UNIVERSITY OF AMSTERDAM

MSc Artificial Intelligence
Master Thesis

Convolutional Neural Networks for Crop Yield Prediction using Satellite Images

by

Helena Russello

Student Number 11138971

29th June 2018

36 Ects
October 2017 - June 2018

Supervisors:
Wenling Shang MSc
Dr. Santiago Gaitan

Assessor:
Dr. Efstratios Gavves

IBM Center for Advanced Studies

ABSTRACT

Crop yield forecasting during the growing season is useful for farming planning and management practices as well as for planning humanitarian aid in developing countries. Common approaches to yield forecast include the use of expensive manual surveys or accessible remote sensing data. Traditional remote sensing based approaches to predict crop yield consist of classical Machine Learning techniques such as Support Vector Machines and Decision Trees. More recent approaches include using deep neural network models, such as CNN and LSTM. We identify the additional gaps in the literature of existing machine learning methods as lacking of (1) standardized training protocol that specifies the optimal time frame, both in terms of years and months of each year, to be considered in the training set, (2) verified applicability to developing countries under the condition of scarce data, and (3) effective utilization of spatial features in remote sensing images. In this thesis, we first replicate the state-of-the-art approach of You et al. [1], in particular their CNN model for crop yield prediction. To tackle the first identified gap, we then perform control experiments to determine the best temporal training settings for soybean yield prediction. To probe the second gap, we further investigate whether this CNN model could be trained on source locations and then be transfered to new target locations and conclude that it is necessary to use source regions that have a similar or generalizable ecosystem to the target regions. This allows us to assess the transferability of CNN-based regression models to developing countries, where little training data is available. Additionally, we propose a novel 3D CNN model for crop yield prediction task that leverages the spatiotemporal features. We demonstrate that our 3D CNN outperforms all competing machine learning methods, shedding light on promising future directions in utilizing deep learning tools for crop yield prediction.

ACKNOWLEDGMENTS

First, I would like to thank my supervisors Wenling Shang and Dr. Santiago Gaitan for their guidance and support. Their precious knowledge, advice and comments steered me in the right direction. Moreover, I thank Dr. Efstratios Gavves for agreeing to be part of my defense committee.

I would like to thank my colleagues at the IBM Center for Advanced Studies for the stimulating discussions, and the MakerLab team for their delicate music selection, bringing rhythm to my days at the office.

I would also like to thank Iva for being my friend during the Masters degree and for proofreading this work.

Finally, I want to express my profound gratitude to Julius and Mum for their unconditional support and continuous encouragements throughout the degree and thesis.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

Ending hunger in the world is one of the top sustainable development goals of the United Nations (UN) [2]. A way to tackle the problem is to improve food supply at a global level. The global food demand increases as the population grows and impacts food prices and availability [3]. Crop yield forecasting can address the food supply problem. By being able to predict the crop yield during the growing season, market prices can be forecasted, import and export can be planned, the socio-economical impact of crop loss can be minimized and humanitarian food assistance can be planned [3, 4].

Crop yield can be forecasted by using manual surveys, crop simulation models or remote sensing data. Manual surveys are the traditional ways of predicting crop yield and require in-situ information about the crops e.g., counting the plants, assessing their health, assessing the damage from pest, etc. The yield is then forecasted by comparing the data to previous years observations, by means of regression tools or expert knowledge [5]. However, manual surveys are expensive and difficult to scale to other regions and countries [1]. Crop Simulation Models (CSM) simulate crop development throughout the growing season by means of mathematical models of soil properties, weather and management practices [6, 5]. Simple CSMs use climate and historical yield data to estimate crop yield over large areas [6, 5]. However, CSMs require large datasets for calibrating the model and may therefore not be a suitable approach in developing countries [5]. Lastly, remote sensing for crop yield prediction is often the preferred technique. Remote Sensing (RS) is defined as the science of observing an object without touching it [7]. RS data can be collected from satellites, airplanes, drones, or even from a simple camera through the windshield of a truck. Remote sensing has the ability to provide more affordable yield forecasting tools as large collections of free and open-source RS images are available [8].

RS for crop yield prediction has been widely studied over the last decades. Until recently, simple regression models [9, 10] combined with remotely sensed data and vegetation indices such as Normalized Difference Vegetation Index (NDVI) [9, 10, 11], Enhanced Vegetation Index (EVI) [9, 12, 11] and Normalized Difference Water Index (NDWI) [9] have been used to predict crop yield. Vegetation indices usually include

the information of only 2 or 3 spectral bands. These models involve subsequent processing and feature engineering.

With the recent advances in Machine Learning (ML), researchers started applying ML techniques to multispectral satellite images for crop yield prediction. These techniques include Support Vector Machine (SVM) [11, 12], Decision Trees [11], Multi-Layer Perceptron (MLP) [11] and Restricted Boltzmann Machine (RBM) [12]. These methods lead to an overall improvement of prediction accuracy. In 2017, Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) networks were used for the first time for crop yield prediction, outperforming all the competing approaches [1]. Most of the listed ML and Deep Learning (DL) approaches allow to feed the processed images almost directly to the model, without requiring much feature engineering. Furthermore, by using multispectral satellite images, bands that are traditionally not used in RS-based crop yield prediction can provide more information about crops, vegetation, soil and temperature.

Each of the listed approaches uses different observation intervals and training years. There has not been any study, to our knowledge, which systematically determined the best temporal training settings for soybean yield prediction. Intuitively, finding the period that captures the evolution of specific crops the best could improve the performance of crop yield prediction models. Hence, our first research question:

RQ1 *What are the best temporal training settings for soybean yield prediction?*

The lack of training data in developing countries make it infeasible to train a neural network for predicting crop yield in these regions. However, all the presented models are designed for predicting yield in specific regions, and whether RS models trained on given locations can be transfered to different geographical locations remains an open question [13, 10]. Hence, our second research question:

RQ2 *Can a RS model trained on specific locations be generalized to new locations?*

Finally, the listed approaches focus on learning temporal features (i.e., identifying changes throughout the growing season) but fail to leverage the spatial dimension of remotely-sensed images. By looking at the properties of the areas surrounding croplands, one could derive information about the crops' environment, such as soil properties and elevation and potentially increase the accuracy of yield predictions. Hence, our third research question:

RQ3 *Can we leverage both the spatial and temporal dimension of remotely sensed images for better crop yield predictions, e.g. through 3D CNN?*

In this thesis, we focus on soybean yield prediction in the United States (U.S.), using

Moderate-Resolution Imaging Spectroradiometer (MODIS) satellite images. We answer the three research questions (RQ) to tackle the identified gaps in remote sensing-based crop yield prediction. Our first step is to replicate part of the work of You et al. [1], by using 2D CNNs to predict soybean yield in the U.S. using remotely-sensed multispectral images from the MODIS satellite mission. We train the model using the same methodology and use it as a baseline. We then use the baseline to perform time control experiments and location control experiments to answer RQ1 and RQ2 respectively. Finally, to answer RQ3, we create a novel way of sampling multispectral RS images to a neural network and design a 3D CNN architecture for spatiotemporal feature learning.

THESIS OUTLINE

The thesis is organized as follows:

- In Chapter 2, we provide background information about remote sensing, soybean phenology and Convolutional Neural Networks.
- Chapter 3 reviews previous work on crop yield prediction with RS.
- Chapter 4 presents the dataset and a guideline to satellite images preprocessing.
- In Chapter 5, we describe one of the methods used by You et al.[1], the histogram CNN. We then present the baseline replication results along with analysis.
- Chapter 6 describes the approach taken to perform the control experiments that answer RQ1 and RQ2.
- In Chapter 7, we answer RQ3 by proposing a 3D CNN and evaluate its empirical performance.
- The thesis ends with concluding remarks and future work directions in Chapter 8.

2

BACKGROUND

In this Chapter, we present background knowledge required to grasp the underlying concepts used in this thesis. In particular, we introduce remote sensing concepts for vegetation observation, soybean phenology and convolutional neural networks.

2.1 REMOTE SENSING AND SATELLITE IMAGERY

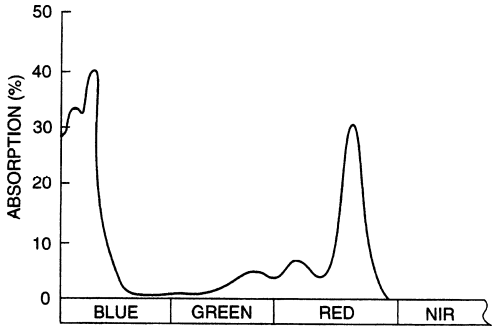
Remote sensing (RS) data is generated by sensors that record the electromagnetic radiation of physical objects such as buildings, roads, vegetation, soil or water. Physical objects have different spectral signatures, i.e. the emitted or reflected energy differs in a range of wavelengths [7]. RS data can be collected from satellites, airplanes, drones, or even from a simple camera through the windshield of a truck. In this thesis, we use satellite images.

Satellite images are widely used for agricultural applications. The reason of their success is due to large global and temporal availability and easy accessibility. Multiple satellite missions were launched by the National Aeronautics and Space Administration (NASA) and the European Space Agency (ESA), among others. In this work, we primarily leverage MODIS data. The MODIS instrument, carried by NASA's Terra and Aqua satellites, is capable of capturing data in 36 spectral bands [14].

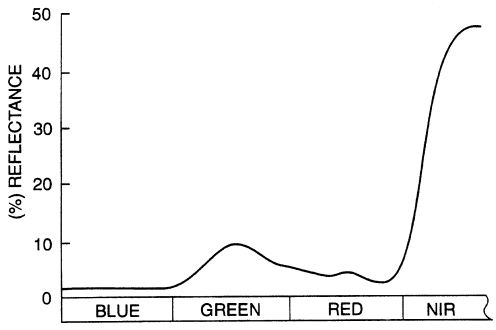
2.1.1 *Remote Sensing for Vegetation Observation*

RS finds applications in many disciplines, such as geology, geography, agriculture, urban planning, meteorology, climate change, and many more [7]. In this thesis, we focus on remote sensing for vegetation observation.

ELECTROMAGNETIC SPECTRUM. The visible and infrared spectrum are commonly used for vegetation observation. The visible spectrum, also called visible light, is a section of the electromagnetic spectrum where radiations are visible to the human eye (380 nm to 750 nm). Only the colors violet, blue, green, yellow, orange and red are present in the visible spectrum, and are called pure colors. As opposed to the visible light, infrared radiations are not visible to the human eye. The infrared spectrum



(a) Absorbtion spectrum of a leaf.



(b) Reflectance spectrum of a leaf.

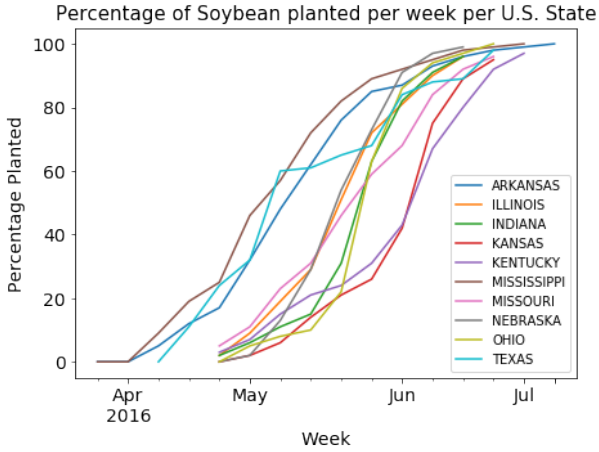
Figure 1: Absorption and reflectance of a healthy leaf. A healthy leaf typically absorbs blue and red light, and reflects green light and near-infrared waves [7]

includes wavelengths between 700 nm and 1 mm and is commonly divided into 5 regions: near, short-wavelength, mid-wavelength, long-wavelength and far infrared [7].

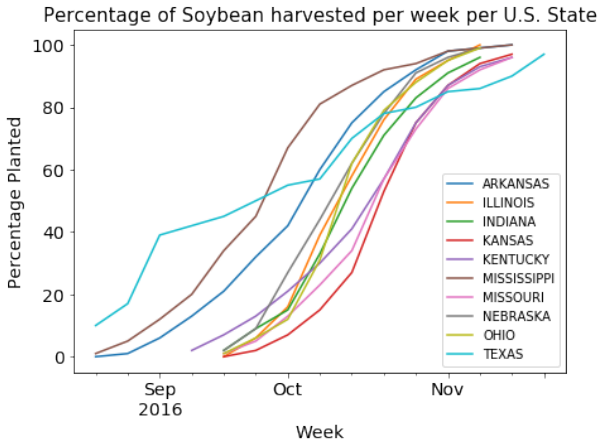
VEGETATION REFLECTANCE. Within the two spectra, we find different bands that provide information about the vegetation. The spectral properties of plants vary among species, but they all share a common pattern. Due to low spatial resolution, remote sensing images record the reflectance of the canopy rather than of individual plants. Therefore, the spectral properties of plants in RS can be generalized to the one of a typical leaf (Figure 1). The upper part of the leaf (upper epidermis) is rich in chloroplasts, which contain chlorophyll. During photosynthesis, chlorophyll molecules absorb blue and red light (Figure 1a), and reflect green light (Figure 1b). However, the upper epidermis is "transparent" to infrared light and lets it penetrate the internal part of the leaf (mesophyll tissue). In turn, the mesophyll tissue reflects the infrared energy, as indicated in Figure 1b. The spectral reflectance of the leaf changes as plants age, are subject to drought or are affected by diseases and pest infestation [7].

