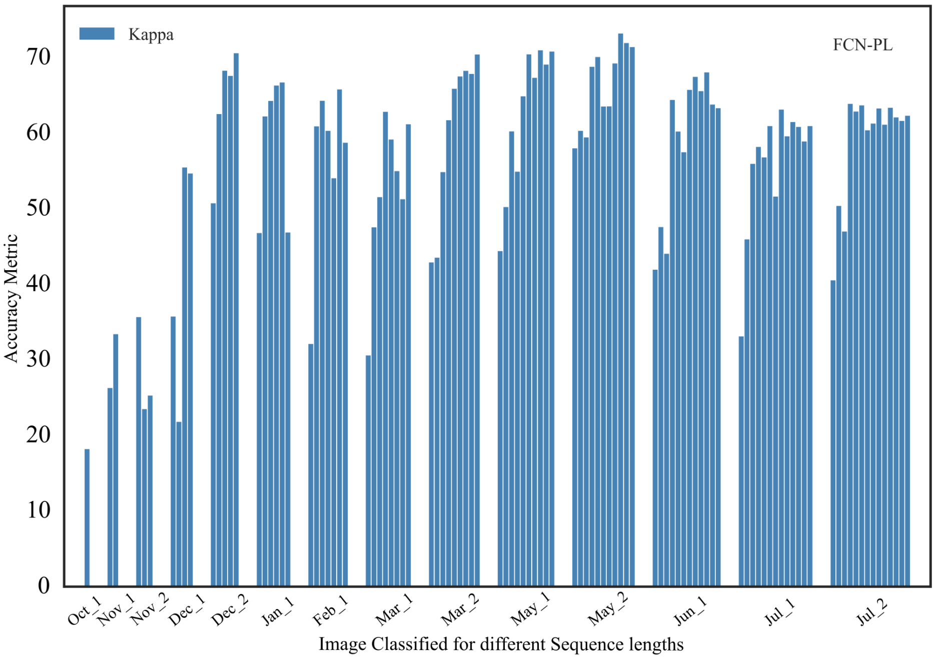


Figure 49: Kappa index. FCN-PL performance for different sequences (bar groups), taking images in the dataset and adding former images to classify that image. Campo Verde.

As observed in the discussion of the results on Hanover dataset, all metrics tended to increase as prior images were added to the sequence in almost all experiments. In certain cases, for longer sequences (see three leftmost groups), the inclusion of one more image to the sequence was even deleterious. The For RF-PB, AE-PB, and CNN-PC this improvement was generally significant for sequences with 2 to 6 images, staying nearly constant for longer sequences.

On the other hand, FCN-PL manifested a different behavior, being the most accurate one for short sequences. Almost all images reached more than 60% in terms of OA for sequences containing one or two images. Contrary to other methods, FCN-PL captures the structure in terms of classes withing the patch, which reduces the salt-and-paper effect in the classification. Also, for some images FCN-PL reached low average F1 and AA values compared with other models. Again, this can be explained by the difficulty to balance training samples among classses when working with FCN-PL. Similar to what had been observed for the single class analysis, in some cases adding more images to the sequence was even deleterious.

Results for Campo Verde - Protocol II: Figure 50, Figure 51 and Table 15 show the results of experiments on Campo Verde following Protocol II. Once again the use of data from earlier and later epochs improved the accuracy. Compared with Protocol I the benefits of exploiting data of earlier epochs were more significant here. RF-PB and CNN-PC alternated as the best performing method in term of OA and AA.

The average F1 score for CNN-PC and RF-PB exhibited comparable results. As expected FCN-PL showed low values in terms of average F1 compared with the other three methods. For AA, CNN-PC was the best performing approach in six out of the nine months, following by FCN-PL and RF-PB methods.

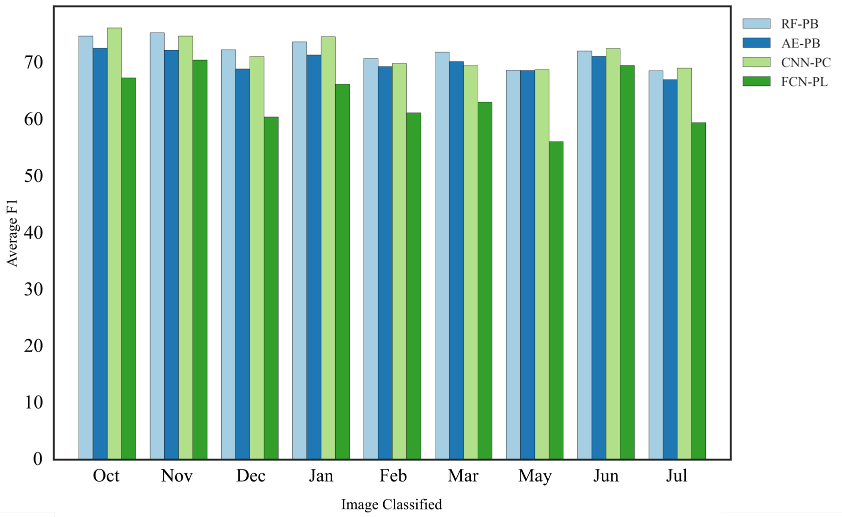


Figure 50: Average F1 for all images within the sequence. Campo Verde dataset.