VEGETATION INDICES. Vegetation Indices (VI) take advantage of the spectral signature of the vegetation to measure biomass. By combining several spectral measurements, a VI indicates the amount or the health of vegetation in a pixel. The most commonly used vegetation index is the NDVI (Equation 1). It normalizes the difference between the red (R) and the near-infrared (NIR) signal, providing high values for healthy vegetation and low values for non-vegetated areas, as they don't display the same spectral response [7]. Other common vegetation indices include Enhanced Vegetation Index (EVI) or Leaf Area Index (LAI).

$$NDVI = \frac{NIR - R}{NIR + R} \tag{1}$$



(a) Weekly progress of soybean planting.



(b) Weekly progress of soybean harvesting.

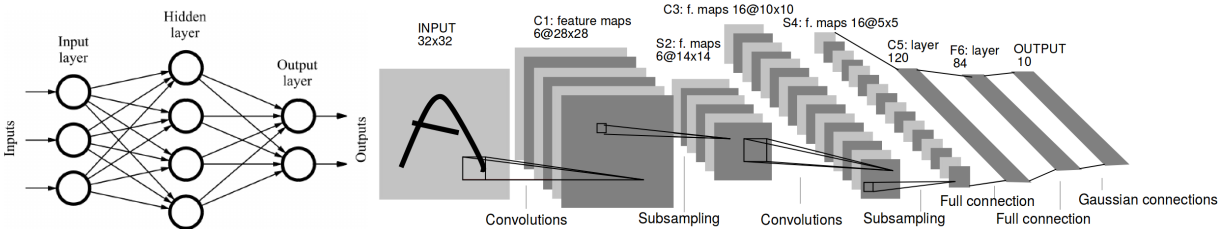
Figure 2: Progress of soybean planting and harvesting per state in 2016. The data was collected from USDA [17] and is plotted for a selection of states that have early and late planting dates.

2.2 SOYBEAN PHENOLOGY

Several factors influence the soybean crop phenology (or life cycle), namely planting date, day-length and temperature, among others [15]. The soybean growth is divided in two stages: vegetative stage and reproductive stage. The vegetative stage starts with the emergence of the plant above the soil surface and ends when the plant stops developing leaf nodes on the stem. The reproductive stage starts with blooming and ends with full maturity, i.e. when pods have reached their mature pod color [16]. Soybean crops are harvested when they reach full maturity [16]. Even though knowledge of the growth stages provides a general understanding of the soybean crop growth, the only factors that we consider in this work are the planting and harvesting dates.

The planting date is usually determined by day-length and soil temperature. Yields may decrease if soybean seeds are planted too early or too late. Soil temperatures that are too low or too high will impact the crop growth. Similarly, short day-length can lead to early blooming, preventing the plant from further growth [16]. J. Ruiz-Vega [15] recommends a soil temperature between 8 to 35°C and a day-length between 12 to 14 hours. Typically, soybean is planted from early April to late June depending on the region. Figure 2a shows the planting progress per U.S. state in 2016.

Soybean crops reach maturity when 95% of their pods achieve a mature pod color. Then, the pods are left to dry for 5 to 10 days until their moisture decreases to at least 15%. Thence, soybean seeds can be harvested. Assuming crops in the Northern hemisphere, and depending on the region, the harvesting period starts in early September and ends in late November. Figure 2b shows the planting progress per U.S. state in 2016.



(a) Example of a NN [19]

(b) Example of a 2D CNN [18]

Figure 3: Examples of a NN and a CNN.

2.3 CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Network (CNN) [18] is a category of neural networks used for tackling 2D or 3D structured data such as images and videos. In a typical neural network (NN), all the input units are connected to all the output units via a Fully-Connected (FC) layer (Figure 3a). However, in a CNN, each output unit is connected to only a subset of the entire input units via a convolutional layer. These subsets are known as receptive fields (Figure 3b).

In this section, we first introduce 2D and 3D CNNs. Then, we illustrate few components that aid the functionality of CNNs such as batch normalization and dropout.

2.3.1 *2D Convolutional Neural Network*

The 2D CNN is one of the most widely used feature extractor for images in the field of computer vision. CNNs are usually composed of convolutional layers, nonlinear activations such as Rectified Linear Unit (ReLU) and pooling layers.

CONVOLUTIONAL LAYER. A convolutional layer uses filters to perform convolutions on the input. Each layer learns from the previous layer and convolutional filters detect different types of features at different layer depths in the network. In the first layer, for example, the filters detect edges and colors; in the second layer, filters detect combination of edges, e.g. corners; and in the third layer, filters detect combination of corners, e.g. more complicated shape concepts like circles or squares [20].

Specifically, an input to a 2D convolutional layer is of dimension $c \times w \times h$, where c is the number of channels (e.g. 3 for an RGB image), w the width and h the height. For each 2D convolutional layer, there are n filters, each of kernel size $k \times k$, i.e. covering a receptive field of spatial dimension $k \times k$ and hence each filter is of dimension $c \times k \times k$. The convolutional filters are translated across the input with a stride s (i.e. every s

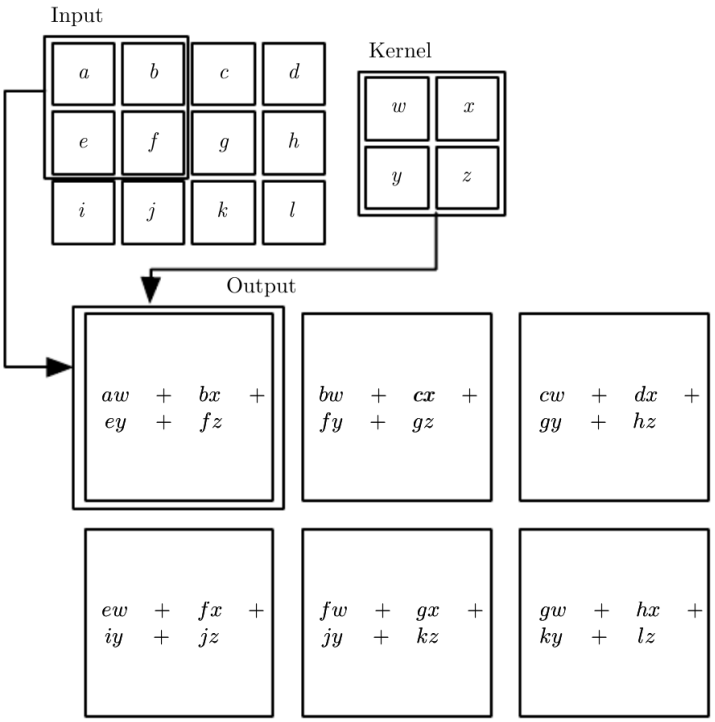


Figure 4: Example of a valid convolution with stride 1 on an input of size 4×3, kernel of size 2×2 [21].

pixel) and perform valid convolutions. Illustrated in Figure 4, the result S of a valid convolution of a kernel K on an input I is written as [21]:

$$S = I(x, y) * K = \sum_{i=1}^{k} \sum_{j=1}^{k} I(x + i, y + j) \cdot K(i, j) \tag{2}$$

The size of the resulting outputs is determined by the number of filters, the input size, kernel size, stride size and padding size, if any.

NON-LINEAR ACTIVATION. Non-linear activations are applied after convolutional layers to introduce non-linearity in the network. Today, the preferred non-linear activation function is the Rectified Linear Unit (ReLU) [22]. ReLU filters activations with negative values to zero, i.e. $ReLU(x) = \max(0, x)$. Comparing to sigmoid nonlinearity [23], ReLU has non-saturated outputs and this allows the gradient to flow better when training very deep networks. Leaky ReLU (LReLU) [24] improves the gradient flow of ReLU by setting a linear activation with a small positive slope even for negative values of x, i.e. $LReLU(x) = \alpha \max(0, x), x < 0$, enforcing non-zero gradient over its entire domain. An example of the effect of the ReLU on a feature map is shown in Figure 5.

MAX-POOLING LAYER. A pooling layer is often added after a ReLU. When performing convolutions with a small stride, the receptive fields overlap and the inform-

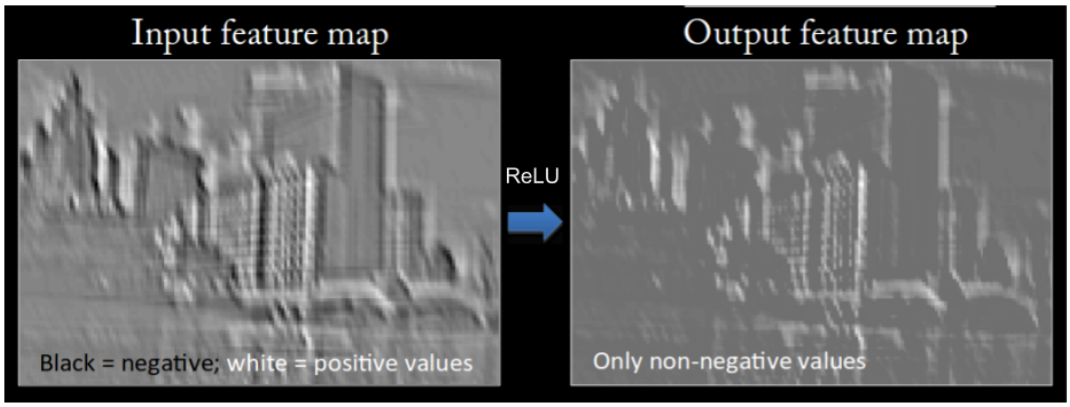


Figure 5: Example of the effect of a ReLU activation on a feature map [25].

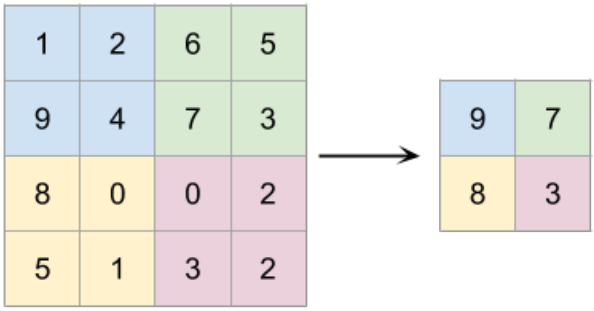


Figure 6: Example of a max-pooling operation with kernel and stride of size 2×2.

ation can become redundant. Pooling layers are then used to distill the most relevant information and downsample the spatial dimensionality. Additionally, pooling allows to produce translation invariant representations. The most commonly used pooling technique is max-pooling. Specifically, a small window (usually 2×2) with a stride of the same size is applied along the input and outputs the maximum value within the window. An example of max-pooling is shown in Figure 6.

BATCH NORMALIZATION. Batch normalization [26] layers are used to accelerate the optimization of very deep CNNs. Batch normalization reduces the covariate shift of the hidden values of each convolutional or fully-connected layer. The normalization of individual activation is done by subtracting the mean and dividing with the variance that are computed from each mini-batch.

2.3.2 *3D Convolutional Neural Network*

The 3D CNN inherits the same high-level concepts from 2D CNN. Figure 7 compares 2D and 3D convolutions. In 3D convolutions, the receptive field is not only along the two spatial dimensions as in a 2D convolutions but also along the temporal dimension. The same applies for 3D pooling. When using 3-dimensional inputs such as videos,

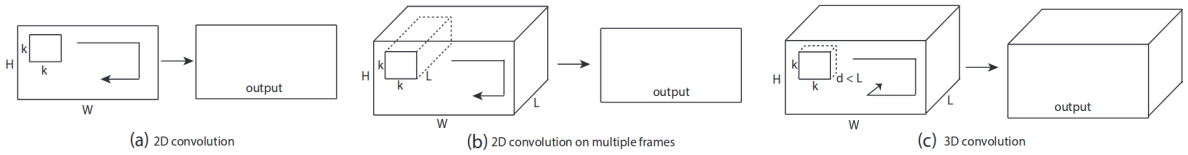


Figure 7: Comparison of 2D and 3D convolutions. (a) Applying 2D convolutions on a 2D input results in a 2D output. (b) Applying 2D convolution on a 3D input also results in a 2D output. (c) Applying 3D convolution on a 3D input results in 3D output, keeping the temporal signal [27].