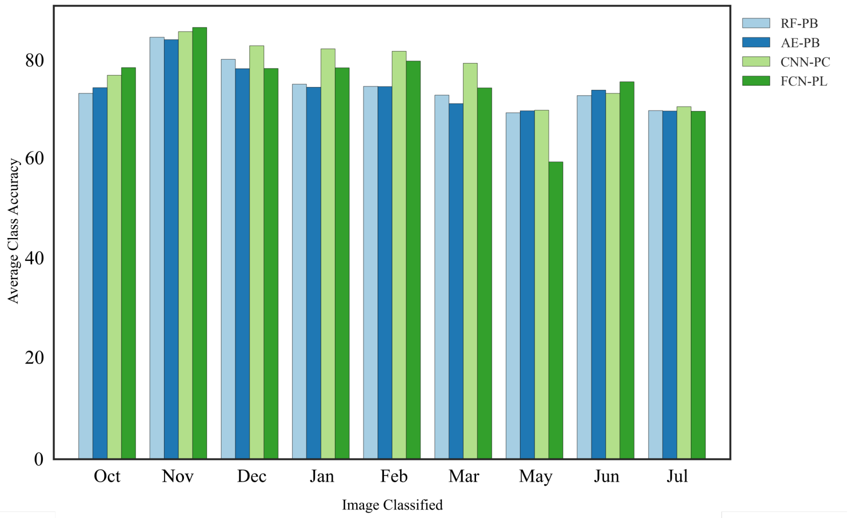


Figure 51: AA for all images within the sequence. Campo Verde dataset.

Table 15: OA and Kappa index for all images within the sequence. Campo Verde dataset.

	RF-PB		**AE-PB**		**CNN-PC**		**FCN-PL**	
Images	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**	**OA**	**Kappa**
Oct	91.46	75.20	89.90	72.12	**92.99**	**79.41**	91.66	76.22
Nov	82.37	72.93	79.57	69.51	**83.33**	**74.11**	81.06	71.31
Dec	91.51	82.82	89.62	79.46	**92.15**	**83.98**	91.37	82.47
Jan	83.50	73.91	81.63	71.16	**85.32**	**76.67**	85.04	76.21
Feb	86.20	78.17	84.94	76.35	**87.30**	**79.76**	85.76	77.49
Mar	**81.44**	**74.15**	79.60	71.76	81.20	73.73	78.77	70.61
May	**87.76**	**81.92**	86.61	80.39	87.16	80.92	85.15	77.99
Jun	**78.83**	**71.37**	78.25	70.77	78.12	70.36	75.08	66.59
Jul	**75.43**	**67.08**	74.18	65.70	74.61	65.89	71.04	61.13

Figure 52 presents some clippings of the predictions maps produced by the CNN-PC approach (generally presents a more stable behavior over the DL outcomes). For conciseness we show the results for October, December and March. The prediction maps were more accurate for more abundant classes: *soil* for October, *soybeans* for December and *cotton* for March.

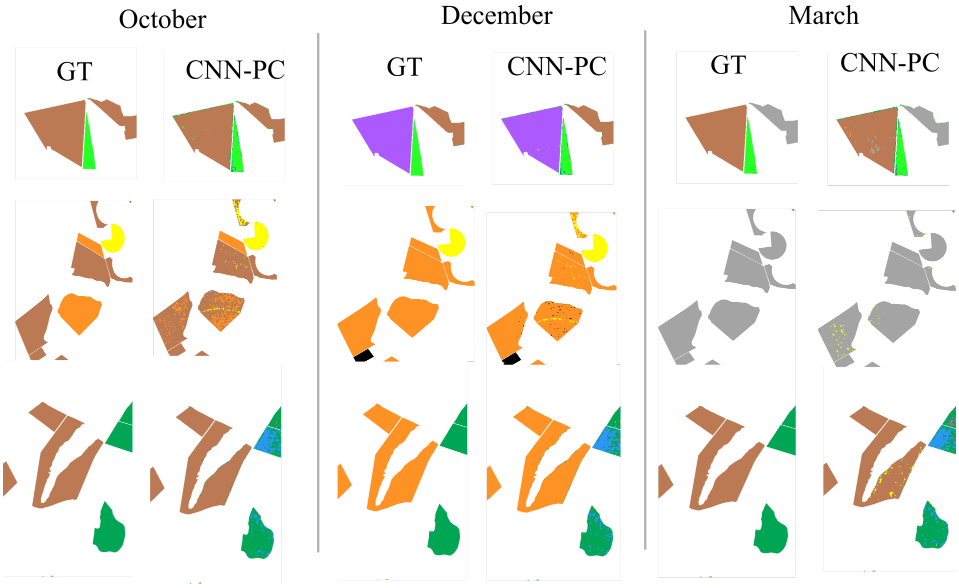


Figure 52: Campo Verde. Example of predictions for October, December and March. GT stands for ground truth. Same color legend as in Figure 29.

Inference time CNN versus FCN: Table 16 presents the training and inference time for the CNN-PC and FCN-PL models (example case for Single Class experiment on Campo Verde dataset). As expected, the FCN-PL average training time per epoch was longer (approximately twice) than for CNN-PC. On the other hand, the FCN-PL inference time was more than one hundred times shorter than for CNN-PC. It can be easily demonstrated that these differences would have been even larger if we had adopted larger patches for CNN-PC.

Table 16: Average training and inference time for the CNN-PC and FCN-PL models.

	CNN-PC	FCN-PL
Training (s/epoch)	31	61
Inference (s)	240	2

Single class versus Multiclass: For decision makers the information what matters in most cases is the crop in the harvest rather than the crop in each month. However, the multiclass approach delivers a class in each epoch. In order to compare the single class and multiclass variants, we adopted a majority voting strategy to summarize the multiclass responses into a single response per image site for the whole observed sequence. For the Hanover dataset, the results of AE-PB (best performing model) Protocol II were compared with

AE-PB single class results. For both Campo Verde sub-sequences following Protocol II, we only took the results of CNN-PC, since it achieved the best results in terms of absolute performance and stability.

Table 17 summarizes the results in terms of OA and Kappa index. The results for the single class and the multiclass approaches were not significantly different.

Table 17: OA and Kappa index for Single class and Multiclass analysis. Results for Hanover and Campo Verde Seq-1 and Seq-2.

	Hanover		Seq-1		Seq-2	
	OA	Kappa	OA	Kappa	OA	Kappa
Single class	94.7	93.5	88.0	74.2	83.3	75.4
Multiclass	94.8	93.6	86.8	72.7	83.7	76.0

In addition, on the Hanover dataset, the multiclass Protocol II presented better results in epochs outside the crop cycle period, in which the crop label was replaced by the label *soil*. Notice that on images between May and July lower accuracy values were achieved than on images from October to April and from August to September.

5.4.3 Evaluation of post-processing algorithms

As described in chapter 4, the MLCS post-processing algorithms were conceived to refine the results produced by any of the methods evaluated in the preceding section by exploiting prior knowledge about crop dynamics.

The Hanover dataset does not contain crop rotation and temporal displacements of crop circles occur rarely. Thus, the assessment of the post-processing strategies will be limited on Campo Verde dataset. Since CNN-PC was the best performing method in most cases, we limit ourselves to present the results of the post-processing algorithms for CNN-PC only.

Figure 53 and Figure 54 show the improvements in F1 per class brought by both post-processing algorithms in different epochs. For the first 4 epochs (Figure 53) improvements ranged from ∼5% to ∼7% for *maize*, *turfgrass*, *cotton*, cerrado and *NCC*. For epochs 5 onwards (Figure 54), even higher values were achieved for *maize*, *soybeans*, and *beans*, that is, from ∼10% to ∼21%. Notice that, MLCS-SL outperformed or stayed nearly equal to MLCS-CR for almost all crops.

Maize is a good example of MLCS-SL post-processing benefits when there is a temporal displacement in the crop cycle. In December *maize* is at the end of its life cycle in some parcels, whereas it is at the beginning of its vegetative growth in other parcels. A similar phenomenon occurs for *NCC*. In January, *NCC* is at the end of its vegetation season in some parcels, whereas it is continuing its vegetation season in other parcels. This explains the better performance presented by MLCS-SL compared to MLCS-CR.

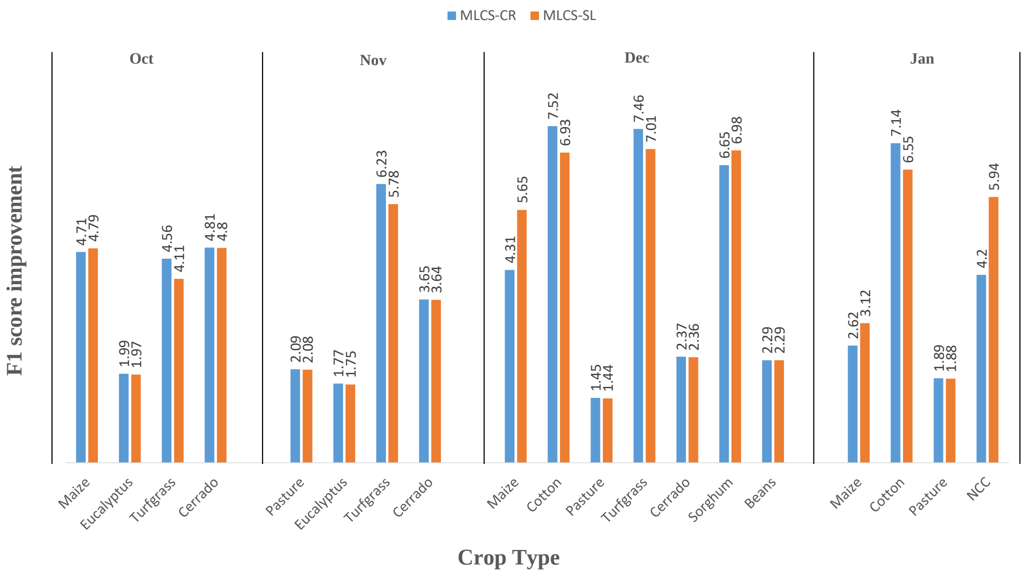


Figure 53: F1 score improvements for MLCS-CR and MLCS-SL. Images from October to January.