3D convolutions allow to learn the temporal information, while 2D convolutions ignores the temporal signal. In other words, the input to a 3D convolutional layer is $c \times d \times w \times h$, where c is the number of channels, d the temporal depth and w, h the width and height respectively. Each convolutional kernel is of size $k_d \times k_w \times k_h$ and there are n of them. The convolution operation slides with stride $s_d \times s_w \times s_h$, thus in both spatial and temporal direction. Since the 3D convolution preserves the temporal information the same way as for the spatial information, it learns invariant features to both spatial and temporal translations.

2.3.3 *Regularization Techniques*

Neural networks are prone to overfit, i.e. they model the training data too well and perform poorly on unseen data. One way to overcome this problem is to apply regularization. The two most popular regularization techniques are dropout and early stopping.

DROPOUT. Dropout [28] is proposed as an effective way to regularize deep NNs by introducing stochasticity in neuron activations. Specifically, dropout randomly drops the value of neurons with a certain rate and sets the values to zero during network optimization. As a result, the network is able to generalize better and is less prone to overfitting.

EARLY STOPPING. As explained earlier, an overfitting model performs well on training data, but poorly on unseen data. During training, the performance of the network on unseen data is evaluated with a validation set, i.e. a part of the dataset that is representative of the test set but that has not yet been seen by the model. An overfitting network results in an increase in the validation error while the training error steadily decreases. By keeping track of the lowest validation error after each epoch, one can halt the training when the model performance on the validation set hasn't improved for a number of epochs.

3

RELATED WORK

In this chapter, we review recent work on remote sensing for U.S. corn and soybean yield forecasting in a chronological manner. We show the evolution of ML techniques used until today and identify the remaining gaps.

As the use of deep learning for crop yield prediction is still in its infancy, additional deep learning approaches for other applications in remote sensing are presented.

3.1 REMOTE SENSING FOR CROP YIELD PREDICTION

Johnson 2014 [10] uses MODIS NDVI and Land Surface Temperature products in combination with precipitation data for 12 U.S. states accounting for 75% of the U.S. corn and soybean production. The data is collected 32 times for 8-days periods from Mid-February to late October from 2006 to 2012. Johnson creates two datasets, one for corn and one for soybean. Image pixels that do not belong to corn farmlands are masked. The same procedure is applied to soybean farmlands. Pixel values are then averaged per county and per year for all observation. The Pearson product-moment correlation coefficient is used to understand the relationship between corn and soybean yield and the remote sensing products. It appears that there is a strong relationship between NDVI values and soybean yield in the summer months, and an inverse relationship during spring. Oppositely, daytime temperature has a positive relationship in spring and inverse relationship in summer. Precipitations showed no relationship with soybean yield. Johnson then applies his findings to predict the yield feeding NDVI and daytime temperature data to Decision Trees (DT). Johnson points out that it should still be investigated whether the method could be applied to other crops or areas.

Kuwata and Shibasaki 2015 [12] use the MODIS EVI product combined with ground temperature measurements to predict corn yield per county in Illinois. It is not clear whether the image pixels were averaged per county or if EVI images were used directly. They feed the data to a SVM and a Deep Neural Network (DNN). Their DNN is able to provide accurate predictions, opening the door to further investigate the use of DL for crop yield prediction.

Kim and Lee 2016 [11] use the MODIS products NDVI, EVI, Leaf Area Index (LAI) and land cover together with climate data (precipitation and temperature) for predicting corn yield in the Iowa state. The data is downloaded monthly from May to September, from 2004 to 2014 for 94 counties in Iowa that have a cropland area larger than 10% of the total county area. Cropland pixels are extracted from the satellite images and averaged per county (zonal operation). They feed the combination of zonal pixels and climate data to 3 models: SVM, DT and MLP. The models were trained on two different sets of months: May to September (the growing season of corn), and July to August. They produce 11 sets of validations with leave-one-out year cross-validation. Their results show that the set of months May-September performed better than the other. No control experiment is performed to show the impact of climate data on the predictions.

You et al. 2017 [1] use the MODIS products Surface Reflectance, Land Surface Temperature and Land Cover from 2003 to 2015 for 11 states that account for 75% of the U.S. soybean production. As in Johnson 2014 [10], the data is collected 32 times for 8-days periods from Mid-February to late October. Instead of averaging the image pixels per county, they create sequences of histograms of pixel-counts for the cropland areas. Continuing on the conclusions of Kuwata and Shibasaki 2015 [12] and Kim and Lee 2016 [11] that Deep Learning methods could be applied to remote sensing for yield prediction, they investigate CNN and LSTM networks to learn temporal features from the histogram sequences. They later integrate a linear Gaussian Process (GP) to the last layer of both the CNN and LSTM, causing the inputs that are close in terms of space (neighboring counties) and time (year) to produce outputs that are close together. Their results outperform all competing techniques. The use of GP improves the results even further, hinting that the spatial dimension of the data is of importance. Finally, they test the importance of bands by shuffling slices of the histograms across time and band dimension. The permutation test shows that SWIR bands have a higher response than traditional bands such as red and NIR (used to calculate NDVI). Additionally, it confirms the findings of Johnson [10] that land surface temperature is correlated with crop growth, especially in early months.

To summarize, most of the approaches [10, 12, 11] use vegetation indices to predict the yield, but as shown in [1], multispectral images and especially the non-traditional bands could add valuable information. Each work uses its own combination of months and years for soybean prediction. The findings of Kim and Lee [11] and Johnson [10] suggest that combining spring and summer months is preferable. It is however not yet established what exact set of months is the most suitable for soybean yield prediction. Similarly, the ideal amount of training years has not yet been determined. In the remote sensing images, non-cropland pixels are always masked.

This method overlooks the potential response value of croplands' surroundings. Additionally, RS images either have their pixels averaged per county [10, 12, 11] or are turned into histograms of pixel intensities [1]. The spatial dimension of RS images is therefore discarded when it could provide crucial information about crops environment such as soil properties and elevation. Furthermore, these models are designed for predicting yield in specific regions, and whether RS models trained on given locations can be transfered to different geographical locations remains an open question [10]. Finally, the listed approaches work with the major soybean-producing states and counties in the U.S. and omit regions with lower yield or smaller cropland concentration. While one could argue that it makes the research easier, they fail to present models that are proven to work with smaller-producing counties where yield forecasting is also needed.

3.2 Deep learning for other applications in remote sensing

Classification tasks in multispectral remote sensing images are widely researched. These tasks involve object detection such as buildings trees and roads, as well as land cover classification and crop classification. Until recently, RS crop classification was facing the similar issues as for RS yield prediction. Common approaches were using basic classification models such as SVM [29], DT [30] and Artificial Neural Network (ANN) [31]. RS images usually consisted of vegetation indices such as NDVI [32, 33, 34]. The first deep learning approaches consisted of 2D CNNs [35] that learned the spatial features of the remote sensing images. Inspired by the success of 3D CNNs for spatiotemporal learning from videos [36, 27], Ji et al. [37] developed a 3D CNN model to learn spatiotemporal features from multispectral remote sensing images for crop classification. They compared their results with 2D CNN and shallow methods such as SVM and k-Nearest Neighbors (KNN). They showed that CNN methods outperformed shallow methods. Furthermore, their 3D CNN achieved a near-perfect crop classification accuracy, establishing that 3D CNNs can efficiently learn spatiotemporal features in multispectral remote sensing images, and that it could be used for other remote sensing applications.

4

DATASET AND SATELLITE IMAGES PREPROCESSING

In this chapter, we describe the data used for this thesis: the yield data and satellite images. Additionally, we detail the preprocessing of satellite images.

4.1 DATASET

In this section, the data used for the experiments is described. The dataset is composed of satellite images (input data) and soybean yield (ground truth label). The data is collected over the years 2003 to 2016. We selected all the U.S. counties that grow soybean, with no constraints on the yield or the cropland concentration.

4.1.1 *Yield Data*

The U.S. Department of Agriculture (USDA) provides open source data and statistics for agriculture in the U.S.. The yield is downloaded from the USDA NASS Quick Stat tool [17] at a county level for the years of interest (2003 to 2016). A total of 1848 U.S. counties cultivate soybean. Figure 8 shows the average yield per year in the U.S.. The increasing trend of the yield over the years can be attributed to improved crop management practices (fertilizer, pesticide and herbicide), plant breeding and GMOs [38].

4.1.2 *Satellite Images*

NASA's Terra and Aqua satellites, launched in 1999 and 2002 respectively, carry the MODIS payload instrument capable of capturing data in 36 spectral bands [14]. Multispectral snapshots are taken daily at different spatial resolutions (250, 500 and 1000 meters). The spatial resolution is the side lenght of the area covered by one pixel in the image. For example, a resolution of 10 meters means that each pixel in the image covers an area of 100 m^2.

NASA provides four open-source MODIS product categories, namely atmosphere, land, cryosphere and ocean products.

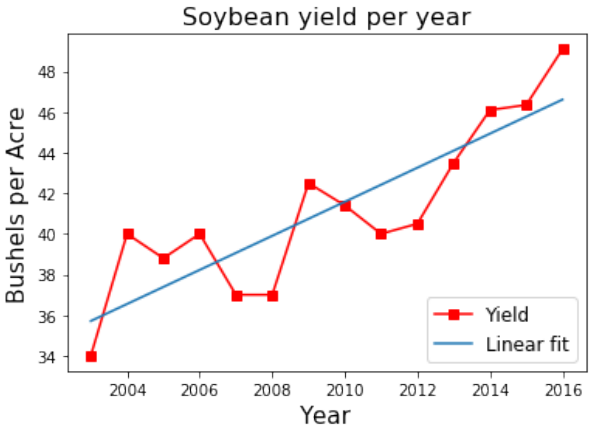


Figure 8: Average soybean yield per year in Bushels per Acre. The graph is plotted using the yield data collected from USDA [17].

Following the protocol of You et al. [1], NASA's MODIS land products *Surface Reflectance*, *Land Surface Temperature* and *Land Cover Type* are collected via Google Earth Engine [8]. For each of the 1848 soybean-growing counties, satellite images are collected every 8 days, 46 times per year , from 2003 to 2016.

More information about the collected MODIS products is provided in the following subsections.

MODIS Surface Reflectance

The *MODIS Surface Reflectance* product [39] provides 7 bands of spectral reflectance at the surface at a 500 meter resolution over 8 days. Each pixel contains the best possible observation during the 8-day period. The best observation is determined by high-observation coverage, low-view angle, the absence of clouds or cloud shadow, and aerosol loading. The spectral bands of the surface reflectance product are described in Table 1. Figure 9 shows an example of a measurement of the surface reflectance product.

The surface reflectance represents the percentage of light reflected by the Earth's surface, and is therefore comprised between 0 and 1. In this MODIS product, the reflectance is scaled between 0 and 10000. Then, an atmospheric correction algorithm is applied (i.e., correct the effect of the atmosphere on the reflectance) and returns values between -100 to 16000 (valid range). Values that are outside that range could not be corrected and should be discarded [40].

MODIS Land Surface Temperature

The *MODIS Land Surface Temperature and Emissivity* product [41] provides the average daytime and nighttime surface temperature over 8 days at a 1 km resolution. The temperature is retrieved by combining seven thermal infrared bands using the LST

Band	Description	Unit	Valid range
1	Red	Reflectance	-100 to 16000
2	Near-Infrared (1)	Reflectance	-100 to 16000
3	Blue	Reflectance	-100 to 16000
4	Green	Reflectance	-100 to 16000
5	Near-Infrared (2)	Reflectance	-100 to 16000
6	Short-Wave Infrared (1)	Reflectance	-100 to 16000
7	Short-Wave Infrared (2)	Reflectance	-100 to 16000

Table 1: MODIS Surface Reflectance: Bands description.

algorithm [42], returning values between 7500 and 65535. The actual temperature in Kelvin can be obtained by applying a scaling factor of 0.02.

The bands of the Land-surface temperature product are described in Table 2. Figure 10 shows an example of a measurement of the *Land Surface Temperature and Emissivity* product.

Band	Description	Unit	Valid range	Scaling factor
1	Day time Temperature	Kelvin	7500 to 65535	0.02
2	Night time Temperature	Kelvin	7500 to 65535	0.02

Table 2: MODIS Land Surface Temperature: Bands description.