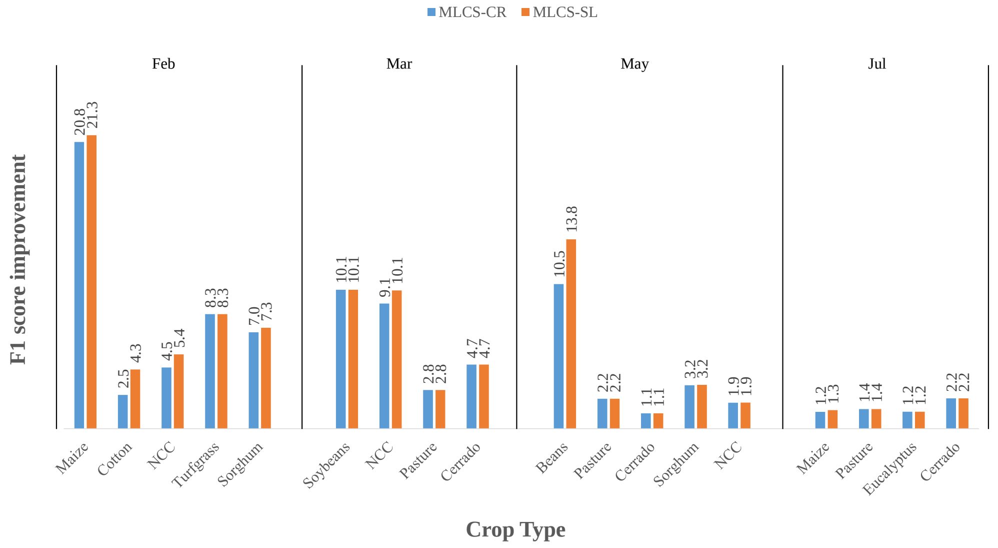


Figure 54: F1 score improvements for MLCS-CR and MLCS-SL. Images from February to July.

5.4.4 Summary and Discussion

The results collected in our experiments confirmed that the accuracy improves as more multitemporal data are added to the data set. However, the gain tends to stabilize after some sequence length. As expected, the results for the temperate region were superior to what has been obtained on the dataset of a tropical region. The favorable climate conditions allow more flexibility in the land use, which implies in a more complex dynamics in tropical region, and consequently, makes the classification task more challenging. Furthermore, the analysis indicated that the DL techniques yielded better results than the standard RF approach in almost all experiments.

The results also confirmed the high potential of FCN for crop recognition, and the need to conceive a strategy to handle databases with complex crop dynamics and unbalanced classes.

In addition, the approaches tested in this study can be easily applied in temperate as well as in tropical regions, both in their single or multiclass variants. For the data set of a temperate region the multiclass approach brought no significant gain in relation to the single class per season approach. However, the multiclass protocol achieved better results when dealing with data from a tropical region.

Finally, the proposed post-processing algorithms, which exploit prior knowledge, brought significant gains when working with multiclass approaches. The algorithm brought a F1 score increase in almost all classes and a reduction in the number of sequences to be evaluated from 384 (CNN-PC) to 44 and 21 for MLCS-CR and MLCS-SL, respectively.

6
CONCLUSIONS AND FUTURE WORKS

In this work three Deep Learning (DL) based approaches for crop recognition from multitemporal SAR image sequences were investigated: Autoencoders (AE), Convolutional Neural Networks (CNN), and Fully Convolutional Networks (FCN). The AE method combines unsupervised feature learning with a Random Forest (RF) classifier in a pixel-wise analysis. The CNN method uses a three-layer network for supervised patch-wise classification with spatially independent predictions. Finally, the FCN methods investigated in this work implement a full patch semantic segmentation with structured predictions. As baseline we took a RF classifier running upon hand-crafted textural features.

The DL based methods performed better than the baseline in almost all experiments. In fact, AE provided the best results on the dataset of the temperate region. It achieved the best or close to the best performance for each metrics in all experiments. On the other hand, for the dataset of a tropical region the CNN patch-based approach alternated with RF and FCN, having reached the best or close to the best accuracy in most experiments. Moreover, the CNN approach presents a more stable behavior when compared with FCN.

Although the FCNs have performed well, their full potential was not fully exploited in our experiments, mainly due to the difficulty in balancing the number of training samples among the crop types.

Finally, the post-processing strategies were able to incorporate prior-knowledge about crop rotations and temporal displacements. Indeed, the post processing improved accuracy in term of F1 score in almost all classes, reached from 10% to 20% for some crops.

As future works we plan to investigate procedures to handle class unbalance, specially for the FCN approaches, and the inclusion of a data augmentation strategy. In addition, it is to remark that these methods are not tailored to SAR data, and could be straightforwardly applied to optical data. In the continuation of this research we intend to extend the methods to exploit multisensor data.

Bibliography

1 OTUKEI, J. R.; BLASCHKE, T.. **Land cover change assessment using decision trees, support vector machines and maximum likelihood classification algorithms**. International Journal of Applied Earth Observation and Geoinformation, 12:S27–S31, 2010.

2 MUÑOZ-MARÍ, J.; BRUZZONE, L. ; CAMPS-VALLS, G.. **A support vector domain description approach to supervised classification of remote sensing images**. IEEE Transactions on Geoscience and Remote Sensing, 45(8):2683–2692, 2007.

3 XU, M.; WATANACHATURAPORN, P.; VARSHNEY, P. K. ; ARORA, M. K.. **Decision tree regression for soft classification of remote sensing data**. Remote Sensing of Environment, 97(3):322–336, 2005.

4 BLASCHKE, T.. **Object based image analysis for remote sensing**. ISPRS journal of photogrammetry and remote sensing, 65(1):2–16, 2010.

5 HUA, B.; FU-LONG, M. ; LI-CHENG, J.. **Research on computation of glcm of image texture [j]**. Acta Electronica Sinica, 1(1):155–158, 2006.

6 MELGANI, F.; SERPICO, S. B.. **A markov random field approach to spatio-temporal contextual image classification**. IEEE Transactions on Geoscience and Remote Sensing, 41(11):2478–2487, 2003.