MODIS Land Cover

The *MODIS Land Cover Type* product [43] provides an annual classification of the land at a 1 km resolution. The images contain one band, in which each pixel is given a land cover class, e.g. water, urban built-up, cropland, etc.. The different land cover classes are listed in Table 3. In this study, the land cover is only used to determine whether a location is cropland. A pixel is classified as croplands if at least 60% of the area is composed of cultivated croplands [44]. Figure 11 shows an example of a measurement of the *Land Cover Type* product.

Class	Name
0	Water
1	Evergreen Needle-leaf forest
2	Evergreen Broad-leaf forest
3	Deciduous Needle-leaf forest
4	Deciduous Broad-leaf forest
5	Mixed forest
6	Closed shrub-lands
7	Open shrub-lands
8	Woody savannas
9	Savannas
10	Grasslands
11	Permanent wetlands
12	Croplands
13	Urban built-up
14	Cropland/Natural vegetation
15	Snow and Ice
16	Barren or sparsely vegetated

Table 3: MODIS Land Cover Type: Land cover classes description.

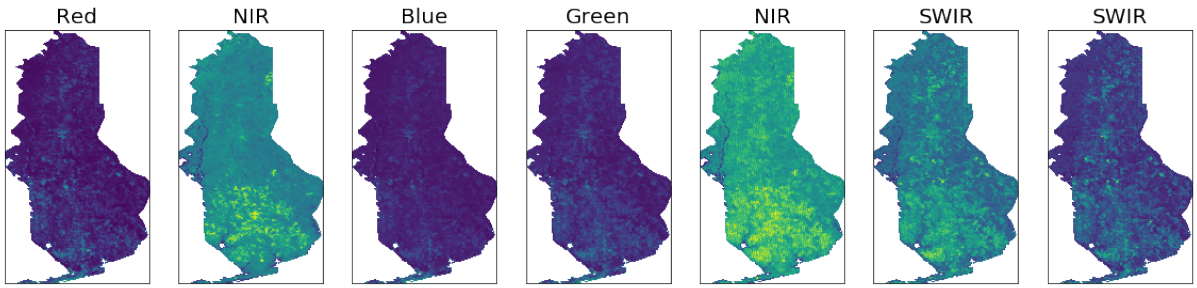


Figure 9: Observed surface reflectance from July 27th to August 4th, 2016 (Baldwin county, Alabama).

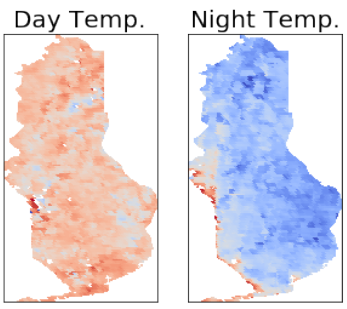


Figure 10: Observed land surface temperature from July 27th to August 4th, 2016 (Baldwin county, Alabama).

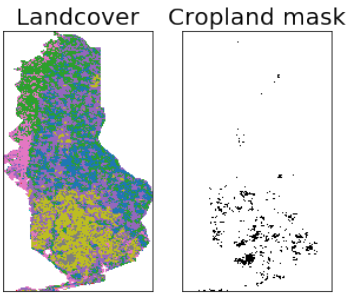


Figure 11: Landcover classification in 2016 (Baldwin county, Alabama).

4.2 SATELLITE IMAGES PREPROCESSING

In this section, we detail the preprocessing pipeline of the satellite images (Algorithm 1).

Algorithm 1: MODIS images preprocessing

```
1  foreach county do
2      sr ← TifToNumpy(sr_path, county) ;              // Surface Reflectance
3      lst ← TifToNumpy(lst_path, county) ;        // Land Surface Temperature
4      lc ← TifToNumpy(lc_path, county) ;                      // Land Cover
5      ValidRange(sr)
6      ValidRange(lst)
7      sr ← Reshape(sr, #years, #measurments)
8      lst ← Reshape(lst, #years, #measurments)
9      lc ← Reshape(lc, #years, 1)
10     lc ← Tile(lc, #measurments)
11     for y ← 0 to #years do
12         sr_year ← GetWeeks(sr,y)
13         lst_year ← GetWeeks(lst,y)
14         lc_year ← GetWeeks(lc,y)
15         mask ← GetMask(lc_year) ;                    // Cropland mask
16         MergeProducts(sr_year,lst_year,mask)
```

When collected from Google Earth Engine [8], the satellite images come in the *.tif* format and are cropped to the exact counties boundaries. For each product (i.e. surface reflectance, land surface temperature and land cover), we receive one file per county that contains the measurements per band for all years. We use the OSGeo/gdal

Python library [1] to read the *.tif* images into NumPy[2] arrays (Algorithm 1 line 2-4). For the surface reflectance and the temperature, we discard the values that lie outside the valid range (Algorithm 1 line 5 and 6). The resulting NumPy arrays contain the images of all years, all weeks and all bands, thus have shape:

$$\#\text{years} \times 46 \times \#\text{bands}, \text{height}, \text{width}$$

where *#years* represents the number of years, 46 is the number of 8-day measurements in a year, *#bands* is the number of bands in the data and height and width are the size of the image for a given county. The images are reshaped (Algorithm 1 line 7-9) to a size of:

$$\#\text{years}, 46, \#\text{bands}, \text{height}, \text{width}$$

As land cover measurements are provided once a year, the images have to be tiled to match the dimension of the temperature and the surface reflectance (Algorithm 1 line 10).

For each year, we extract the 32 measurements from February 18th to November 1st (Algorithm 1 line 12-14). From the land cover information, we create a cropland mask that allows us to determine the croplands location in the county (Algorithm 1 line 15). Finally, the bands of the 3 products are concatenated (Algorithm 1 line 16), starting from surface reflectance (bands $0-6$), then temperature (bands $7-8$) and finally land cover (band 9).

4.3 EVALUATION METRICS

4.3.1 *Bushels per Acre*

Grain (e.g., corn, soybean or wheat) yield in the U.S. is typically expressed in terms of Bushels per Acre (bsh/ac), whereas in the metric system, grain yield is expressed in terms of Kilograms per Hectare (kg/ha). In the context of grain production, a bushel expresses the weight of a certain type of grain depending on the moisture level of the grain. For instance, a bushel of corn at 15.5% moisture weights 56 pounds (lb), soybean at 13% moisture and wheat at 13.5% moisture weight 60 lb. [45]. The moisture level of reference for soybean grains is 13%, that is, the average moisture level at which soybean is harvested [15].

One soybean Bushel per Acre corresponds to 67.26 Kilograms per Hectare.

$$1 \text{ bsh/ac } = 67.26 \text{ kg/ha} \tag{3}$$

This conversion of the yield from U.S. customary units to metric units is provided for information purposes only. In this work, the soybean yield is always expressed in terms of Bushels per Acre.

1 https://github.com/OSGeo/gdal

2 http://www.numpy.org/

4.3.2 *Root Mean Squared Error*

To allow a fair comparison with the results of You et. al [1], we use the same evaluation metric: the Root Mean Squared Error (RMSE). The RMSE is a loss function commonly used in regression tasks. The RMSE formula is presented in Equation 4, where y_i is the target value, $\tilde{y}_i$ is the predicted value, and n is the number of samples. The square operation ensures that the error is always positive. The mean term ($\frac{1}{n}\sum_{i=1}^{n}$) allows to compare the prediction error on datasets of different sizes. The root function allows to have the loss on the same scale as the target values [46] .

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2} \tag{4}$$

The RMSE is presented in terms of Bushels per Acre.

4.4 DATASET SPLIT

Training Neural Networks is commonly done by splitting the data into three different sets: (1) the *training set* is used in the training phase for network learning, (2) the *validation set* is used to perform model section and early stopping if necessary, and (3) the *test set* is used to evaluate the final model performance [46].

The training set consists of remotely sensed images of randomly selected counties. To ensure that the counties are geographically evenly distributed, we randomly select 80% of the counties in each U.S. state in the training set. The remaining 20% of counties are used in the location validation set (LocVal). Figure 12 shows the distribution of counties in the train and location validation set.

Two training sets are used for the baseline replication [1]: a training set with images taken from 2003 to 2014 (Train14) and second one with images from 2003 to 2015 (Train15). The year 2013 is excluded of both training sets and used as a year validation set (YearVal).

Besides model selection, both of the validation sets are used for early stopping, i.e. stop the training at the step where the validation error is the smallest. LocVal is used to asses how well the network generalizes with unseen locations, whereas YearVal determines how well the network can predict yield for an new year. Additionally, we combine both of the validation set in a third set called CombVal.

The test set contains data for all the counties in the next year after training: 2015 for Train14 and 2016 for Train15.

The subsequent experiments use this dataset split, unless stated otherwise.

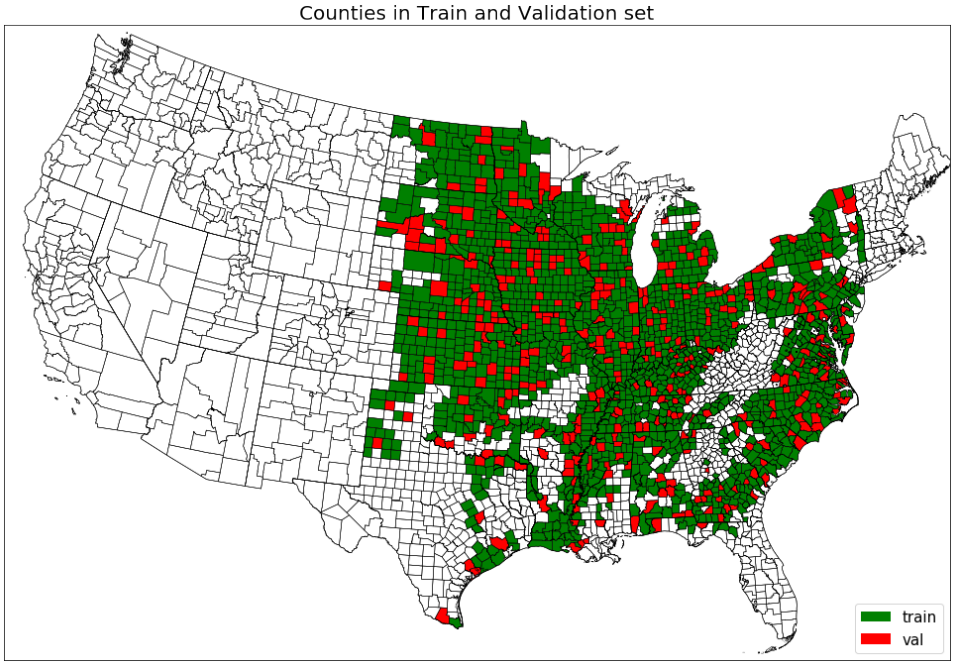


Figure 12: Distribution of the counties in the training and location validation set.

5

HISTOGRAM CNN: A TEMPORAL APPROACH

In today's scientific research, particularly in Artificial Intelligence, reproducibility and replicability is not often well addressed. According to Hutson [47], the AI field faces a reproducibility crisis. Very few publications (6%) have their code published, and even if it is, not all of them are extensively documented, making it hard to replicate the reported results. Furthermore, failed replications are often not reported.

An important contribution of this work is replicating part of the work of You et al. [1]. More specifically, we use the same dataset and follow the same data preprocessing pipeline, model architecture and training protocol for their CNN based algorithm. Then we compare our results to theirs and later on set this algorithm as our baseline for further control experiments (Chapter 6) to answer the first two research questions defined in the Introduction.

We first introduce the histogram CNN proposed by You et al. [1]. Then we describe the quite involved preprocessing pipeline to obtain the histograms (input to histogram CNN) from raw data and the network architecture. We continue with a thorough documentary of implementation details and finally report the results of the baseline replication.

5.1 WORKFLOW

In this section, we detail the workflow (Figure 13) of You et al. [1]. We choose to use their proposed histogram CNN model for our experiments as it performs better than the LSTM model. Here we present the overview of the workflow:

1. The counties of 11 major soybean-producing states are selected.

2. As mentioned in Chapter 4, yearly yield data at a county level is downloaded from USDA NASS [17] from 2003 to 2015. The yield data is used as ground truth target. The remote sensing data is downloaded via Google Earth Engine [8]. Three MODIS products are selected: Surface reflectance, Land surface temperature and Land cover. The products are downloaded for every selected county, 32 times a year from 2003 to 2015.

3. The remote sensing data is transformed into 3D histograms of pixel intensities. This will be further explained in Section 5.2.

4. The histograms are fed to a CNN.

5. After training, the CNN predicts soybean yield per county for the testing year.

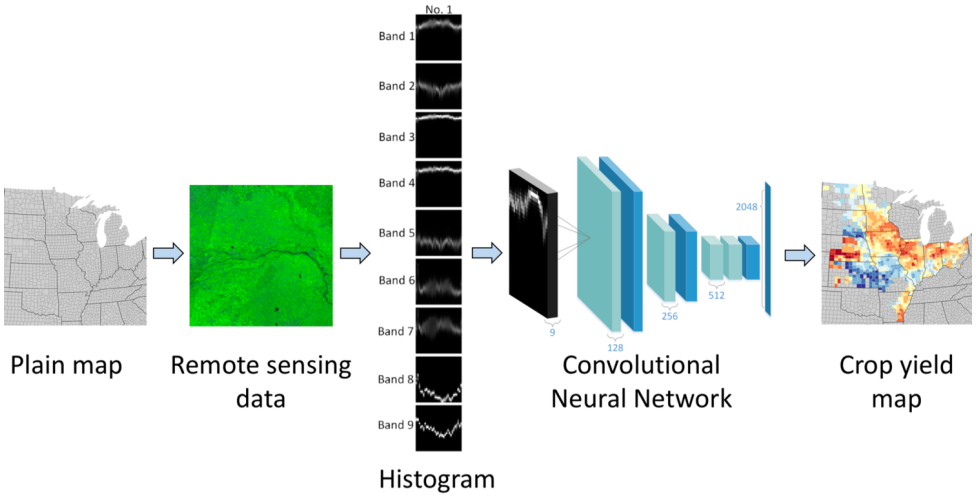


Figure 13: The workflow of You et al.'s approach [1].

5.2 HISTOGRAMS PREPROCESSING

You et al. [1] make the assumption that image pixels are permutation invariant. That is, they consider that the position of pixels only marks the location of farmlands and that the yield prediction would not differ if the pixels were moved. By making this assumption, they ignore the fact that pixel positions may provide useful information about soil- and eco-properties of the area, such as whether there are water sources nearby the croplands and whether they are close to urban areas. We later on tackle this problem in Chapter 7 by introducing a new algorithm but for now and for the purpose of baseline replication, we consider that this assumption holds and transform remote sensing images to 3D-histograms as inputs to the histogram CNN.

To perform the transformation, we follow the same procedure as You et al. [1] illustrated in Figure 14. First, we use the land cover information to mask the pixels that are not marked as croplands. Then, each band is transformed into a 32-bins histogram of pixel counts, producing a 32×9 (there are 9 bands) histogram matrix per image. As county images are collected 32 times a year, we gather the histograms throughout a year and stack them together, arriving at a 3D-histograms of size $32 \times 32 \times 9$ per year per county.

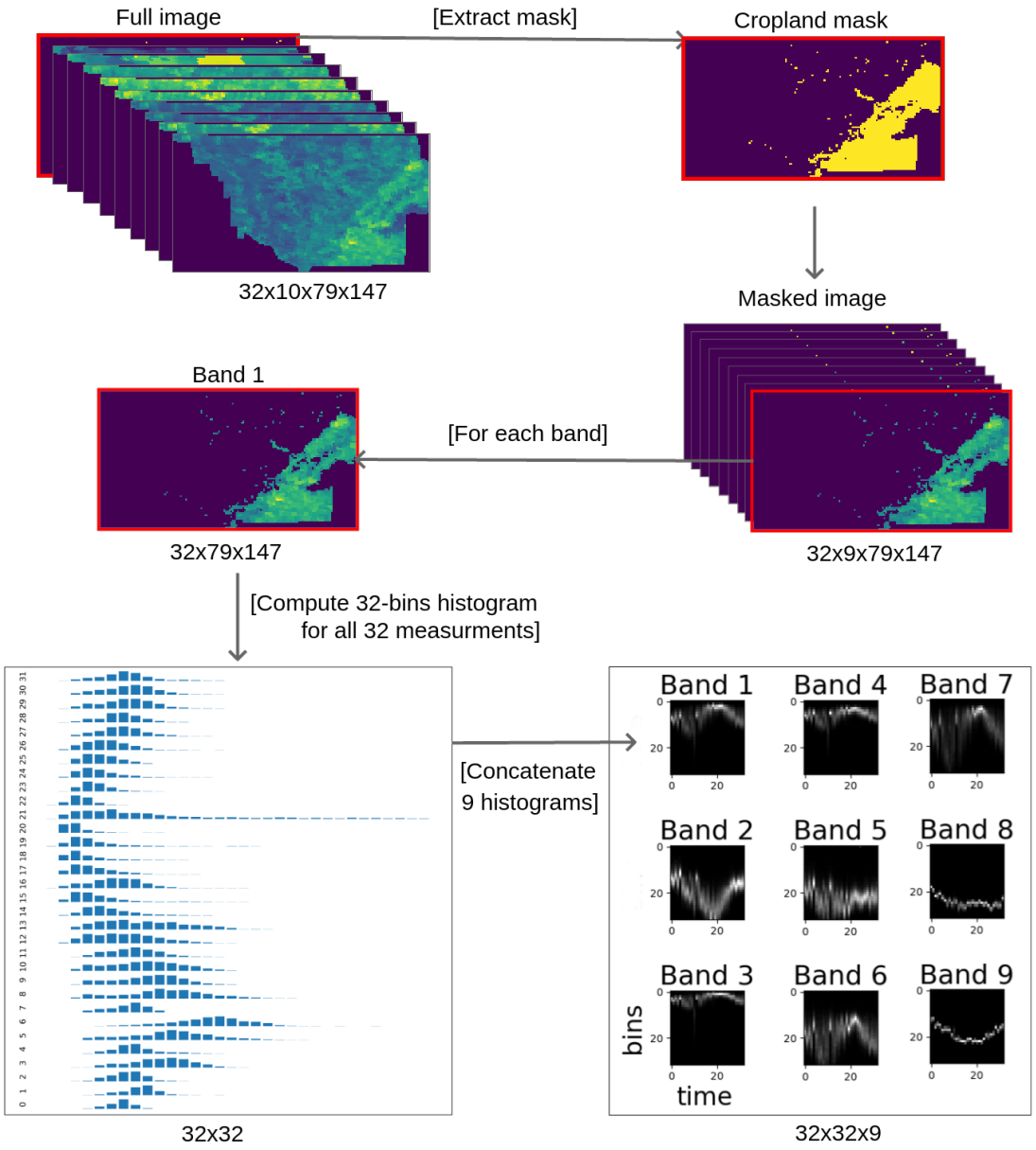


Figure 14: Workflow of the preprocessing of 3D-histograms.

5.3 MODEL ARCHITECTURE

Inspired by [48], where a 2D Convolutional Neural Network (2D-CNN) is capable of learning temporal patterns for video classification tasks, You et al. [1] also propose a 2D histogram CNN for the 3D-histogram data, henceforth we refer to this model as HistCNN. In particular, HistCNN receives 3D-histograms of size $32 \times 32 \times 9$ as input. The network is composed of 3 convolutional blocks and one fully-connected layer. Each block consists of two convolutional layers; the first convolutional layers are of kernel size 3×3 and stride-1 and the second convolutional layers are of kernel size

3×3 stride-2. The second convolutional layer is specifically placed to replace pooling layers as histograms are not translation invariant [1]. The number of filters starts at 128 at the first convolution block, and is doubled after every stride-2 convolutional layer. Each convolutional layer is followed by batch normalization, a ReLU non-linear activation and a dropout layer with rate 0.5.

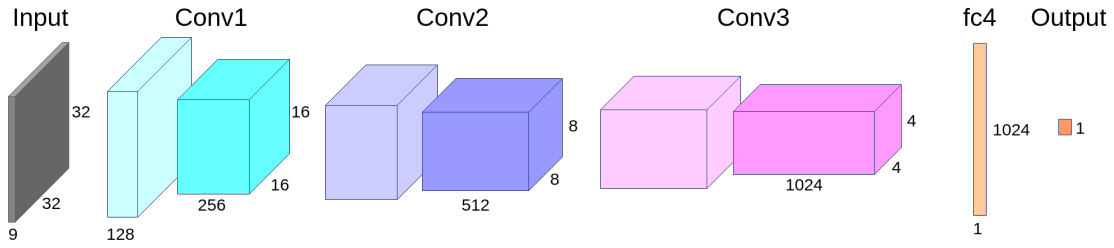


Figure 15: Architecture of the HistCNN. Stride-1 convolutional layer are light-colored, stride-2 convolutional layers are dark-colored.

5.4 TRAINING

The HistCNN is trained for a maximum of 1500 iterations. We use early stopping to stop the training at the best validation error, preventing the network from overfitting. As aforementioned, we construct three different validation sets (namely LocVal, YearVal, CombVal). Thus, during the training we use three sets to evaluate the network. Depending on the validation set we stop the training at different epochs, which results in several HistCNNs that have slightly varying performances.

We use the same hyperparameters as in [1] without additional tuning, where the mini-batch size is 64, the optimizer is Adam with initial learning rate 0.001 and a combined L2-norm and L1-norm.

5.5 RESULT REPLICATION AND VALIDATION METHOD SELECTION

In [1], a Root-Mean-Squared Error (RMSE) of 6.4 for the HistCNN model trained on the years 2003 to 2014 and tested on 2015 is reported. No results are available for the year 2016 because MODIS satellite images for that year were not available yet at the time of their work.

In addition, the composition of the validation is not clear based on the information from [1] . Therefore, we assess our three different kinds of validation to determine which one provides the best stopping criteria while performing baseline replication. The results in Table 4 and in Figure 16 consistently report better results when using the year validation. This is likely due to the fact that the year validation is the

evaluation method that is the most consistent to the test set: although the year is unseen, the counties have already appeared in the validation set. If not stated otherwise, subsequent experiments always validate on the year.

When testing on 2015, similar results to You et al. are achieved, with a RMSE of 6.6 using the year validation (Table 4). The small difference in terms of RMSE can be caused by different factors. First, weights are initialized randomly and DNN optimization is non-convex, therefore training the same network multiple times will always lead to very similar but different results. Second, their models were implemented in Tensorflow 1.2 [49], whereas we used Pytorch 0.2 [50]. Even though these Machine Learning frameworks in theory follow the same formulation, there always exist some small numerical differences in the back-end implementation. Third, our validation set is different than theirs, and it was not possible to infer the exact way of train/val splitting from the content of [1]. Last, the paper claims to only include counties from 11 major soybean producing U.S. states, but without specifying the exact states and counties selected. We use *all* the soybean producing counties (and therefore states), including the ones that have low yield and low cropland density, which can lead to inferior testing results. Despite these small differences, achieved RMSEs are on par with You et al. [1], which indicates a reasonable replication of their pipeline.

Training on the years 2003 to 2015 and testing on 2016 results in a RMSE of 7.5 on the year validation. The error is higher than for the 2015 prediction probably because the average yield in 2016 was significantly higher than in the previous years as seen in Figure 8 (Table 4); as a result, the model was not able to catch the causes that led to such a big difference only based on the remote sensing images.

Train Years	**Test Year**	**Validation**	**You et al.[1] Results (RMSE)**	**Our Results (RMSE)**
$2003-2014$	2015	Year	6.4	**6.599**
$2003-2014$	2015	Locations	6.4	7.014
$2003-2014$	2015	Combined	6.4	7.043
$2003-2015$	2016	Year	-	**7.538**
$2003-2015$	2016	Locations	-	8.437
$2003-2015$	2016	Combined	-	8.253

Table 4: Results of the baseline replication in terms of RMSE. The model is trained from 2013 to 2014 and tested on 2015. We compare our performance with the one from You et al.[1]. We later train on an additional year. The best results are marked in bold.