7 HOBERG, T.; ROTTENSTEINER, F. ; HEIPKE, C.. **Classification of multitemporal remote sensing data of different resolution using Conditional Random Fields**. IEEE International Conference on Computer Vision (ICCV) Workshops, p. 235–242, 2011.

8 GEOD, F.; BAU, F.; UNIVERSIT, U. T. ; ISBN, D.. **Spatial-temporal Dynamic Conditional Random Fields crop type mapping using**. PhD thesis, 2016.

9 SIACHALOU, S.; MALLINIS, G. ; TSAKIRI-STRATI, M.. **A hidden markov models approach for crop classification: Linking crop phenology to time series of multi-sensor remote sensing data**. Remote Sensing, 7(4):3633–3650, 2015.

10 LEITE, P. B. C.; FEITOSA, R. Q.; FORMAGGIO, A. R.; DA COSTA, G. A. O. P.; PAKZAD, K. ; SANCHES, I. D. A.. **Hidden Markov Models for crop recognition in remote sensing image sequences**. In: PATTERN RECOGNITION LETTERS, volumen 32, p. 19–26, 2011.

11 ROMERO, A.; GATTA, C. ; CAMPS-VALLS, G.. **Unsupervised deep feature extraction for remote sensing image classification**. IEEE Transactions on Geoscience and Remote Sensing, 54(3):1349–1362, 2016.

12 FAUVEL, M.; TARABALKA, Y.; BENEDIKTSSON, J. A.; CHANUSSOT, J. ; TILTON, J. C.. **Advances in spectral-spatial classification of hyperspectral images**. Proceedings of the IEEE, 101(3):652–675, 2013.

13 ZABALZA, J.; REN, J.; ZHENG, J.; ZHAO, H.; QING, C.; YANG, Z.; DU, P. ; MARSHALL, S.. **Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging**. Neurocomputing, 185:1–10, 2016.

14 KUSSUL, N.; LEMOINE, G.; GALLEGO, F. J.; SKAKUN, S. V.; LAVRENIUK, M. ; SHELESTOV, A. Y.. **Parcel-Based Crop Classification in Ukraine Using Landsat-8 Data and Sentinel-1A Data**. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 9(6):2500–2508, 2016.

15 HAO, P.; ZHAI, J. H. ; ZHANG, S. F.. **A simple and effective method for image classification**. In: 2017 INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND CYBERNETICS (ICMLC), volumen 1, p. 230–235, July 2017.

16 VOLPI, M.; TUIA, D.. **Dense semantic labeling of subdecimeter resolution images with convolutional neural networks**. IEEE Transactions on Geoscience and Remote Sensing, 55(2):881–893, 2017.

17 WALDHOFF, G.; CURDT, C.; HOFFMEISTER, D. ; BARETH, G.. **Analysis of Multitemporal and Multisensor Remote Sensing Data for Crop Rotation Mapping**. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, I-7(September):177–182, 2012.

18 KUSSUL, N.; SKAKUN, S.; SHELESTOV, A. ; KUSSUL, O.. **THE USE OF SATELLITE SAR IMAGERY TO CROP CLASSIFICATION IN UKRAINE WITHIN JECAM PROJECT Space Research Institute NAS Ukraine and SSA Ukraine; National Technical**

University of Ukraine " Kyiv Polytechnic Institute "; National University of Life and Environ. Igarss 2014, p. 1497–1500, 2014.

19 SKAKUN, S.; KUSSUL, N.; SHELESTOV, A. Y.; LAVRENIUK, M. ; KUSSUL, O.. **Efficiency Assessment of Multitemporal C-Band Radarsat-2 Intensity and Landsat-8 Surface Reflectance Satellite Imagery for Crop Classification in Ukraine.** IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 9(8):3712–3719, 2016.

20 KENDUIYWO, B.; BARGIEL, D. ; SOERGEL, U.. **Crop Type Mapping From a Sequence of Terrasar-X Images With Dynamic Conditional Random Fields.** ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci., III-7(2011):59–66, 2016.

21 KENDUIYWO, B. K.; BARGIEL, D. ; SOERGEL, U.. **Higher Order Dynamic Conditional Random Fields Ensemble for Crop Type Classification in Radar Images.** IEEE Transactions on Geoscience and Remote Sensing, 55(8):4638–4654, 2017.

22 INGLADA, J.; VINCENT, A.; ARIAS, M. ; MARAIS-SICRE, C.. **Improved early crop type identification by joint use of high temporal resolution sar and optical image time series.** Remote Sensing, 8(5), 2016.

23 SANCHES, I. D.; FEITOSA, R. Q.; DIAZ, P. M. A.; SOARES, M. D.; LUIZ, A. J. B.; SCHULTZ, B. ; MAURANO, L. E. P.. **Campo verde database: Seeking to improve agricultural remote sensing of tropical areas.** IEEE Geoscience and Remote Sensing Letters, PP(99):1–5, 2018.

24 PAL, M.. **Random forest classifier for remote sensing classification.** International Journal of Remote Sensing, 26(1):217–222, 2005.

25 MELGANI, F.; BRUZZONE, L.. **Classification of hyperspectral remote sensing images with support vector machines.** IEEE Transactions on geoscience and remote sensing, 42(8):1778–1790, 2004.

26 ATKINSON, P. M.; TATNALL, A.. **Introduction neural networks in remote sensing.** International Journal of remote sensing, 18(4):699–709, 1997.