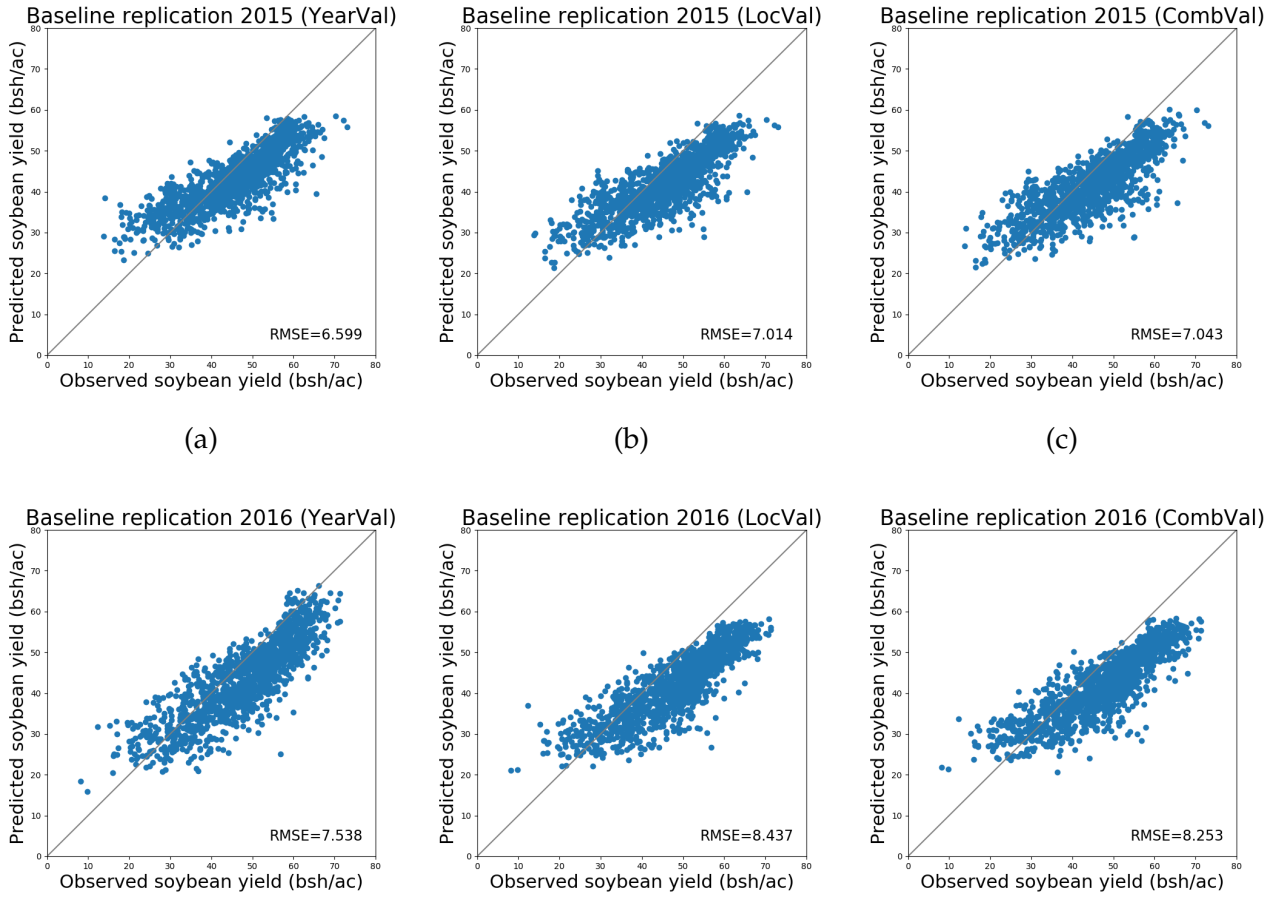


Figure 16: Scatter plots of the predicted yield per county vs. the observed yield for the different validations. Each dot represent a prediction for a county. The plots indicates a positive linear relationship with the predictions. In overall, low yield prediction are overestimated, while high yield predictions are underestimated. The plots confirm, for both 2015 and 2016 predictions, that the year validation is better as the dots lie closer to the $y = x$ line (i.e. perfect prediction)

6

CONTROL EXPERIMENTS WITH HISTOGRAM CNN

The previous chapter introduces the approach taken for reproducing You et al.'s [1] results, tests the model on the year 2015 to conclude that our results are close to the reported results, and further tests the model on the year 2016. In this chapter, we perform control experiments with HistCNN to answer the first two of the research questions proposed in the Introduction. First, we perform time control experiments to answer the research question: *RQ1: What are the best temporal training settings for soybean yield prediction?* Second, we perform location control experiments to answer the research question: *RQ2: Can an RS model trained on specific locations be generalized to new locations?*

For the rest of this chapter, we motivate and elaborate the experimental setup of each control experiment, followed by results and analysis.

6.1 TIME CONTROL EXPERIMENTS

You et al. [1] trained their model with data from 2003 to 2014. However, it is unclear if these years are the optimal source to predict the soybean yield of year 2015. Furthermore, You et al. [1] use a sequence of satellite measurements from February 18th to November 1st to compose the histograms, claiming this period to be from before planting to before harvesting. However, as explained in section 2.2, U.S. soybean is planted between April and June and harvested from September to November. Therefore, the input of HistCNN in fact overlaps the harvesting time for some regions, which appear to be counter intuitive as forecasting is mostly needed before harvesting.

Hence, we set up two types of time control experiments to determine the best temporal training settings for soybean yield prediction (RQ1). In the first control experiment, we test which years are optimal to be used in the training set. In the second control experiment, we identify the most effective months to include as inputs to the HistCNN.

6.1.1 *Control experiment: optimal period for training*

To determine the ideal interval of years for training, we train HistCNN on different year combinations and test the corresponding models on the 2016 test set. We design three training sets including data from different year ranges:

- First 6 years (2003-2009).
- Last 6 years (2009-2015).
- Odd years (2003,2005,2007,...,2015).

By training the model on older years and testing it on a recent year (2016) we can evaluate if older years are still representative of the current soybean growth scheme. On the other hand, using only the most recent six years allows us to determine if a training set that only contains more recent years is more suitable for our problem. Training on odd years allows us to have a similar training set as the baseline, but still keeping the same number of training years as for the other two control experiments, so as to see whether a wider time span of training years is beneficial.

As the year used for year validation is included in training, we perform validation with the location validation set only.

The results of the control experiment over the years (Table 5) show that it is highly preferable to only include recent years in the training set. Older years only bring noise as they are not representative of the soybean growth scheme anymore. This can be due to the constant evolution of agricultural practices, including improvements in fertilizer and pesticides usage, as well as continuous research in plant breeding and GMOs, providing farmers with more productive plants.

Train years	**Start month**	**End month**	**Test year**	**Validation**	**RMSE**
$2003-2009$	Feb	Nov	2016	Locations	9.87
$2009-2015$	Feb	Nov	2016	Locations	**7.532**
$2003, 2005, ..., 2015$	Feb	Nov	2016	Locations	9.172
$2003-2015$	Feb	Nov	2016	Locations	8.437 (B)

Table 5: Results of the control experiments on the number of years. We train the model on old and recent years, and test it on 2016. The best combination is marked in bold. For comparison, the baseline result is provided and marked with a (B).

6.1.2 *Control experiment: Months range*

Although the sequences of observations from February 18th to November 1st are used as inputs for HistCNN in [1], we question whether this is the ideal start and end date. To investigate, we train HistCNNs on different periods of the year. The starting points are chosen from before, during and after the planting period (February, April, June), and the ending point are chosen from before and during the harvesting period (July, September, November). The validation is performed on the year validation set.

The first set of results in Table 6 reports the performance of the model with different *starting* months. When starting in mid-February, the only bands that provide useful information are the temperature bands as soybean is not planted yet. In fact, the soil temperature a month before the planting period may hint the date at which the soybean seeds are planted. For instance, if the month of March is colder than usual, the soil may take a longer time to warm up, delaying the usual planting date. The surface reflectance bands, on the other hand, only provide useful information when the plants fully emerged from the ground, i.e. in summer. Starting in April, i.e. right at the start of the planting period, produces a notable improvement in terms of RMSE. Starting in June delivers the worst setting, likely because the early stages of the plant are ignored, which are crucial to the development, hence the yield, of the plants.

The second set of results in Table 6 reports the performance of the model with different *ending* months. Being able to predict the yield before harvest is important for management practices. Our results also support that predicting in early September, i.e. two weeks before the first soybean crops are harvested, provides the best results. It is likely that the spectral response of farmlands changes as crops are being harvested, and possibly brings noise to the observations after September. The predictions in early July have the worst results, likely because the crops aren't close to maturity yet, and a lot of factors, such as climate or pest could potentially still impact their development.

For the last set of results presented in Table 6, we combine the best settings found in the two time control experiments: training on recent years, combining months from February to September, or April to September. Surprisingly, the months combination Feb-Sep generates slightly better results than the Apr-Sep combination. This may confirm the findings of You et al. [1] and Johnson [10] that land surface temperature is correlated with crop growth, especially in early months.

Train years	Start month	End month	Test year	Validation	RMSE
2003 – 2015	Feb	Nov	2016	Year	7.538 (B)
2003 – 2015	Apr	Nov	2016	Year	**7.157**
2003 – 2015	Jun	Nov	2016	Year	7.667
2003 – 2015	Feb	Jul	2016	Year	8.752
2003 – 2015	Feb	Sep	2016	Year	**6.963**
2003 – 2015	Feb	Nov	2016	Year	7.538 (B)
2009 – 2015	Apr	Sep	2016	Year	7.479
2009 – 2015	Feb	Sep	2016	Year	**7.391**
2009 – 2015	Feb	Nov	2016	Locations	7.532

Table 6: Results of the months control experiments. We perform experiments with different starting and ending months. We later use the best combinations found here and in the year control experiments to validate our results. The best performing months combinations are marked in bold. For comparison, the baseline result are provided and marked with a (B).

6.2 LOCATION CONTROL EXPERIMENTS

One challenge in enabling wide application of machine learning methods in crop yield prediction is the lack of training data for certain regions, especially in developing countries. While remote sensing images are available worldwide, ground truth yield data in developing countries can be scarce. Crop yield forecasting in developing countries is particularly important to improve food security: by being able to forecast the production in developing countries, one is able to plan the humanitarian food aid needed.

Since it is less feasible to train a neural network end-to-end for developing countries due to the (current) lack of data, it is important to determine how well a model trained on specific locations can be generalized to new locations, which is precisely our second research question (RQ2). To answer this question, we first split the locations from our training set by their climate and ecological properties, i.e. by ecoregions.

6.2.1 *Ecoregions*

United States Environmental Protection Agency (EPA)'s ecological regions (ecoregions) classify the North-American continent into areas that share similar ecosystems. Ecoregions are determined based on their geology, landforms, soils, vegetation, cli-

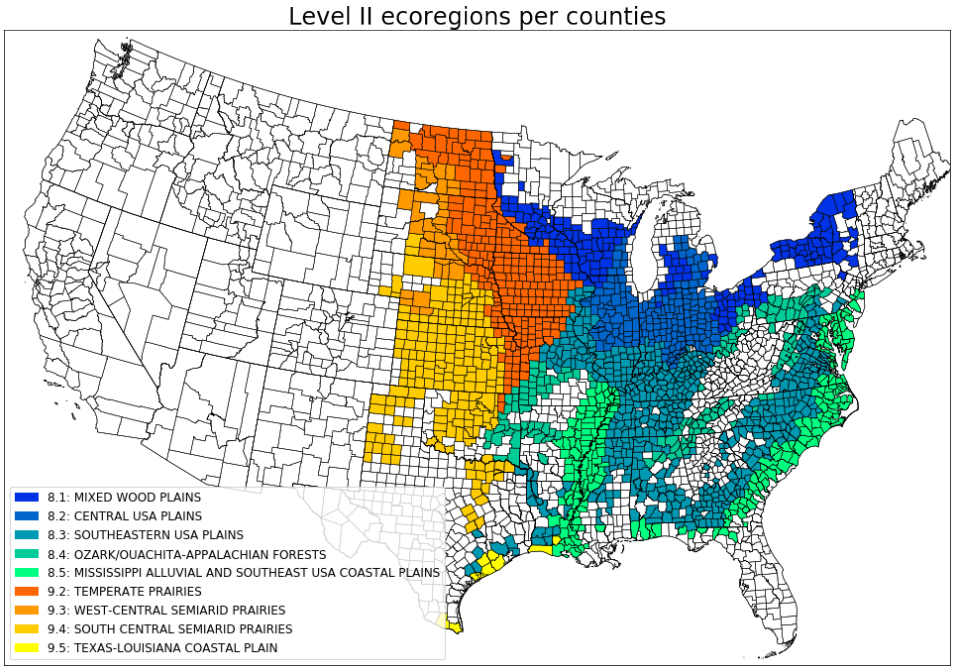


Figure 17: Counties in ecoregion 8 and 9

mate, land use, wildlife, and hydrology [51]. There are three levels of ecoregion divisions. Level I ecoregions is the broadest level and divides the continent into 15 ecoregions. Level II ecoregions are nested within level I ecoregions and provides a more specific description of the ecosystems. Similarly, level III ecoregions are nested into level II ecoregions, describing even finer ecological properties [51].

The most represented level I ecoregions in our dataset are Eastern temperate forests (66%) and Great Plains (27%). We do not take into account the other ecoregions as they represent less than 7% of our data.

The ecosystem of Eastern temperate forests (ecoregion 8) is characterized by a warm, humid and temperate climate, with hot and humid summers and mild to cold winters. It is mainly covered by dense and diverse forests and has a dense population. The main activities consists of urban industries, agriculture and forestry. Great plains (ecoregion 9), on the other hand, mainly consists of flat grasslands, and has a scarcity of forests. The climate is semiarid with high winds. The population is centered around urban areas. The main activities of the region are agriculture, mining as well as gas and oil extraction.

When splitting the regions, we categorize the geographical locations into the two most represented ecoregions: Eastern Temperate Forests (8) and Great Plains (9). However, the distribution of counties in these ecoregions is not balanced (66% vs. 27%). We therefore further divide the dataset into level II ecoregions, as illustrated in Figure 17. The level II ecoregions 9.2 to 9.5 are selected to be part of the Great Plains subset and contains 503 counties. The level II ecoregion 8.3 is selected to represent the Eastern temperate forest region, containing 516 counties.

6.2.2 *Control experiment: locations transfer*

Given a target region for crop yield prediction, we study how models trained on another region (source domain [52]) compare with models directly trained on the target region (target domain [52]). Intuitively, models trained on the source domain can not exceed the performance of models trained on the target domain. Nonetheless, we are interested in gaining insight on the discrepancy in evaluation between source and target models, and what could potentially impact the transferred model performance. Specifically, we perform the following location control experiments: we first split the training regions to 2 types, ecoregion 8 and ecoregion 9 as instructed in the previous section, then we train a crop yield prediction model on each region, and test each of the trained models on the other region as well as on its own region. Additionally, we also present results when training with both regions combined and compare the performances altogether.

Concretely, we train HistCNN with ecoregion 8, later with ecoregion 9 and finally with all regions, using the data from February to September. All models are then tested on both ecoregions. The training years cover from 2003 to 2015, and the test year is 2016. Because of the reduced number of training counties, the validation is performed on the 2013 year validation set. The results are summarized in Table 7.

Training region	**Test on region 8 (RMSE)**	**Test on region 9 (RMSE)**
8. Eastern Temperate Forests	**8.18**$*$	12.611
9. Great Plains	9.682	8.657$*$
All	8.253$*$	9.103

Table 7: Results of the location control experiments. The models are trained on two different sets of ecoregions, and tested against each other. We further test the performance of the target regions with a model trained with both regions. The best performing domains are marked with an asterisk, and the overall best performance is marked in bold.

We first confirm that the best model for a target domain is indeed trained on exactly the target domain. The second best model is obtained when trained with combined data. This is an interesting outcome as adding more training data that comes from a slightly different distribution does not improve the model quality in our case. Thus we infer that the crop yield prediction model with satellite image is very sensitive to domain distribution. Moreover, the model trained on ecoregion 9 performs much better on ecoregion 8 than the other way around. In other words, different ecoregions can possess a varying degree of generalizability. In this case, ecoregion 9 is much more favorable as source domain for transfer learning than ecoregion 8 (e.g. RMSE

evaluated on ecoregion 8 of the model trained on ecoregion 9 is 9.682, whereas that evaluated on ecoregion 9 of the model trained on ecoregion 8 is 12.611).

More advanced domain adaptation or transfer learning techniques can potentially be applied in the case where training data is lacking for a given region, but we leave it as an important future research direction. From our experimental results and analysis, we conclude that:

- the crop yield prediction task with satellite image is very domain sensitive; hence it is essential to learn on a source region as closely resembling the target region as possible.
- some regions can generalize to new target regions better than the others; it is useful to identify these regions and leverage their generalizability for transfer learning.

7

3D CNN: A SPATIO-TEMPORAL APPROACH

In the previous chapters, we introduce the histogram CNN approach of You et al.'s and used it a baseline for several key control experiments, where we determine the best temporal training settings for MODIS-based soybean yield prediction as well as the potential generalizability of the baseline models for transfer learning. In particular, we discover that using early years in the training and that forecasting the yield in September result in more favorable prediction performances.

Nevertheless, with its location invariant assumption, the histogram CNN only allows us to extract temporal and spectral features from the satellite images while discarding spatial information. In other words, this method predicts crop yield by only modeling the spectral response of crops through time. However, intuitively, crop yield does not only depend on climate and health of the crops, which status can be perceived by looking only at farmlands areas, but also on soil properties, elevation of the farmlands and their surrounding areas. Therefore, we hypothesize that by including spatial information along with the spectral and temporal signal, more accurate crop yield predictions can be achieved.

In recent years, applications of 3D Convolutional Neural Networks (3D CNN) on human activity recognition from video [27, 36] and crop classification from remote sensing data [37] have been studied. Thanks to its convolutional kernel covering the receptive field along both spatial and temporal dimensions, 3D CNN have proven to be well-suited for tasks involving learning from spatiotemporal data. Inspired by these works, we hereby propose a 3D CNN framework for crop yield prediction, which, to our knowledge, has not been investigated by prior works.

In this chapter, we first illustrate the model architecture of our proposed 3D CNN model as well as the training and testing settings. We then collect and analyze the results. Lastly, we identify future possible improvements.

7.1 MODEL ARCHITECTURE

The input of each time step for our 3D CNN concatenates 10 bands from three MODIS products: 7 Surface Reflectance bands, 1 day and 1 night surface temperature bands, and an additional binary cropland mask derived from the MODIS land cover product.

The latter is used to differentiate the location of croplands and non-croplands areas. In addition, each channel (excepted the mask) is normalized to approximately have zero mean and unit variance across that channel. Finally, we use the period that works optimally with the baseline HistCNN, i.e. from February 18th to September 1st, that is we concatenate the observations over these 24 weeks to obtain the final input.

As [1] points out, not all the 10 channels are of equal importance for crop yield prediction. Thus, also taking computational cost into consideration, we start our network with a channel compression module to reduce the channel dimension from 10 to 3 without altering the temporal or spatial dimension. The module is composed of two 2D-convolutional layers. The first layer takes the input tensor from each time step individually and uses 10 filters performing 2D-convolutions of size 3×3. The output of the first layer is given as input to the second layer, which then performs 1×1 2D-convolutions with 3 filters. The detailed architecture is presented in Table 8.

Following the initial dimensionality reduction module, we stack a 3D CNN to regress the yield from the processed input. Here, we take inspiration from the 3D CNN for activity recognition, named C3D in [27], which in turn was motivated by the 2D VGG net [53]. The following sizes are noted $d \times k \times k$, where d is the temporal depth and k is the kernel height and width. As illustrated in Figure 18, the model consists of convolutional blocks and max-pooling layers. The convolutional blocks are composed of:

- 3D convolutional layers [36] with $3 \times 3 \times 3$ kernel, stride $1 \times 1 \times 1$ and padding $1 \times 1 \times 1$.
- 3D batch normalization [54] with $\epsilon = 1e^{-6}$ and momentum $= 0.1$
- ReLU activation [22]

The blocks are followed by max-pooling layers with kernel and stride size $2 \times 2 \times 2$, excepted for the first layer where the kernel and stride are of size $1 \times 2 \times 2$ to avoid shrinking the temporal depth too early. Since our dataset is significantly smaller than the activity dataset and the inputs are of smaller spatial dimension (UCF-101 [55]) used in [27], we remove one block from the original model to prevent overfitting. Finally, the 3D features are flattened into fully connected (FC) layers that output a final crop yield prediction. Besides the size of dataset, activity recognition task requires more abstract features than crop yield prediction, hence we further remove one FC layer from the model used in [27]. We list out the layers of the 3D CNN in Table 8.

Finally, we arrive at our proposed architecture that is customized to our data type, dataset size and learning objective. The whole pipeline of our proposed framework is shown in Figure 18.

layer name	operation	filters	kernel size	stride	padding	output size
DimR_1	conv2D	10	$1 \times 3 \times 3$	$1 \times 1 \times 1$	$0 \times 1 \times 1$	$10 \times 24 \times 64 \times 64$
DimR_2	conv2D	3	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$0 \times 0 \times 0$	$3 \times 24 \times 64 \times 64$
Conv_1	conv3D	64	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$64 \times 24 \times 64 \times 64$
Pool_1	max-pool	-	$1 \times 2 \times 2$	$1 \times 2 \times 2$	-	$64 \times 24 \times 32 \times 32$
Conv_2	conv3D	128	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$128 \times 24 \times 32 \times 32$
Pool_2	max-pool	-	$2 \times 2 \times 2$	$2 \times 2 \times 2$	-	$128 \times 12 \times 16 \times 16$
Conv_3_a	conv3D	256	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$256 \times 12 \times 16 \times 16$
Conv_3_b	conv3D	256	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$256 \times 12 \times 16 \times 16$
Pool_3	max-pool	-	$2 \times 2 \times 2$	$2 \times 2 \times 2$	-	$256 \times 6 \times 8 \times 8$
Conv_4_a	conv3D	512	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$512 \times 6 \times 8 \times 8$
Conv_4_b	conv3D	512	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$512 \times 6 \times 8 \times 8$
Pool_4	max-pool	-	$2 \times 2 \times 2$	$2 \times 2 \times 2$	-	$512 \times 3 \times 4 \times 4$
Fc_5	linear	-	-	-	-	1×1024
Fc_6	linear	-	-	-	-	1

Table 8: Layers architecture of the 3D CNN.

7.2 TRAINING

The training protocol of our 3D CNN follows the general guideline in [27] but with adjustments to fit the framework to our task. As our goal is to regress the yield per county of a given year, the network is hence optimized via mean square error using mini-batch of size 30 (as in [27]) and gradient descent on top of ADAM optimizer. As in [27], we also train the model for 16 epochs and initialize the network weights with zero bias and uniform distribution $\mathcal{U}(-a, a)$ where a is formulated in equation 5 (K is the kernel size):

$$a = \frac{1}{\sqrt{\prod_{i=1}^{len(K)} K_i}} \tag{5}$$

Our 3D CNN takes in a fixed-size $24 \times 10 \times 64 \times 64$, i.e. *days* $\times$ *number of channels* $\times$ *height* $\times$ *width*, input. To meet this requirement as well as to provide additional data augmentation, for each randomly sampled county during training, we randomly crop a 64×64 patch from its MODIS image as illustrated in figure 19. We first place a bounding box around the cropland area. If the bounding box or the original size of the image is too small, we first pad it to size 64. We then randomly select a 64×64 patch from the cropland bounding box. Since the distribution of croplands is not even, to ensure the training image quality, we enforce a minimum 10% cropland coverage

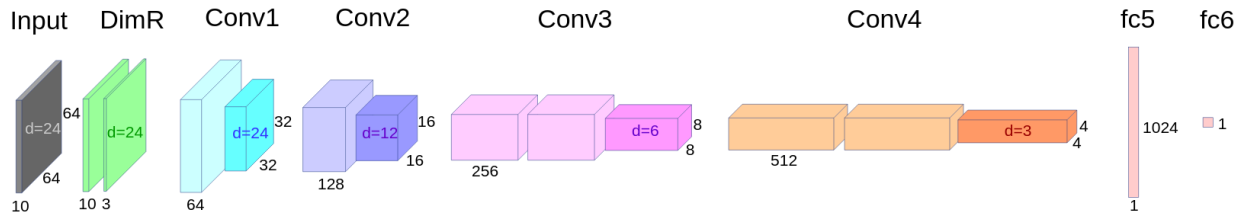


Figure 18: Architecture of the 3DCNN. Convolutional layers are light-colored, pooling layers are dark-colored. The change in temporal depth is denoted by d.

within the 64×64 patch. If the requirement is not met, we repeat the procedure a maximum of 10 times. If no patch meeting the 10% coverage requirement is found, we discard the county and move on to the next one.

We perform a hyperparameter search over SGD [56] and Adam [57] with different learning rates. Additionally, we tried adding a dropout layer to the FC layers and tried two different batch sizes: 30 and 50 (the largest batch size that the employed GPU[1] memory could afford). The optimal combination of hyperparameters found is Adam optimizer with $lr = 0.001$ and a batch size of 30.

7.3 TESTING

During evaluation stage, we feed in multiple 64×64 patches covering a given county to the trained 3D CNN and average the predicted crop yield over these patches as shown in Figure 20. Similarly to the random cropping procedure, we first pad the image to at least of size 64 if necessary and place a bounding box around the cropland area of the MODIS image. Then, we extract patches from the bounding box by moving a 64×64 sliding window across the entire bounding box with stride $10\% \times w$ along with width and $10\% \times h$ along the height. If a sliding window does not have at least 10% cropland coverage, we discard the result from final averaging. If we fail to find a least one window meeting the 10% coverage requirement, we include the best window found (i.e. the one with the highest cropland coverage) to ensure that all counties are considered in the evaluation.