27 INGLADA, J.; ARIAS, M.; TARDY, B.; MORIN, D.; VALERO, S.; HAGOLLE, O.; DEDIEU, G.; SEPULCRE, G.; BONTEMPS, S. ; DEFOURNY, P.. **Benchmarking of algorithms for crop type land-cover maps using**

Sentinel-2 image time series. International Geoscience and Remote Sensing Symposium (IGARSS), 2015-Novem:3993–3996, 2015.

28 NITZE, I.; SCHULTHESS, U. ; ASCHE, H.. **Comparison of machine learning algorithms random forest, artificial neuronal network and support vector machine to maximum likelihood for supervised crop type classification**. Proceedings of the 4th Conference on GEographic Object-Based Image Analysis – GEOBIA 2012, p. 35–40, 2012.

29 VAN ZYL, J. J.. **Unsupervised classification of scattering behavior using radar polarimetry data**. IEEE Transactions on Geoscience and Remote Sensing, 27(1):36–45, 1989.

30 FERRO-FAMIL, L.; POTTIER, E. ; LEE, J.-S.. **Unsupervised classification of multifrequency and fully polarimetric sar images based on the h/a/alpha-wishart classifier**. IEEE Transactions on Geoscience and Remote Sensing, 39(11):2332–2342, 2001.

31 CELIK, T.. **Unsupervised change detection in satellite images using principal component analysis and k-means clustering**. IEEE Geoscience and Remote Sensing Letters, 6(4):772–776, 2009.

32 GONCALVES, M.; NETTO, M.; COSTA, J. ; ZULLO JUNIOR, J.. **An unsupervised method of classifying remotely sensed images using kohonen self-organizing maps and agglomerative hierarchical clustering methods**. International Journal of Remote Sensing, 29(11):3171–3207, 2008.

33 LILLESAND, T.; KIEFER, R. W. ; CHIPMAN, J.. **Remote sensing and image interpretation**. John Wiley & Sons, 2014.

34 MCROBERTS, R. E.; NELSON, M. D. ; WENDT, D. G.. **Stratified estimation of forest area using satellite imagery, inventory data, and the k-nearest neighbors technique**. Remote Sensing of Environment, 82(2-3):457–468, 2002.

35 RUIZ, L.; FDEZ-SARRÍA, A. ; RECIO, J.. **Texture feature extraction for classification of remote sensing data using wavelet decomposition: a comparative study**. In: 20TH ISPRS CONGRESS, volumen 35, p. 1109–1114, 2004.

36 LUCIEER, A.; STEIN, A. ; FISHER, P.. **Multivariate texture-based segmentation of remotely sensed imagery for extraction of ob-**

jects and their uncertainty. International Journal of Remote Sensing, 26(14):2917–2936, 2005.

37 HARALICK, R. M.; SHANMUGAM, K. ; OTHERS. **Textural features for image classification**. IEEE Transactions on systems, man, and cybernetics, (6):610–621, 1973.

38 HARALICK, R. M.. **Statistical and structural approaches to texture**. Proceedings of the IEEE, 67(5):786–804, 1979.

39 ZHANG, X.; SUN, Y.; SHANG, K.; ZHANG, L. ; WANG, S.. **Crop Classification Based on Feature Band Set Construction and Object-Oriented Approach Using Hyperspectral Images**. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 9(9):4117–4128, 2016.

40 HE, D.-C.; WANG, L.. **Texture unit, texture spectrum, and texture analysis**. IEEE transactions on Geoscience and Remote Sensing, 28(4):509–512, 1990.

41 WALTER, V.. **Object-based classification of remote sensing data for change detection**. ISPRS Journal of photogrammetry and remote sensing, 58(3-4):225–238, 2004.

42 MYINT, S. W.; GOBER, P.; BRAZEL, A.; GROSSMAN-CLARKE, S. ; WENG, Q.. **Per-pixel vs. object-based classification of urban land cover extraction using high spatial resolution imagery**. Remote sensing of environment, 115(5):1145–1161, 2011.

43 JIAO, X.; KOVACS, J. M.; SHANG, J.; MCNAIRN, H.; WALTERS, D.; MA, B. ; GENG, X.. **Object-oriented crop mapping and monitoring using multi-temporal polarimetric radarsat-2 data**. ISPRS Journal of Photogrammetry and Remote Sensing, 96:38–46, 2014.

44 DURO, D. C.; FRANKLIN, S. E. ; DUBÉ, M. G.. **A comparison of pixel-based and object-based image analysis with selected machine learning algorithms for the classification of agricultural landscapes using spot-5 hrg imagery**. Remote Sensing of Environment, 118:259–272, 2012.

45 LU, D.; WENG, Q.. **A survey of image classification methods and techniques for improving classification performance**. International journal of Remote sensing, 28(5):823–870, 2007.

46 LIU, D.; SONG, K.; TOWNSHEND, J. R. ; GONG, P.. **Using local transition probability models in markov random fields for forest change detection**. Remote Sensing of Environment, 112(5):2222–2231, 2008.

47 HOBERG, T.; MÜLLER, S.. **Multitemporal crop type classification using conditional random fields and rapideye data**. In: INTERNATIONAL ARCHIVES OF THE PHOTOGRAMMETRY, REMOTE SENSING AND SPATIAL INFORMATION SCIENCES:[ISPRS HANNOVER WORKSHOP 2011: HIGH-RESOLUTION EARTH IMAGING FOR GEOSPATIAL INFORMATION] 38-4 (2011), NR. W19, volumen 38, p. 115–121. Göttingen: Copernicus GmbH, 2011.

48 SHEN, Y.; WU, L.; DI, L.; YU, G.; TANG, H.; YU, G. ; SHAO, Y.. **Hidden markov models for real-time estimation of corn progress stages using modis and meteorological data**. Remote Sensing, 5(4):1734–1753, 2013.

49 LI, T.; ZHANG, J. ; ZHANG, Y.. **Classification of hyperspectral image based on deep belief networks**. In: IMAGE PROCESSING (ICIP), 2014 IEEE INTERNATIONAL CONFERENCE ON, p. 5132–5136. IEEE, 2014.

50 LV, Q.; DOU, Y.; NIU, X.; XU, J.; XU, J. ; XIA, F.. **Urban land use and land cover classification using remotely sensed sar data through deep belief networks**. Journal of Sensors, 2015, 2015.

51 CHEN, Y.; ZHAO, X. ; JIA, X.. **Spectral–spatial classification of hyperspectral data based on deep belief network**. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 8(6):2381–2392, 2015.

52 LV, Q.; DOU, Y.; NIU, X.; XU, J. ; LI, B.. **Classification of land cover based on deep belief networks using polarimetric radarsat-2 data**. In: GEOSCIENCE AND REMOTE SENSING SYMPOSIUM (IGARSS), 2014 IEEE INTERNATIONAL, p. 4679–4682. IEEE, 2014.

53 TAO, C.; PAN, H.; LI, Y. ; ZOU, Z.. **Unsupervised spectral–spatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification**. IEEE Geoscience and remote sensing letters, 12(12):2438–2442, 2015.

54 CHEN, Y.; LIN, Z.; ZHAO, X.; WANG, G. ; GU, Y.. **Deep Learning-Based Classification of Hyperspectral Data**. IEEE JOURNAL OF SELECTED

TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, p. 2094–2107, 2014.

55 LECUN, Y.; BOTTOU, L.; BENGIO, Y. ; HAFFNER, P.. **Gradient-based learning applied to document recognition**. Proceedings of the IEEE, 86(11):2278–2324, 1998.

56 MAKANTASIS, K.; KARANTZALOS, K.; DOULAMIS, A. ; DOULAMIS, N.. **Deep supervised learning for hyperspectral data classification through convolutional neural networks**. In: GEOSCIENCE AND REMOTE SENSING SYMPOSIUM (IGARSS), 2015 IEEE INTERNATIONAL, p. 4959–4962. IEEE, 2015.

57 TRAN, D.; BOURDEV, L.; FERGUS, R.; TORRESANI, L. ; PALURI, M.. **Learning spatiotemporal features with 3d convolutional networks**. In: COMPUTER VISION (ICCV), 2015 IEEE INTERNATIONAL CONFERENCE ON, p. 4489–4497. IEEE, 2015.

58 LI, Y.; ZHANG, H. ; SHEN, Q.. **Spectral–spatial classification of hyperspectral imagery with 3d convolutional neural network**. Remote Sensing, 9(1):67, 2017.

59 JI, S.; ZHANG, C.; XU, A.; SHI, Y. ; DUAN, Y.. **3d convolutional neural networks for crop classification with multi-temporal remote sensing images**. Remote Sensing, 10(1), 2018.

60 MOU, L.; GHAMISI, P. ; ZHU, X. X.. **Fully conv-deconv network for unsupervised spectral-spatial feature extraction of hyperspectral imagery via residual learning**. In: GEOSCIENCE AND REMOTE SENSING SYMPOSIUM (IGARSS), 2017 IEEE INTERNATIONAL, p. 5181–5184. IEEE, 2017.

61 MARMANIS, D.; WEGNER, J. D.; GALLIANI, S.; SCHINDLER, K.; DATCU, M. ; STILLA, U.. **Semantic segmentation of aerial images with an ensemble of cnns**. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 3:473, 2016.

62 RYERSON, R.; HENDERSON, F.; LEWIS, A.; FOR PHOTOGRAMMETRY, A. S. ; SENSING, R.. **Manual of Remote Sensing, Principles and Applications of Imaging Radar**. Manual of Remote Sensing - Third Edition. Wiley, 1998.

63 CURLANDER, J.; MCDONOUGH, R.. **Synthetic Aperture Radar: Systems and Signal Processing**. Wiley Series in Remote Sensing and Image Processing. Wiley, 1991.

64 ULABY, F.; BARE, J.. **Look direction modulation function of the radar backscattering coefficient of agricultural fields**. Photogrammetric Engineering and Remote Sensing, 45(11):1495–1506, 1 1979.

65 BRISCO, B.; BROWN, R. J.; SNIDER, B.; SOFKO, G. J.; KOEHLER, J. A. ; WACKER, A. G.. **Tillage effects on the radar backscattering coefficient of grain stubble fields**. International Journal of Remote Sensing, 12(11):2283–2298, 1991.

66 BRISCO, B.; BROWN, R.; GAIRNS, J. ; SNIDER, B.. **Temporal ground-based scatterometer observations of crops in western canada**. Canadian Journal of Remote Sensing, 18(1):14–21, 1992.

67 HO, T. K.. **Random decision forests**. In: PROCEEDINGS OF 3RD INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION, volumen 1, p. 278–282 vol.1, Aug 1995.

68 HO, T. K.. **The random subspace method for constructing decision forests**. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(8):832–844, Aug 1998.

69 BREIMAN, L.. **Random forests**. Machine Learning, 45(1):5–32, 2001.

70 BREIMAN, L.. **Bagging predictors**. Machine Learning, 45(2):123–140, 1994.

71 AMIT, Y.; GEMAN, D.. **Shape quantization and recognition with randomized trees**. Neural Comput., 9(7):1545–1588, Oct. 1997.

72 HINTON, G. E.; ZEMEL, R. S.. **Autoencoders, minimum description length and helmholtz free energy**. In: Cowan, J. D.; Tesauro, G. ; Alspector, J., editors, ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 6, p. 3–10. Morgan-Kaufmann, 1994.

73 BOURLARD, H.; KAMP, Y.. **Auto-association by multilayer perceptrons and singular value decomposition**. Biological cybernetics, 59(4-5):291–294, 1988.

74 LIPPMANN, R.. **An introduction to computing with neural nets**. IEEE ASSP Magazine, 4(2):4–22, Apr 1987.

75 COURVILLE, I. G.; BENGIO, Y. ; COURVILLE, A.. **Deep Learning**. 2016.

76 NG, A.. **CS294A Lecture Notes Sparse Autoencoder**. Cs294a, p. 1–19, 2011.

77 DING, C.; ZHOU, D.; HE, X. ; ZHA, H.. **R 1-pca: rotational invariant l 1-norm principal component analysis for robust subspace factorization**. In: PROCEEDINGS OF THE 23RD INTERNATIONAL CONFERENCE ON MACHINE LEARNING, p. 281–288. ACM, 2006.

78 LEE, H.; BATTLE, A.; RAINA, R. ; NG, A. Y.. **Efficient sparse coding algorithms**. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, p. 801–808, 2007.

79 FUKUSHIMA, K.; MIYAKE, S.. **Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition**. In: COMPETITION AND COOPERATION IN NEURAL NETS, p. 267–285. Springer, 1982.

80 LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W. ; JACKEL, L. D.. **Backpropagation applied to handwritten zip code recognition**. Neural computation, 1(4):541–551, 1989.

81 DUMOULIN, V.; VISIN, F.. **A guide to convolution arithmetic for deep learning**. arXiv preprint arXiv:1603.07285, 2016.

82 NAIR, V.; HINTON, G. E.. **Rectified linear units improve restricted boltzmann machines**. In: PROCEEDINGS OF THE 27TH INTERNATIONAL CONFERENCE ON MACHINE LEARNING (ICML-10), p. 807–814, 2010.

83 LONG, J.; SHELHAMER, E. ; DARRELL, T.. **Fully convolutional networks for semantic segmentation**. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June:3431–3440, 2015.

84 ZEILER, M. D.; KRISHNAN, D.; TAYLOR, G. W. ; FERGUS, R.. **Deconvolutional networks**. In: IN CVPR, 2010.

85 NOH, H.; HONG, S. ; HAN, B.. **Learning deconvolution network for semantic segmentation**. In: PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION, p. 1520–1528, 2015.

86 IOFFE, S.; SZEGEDY, C.. **Batch normalization: Accelerating deep network training by reducing internal covariate shift**. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, p. 448–456, 2015.

87 SHIMODAIRA, H.. **Improving predictive inference under covariate shift by weighting the log-likelihood function**, 2000.

88 SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I. ; SALAKHUTDINOV, R.. **Dropout: A simple way to prevent neural networks from overfitting**. The Journal of Machine Learning Research, 15(1):1929–1958, 2014.

89 HINTON, G. E.; SRIVASTAVA, N.; KRIZHEVSKY, A.; SUTSKEVER, I. ; SALAKHUTDINOV, R. R.. **Improving neural networks by preventing co-adaptation of feature detectors**. arXiv preprint arXiv:1207.0580, 2012.

90 HUANG, G.; LIU, Z.; WEINBERGER, K. Q. ; VAN DER MAATEN, L.. **Densely Connected Convolutional Networks**. 2016.

91 ZHANG, F.; DU, B.; ZHANG, L. ; ZHANG, L.. **Hierarchical feature learning with dropout k-means for hyperspectral image classification**. Neurocomputing, 187:75 – 82, 2016. Recent Developments on Deep Big Vision.

92 LAFFERTY, J.; MCCALLUM, A. ; PEREIRA, F. C.. **Conditional random fields: Probabilistic models for segmenting and labeling sequence data**. 2001.

93 JEGOU, S.; DROZDZAL, M.; VAZQUEZ, D.; ROMERO, A. ; BENGIO, Y.. **The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation**. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2017-July:1175–1183, 2017.

94 BARGIEL, D.. **Remote Sensing of Environment A new method for crop classification combining time series of radar images and crop phenology information**. Remote Sensing of Environment, 198:369–383, 2017.

95 CASTRO, J. D. B.; FEITOZA, R. Q.; ROSA, L. C. L.; DIAZ, P. M. A. ; SANCHES, I. D. A.. **A Comparative Analysis of Deep Learning**

Techniques for Sub-Tropical Crop Types Recognition from Multitemporal Optical/SAR Image Sequences. 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), p. 382–389, 2017.

96 CONGALTON, R. G.; GREEN, K.. **Assessing the accuracy of remotely sensed data: principles and practices.** CRC press, 2008.

97 DUCHI, J.; HAZAN, E. ; SINGER, Y.. **Adaptive subgradient methods for online learning and stochastic optimization.** Journal of Machine Learning Research, 12(Jul):2121–2159, 2011.

98 ZEILER, M. D.. **Adadelta: an adaptive learning rate method.** arXiv preprint arXiv:1212.5701, 2012.