7.4 RESULTS AND ANALYSIS

In Table 9, we compare the 3D CNN performance against the HistCNN and other baseline methods that You et al. [1] replicated to compare their results. These methods include Ridge regression [9], Decision Tree (DT) [10] and DNN [12]. The input to the

1 Nvidia Tesla K80 https://www.nvidia.com/en-us/data-center/tesla-k80/

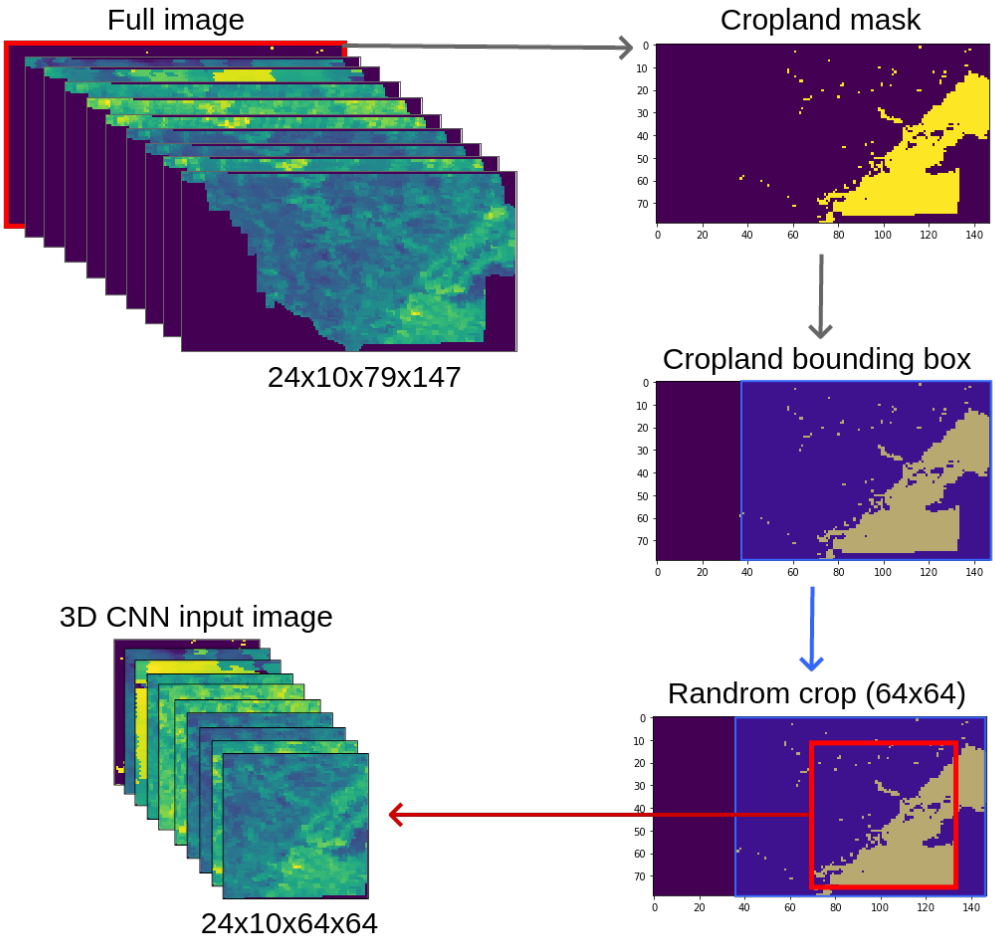


Figure 19: Workflow of the random cropping procedure for training input to the 3D CNN.

(other) baseline methods is composed of NDVI values averaged over the croplands of the county for a sequence of 32 observations per year. The training is performed from 2003 to 2014 and tested on 2015.

As seen in Table 9, the 3D CNN outperforms competing approaches, including Hist-CNN. Leveraging the spectral, temporal and spatial information of remote sensing images clearly improves the performance in comparison to the traditional temporal approaches [9, 10, 12] and the HistCNN spectro-temporal approach, shedding light on promising future directions in utilizing deep learning tools for remote-sensing data.

More experiments could be conducted to further improve the performance of the model. For example, our architecture directly borrows much of its design from [27] thus there is potential room for improvement along this direction, such as investigating the ideal number of filters across the network or changing the temporal depth of the convolution kernels could improve the performance. Additionally, we are not sure if the dimensionality reduction layer brings improvements (besides reducing the computational cost), which is also worth further control experiments.

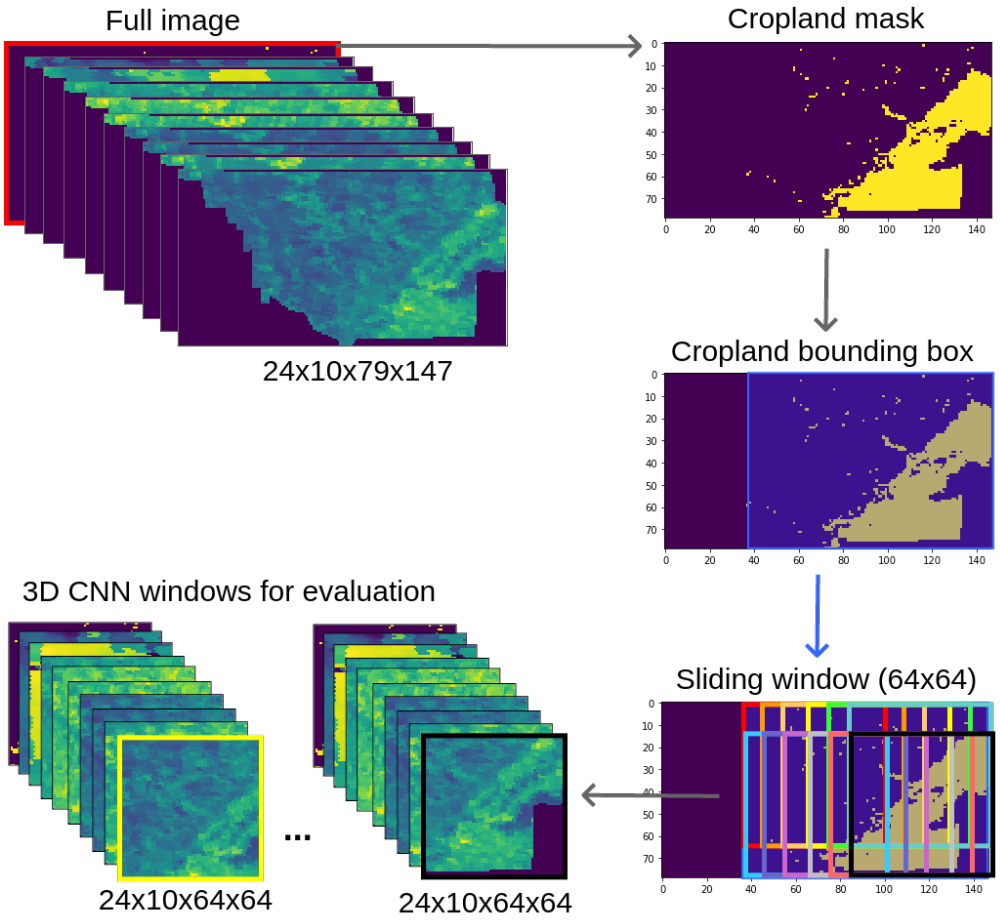


Figure 20: Workflow of the sliding window procedure for evaluation input to the 3D CNN.

Finally, it would be valuable to use deconvolutions [58] to visualize the features learned by the network or apply saliency map [59] to highlight the regions on which the network makes its decisions. Thereafter, we can provide the qualitative interpretation to remote-sensing experts and combine their expertise to better comprehend 3D CNN's functionality for crop yield prediction task.

Year	Ridge	DT	DNN	HistCNN	3D CNN
2015	8.1	7.64	7.19	6.6	**5.22**
2016	-	-	-	7.39	5.32

Table 9: Results of the 3D CNN compared with competing approaches.

8

CONCLUSION

The main purpose in this study is to investigate the use of Convolutional Neural Networks for soybean yield prediction in the U.S. by answering three research questions:

RQ1 *What are the best temporal training settings for soybean yield prediction?*

RQ2 *Can a RS model trained on specific locations be generalized to new locations?*

RQ3 *Can 3D CNNs leverage both the spatial and temporal dimension of remotely sensed images for better crop yield predictions?*

In this work, we successfully replicated the work of You et al. [1] by transforming satellite images into 3D-histograms and feeding them to a 2D-Convolutional Neural Network. We used this Histogram-CNN as a baseline for performing time and location control experiments.

By performing the time control experiments, we conclude that it is preferable to only include recent years in the training set, and that a sequence of observation from February to September is optimal for soybean yield prediction with multispectral remote sensing images.

By performing domain transfer experiments on two different ecoregions, we discovered that the crop yield prediction task with satellite image is very domain sensitive; hence it is essential to learn on a source region as closely resembling the target region as possible.

With the location invariance assumption of the histograms, the HistCNN predicts crop yield by only modeling the spectral response of crops through time, but disregard spatial information. Inspired by recent work on 3D CNN for human action recognition in videos [27] and on crop-classification in remote sensing [37], we propose a 3D CNN for crop yield prediction, leveraging the spatial, spectral and temporal dimension of the remote sensing images. Our results significantly outperform competing state-of-the-art machine learning methods, shedding light on promising future directions in utilizing deep spatiotemporal feature learning tools for crop yield prediction.

One natural future direction is to conduct more experiments to further optimize the architecture design as well as the training of a 3D CNN for crop yield prediction

with remote sensing images, by for instance, further investigating the ideal number of filters across the network or the ideal temporal depth of the convolution kernels, and determine whether the dimensionality reduction layer brings improvements besides reducing the computational cost. Based on our location control experiments, another possible future work direction is to use more advanced domain adaptation or transfer learning techniques to allow crop yield prediction in regions where training data is lacking, e.g. in developing countries. Lastly, visualization of the features learned by the 3D CNN to explain its functionality can be an essential add-on for remote sensing experts to better comprehend this line of techniques. We envision that deep learning based crop yield prediction will contribute to more efficient planning and managing agriculture practices in the foreseeable future.

BIBLIOGRAPHY

[1] J. You, X. Li, M. Low, D. Lobell, and S. Ermon, "Deep Gaussian Process for Crop Yield Prediction Based on Remote Sensing Data." in *AAAI*, 2017, pp. 4559–4566.

[2] G. A. United Nations, "Tansforming our world: the 2030 agenda for sustainable development," New York, 2015.

[3] C. O. Justice and I. Becker-Reshef, "Report from the workshop on developing a strategy for global agricultural monitoring in the framework of group on earth observations (geo)." [Online]. Available: http://www.fao.org/gtos/igol/docs/meeting-reports/07-geo-ag0703-workshop-report-nov07.pdf

[4] A. Ceglar, A. Toreti, C. Prodhomme, M. Zampieri, M. Turco, and F. J. Doblas-Reyes, "Land-surface initialisation improves seasonal climate prediction skill for maize yield forecast," *Scientific Reports*, vol. 8, no. 1, p. 1322, 2018.

[5] B. Basso, D. Cammarano, and E. Carfagna, "Review of crop yield forecasting methods and early warning systems," in *Proceedings of the First Meeting of the Scientific Advisory Committee of the Global Strategy to Improve Agricultural and Rural Statistics, FAO Headquarters, Rome, Italy*, 2013, pp. 18–19.

[6] G. Hoogenboom, J. W. White, and C. D. Messina, "From genome to crop: integration through simulation modeling," *Field Crops Research*, vol. 90, no. 1, pp. 145–163, 2004.

[7] J. B. Campbell and R. H. Wynne, *Introduction to Remote Sensing*. New York: The Guilford Press, 2011.

[8] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore, "Google earth engine: Planetary-scale geospatial analysis for everyone," *Remote Sensing of Environment*, 2017. [Online]. Available: https://doi.org/10.1016/j.rse.2017.06.031

[9] D. K. Bolton and M. A. Friedl, "Forecasting crop yield using remotely sensed vegetation indices and crop phenology metrics," *Agricultural and Forest Meteorology*, vol. 173, pp. 74–84, 2013.

[10] D. M. Johnson, "An assessment of pre-and within-season remotely sensed variables for forecasting corn and soybean yields in the united states," *Remote Sensing of Environment*, vol. 141, pp. 116–128, 2014.

[11] N. Kim and Y.-W. Lee, "Machine Learning Approaches to Corn Yield Estimation Using Satellite Images and Climate Data: A Case of Iowa State," *Journal of the Korean Society of Surveying, Geodesy, Photogrammetry and Cartography*, vol. 34, no. 4, pp. 383–390, Aug. 2016. [Online]. Available: http://koreascience.or.kr/journal/view.jsp?kj=GCRHBD&py=2016&vnc=v34n4&sp=383

[12] K. Kuwata and R. Shibasaki, "Estimating crop yields with deep learning and remotely sensed data," in *Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International*. IEEE, 2015, pp. 858–861.

[13] J. E. Ball, D. T. Anderson, and C. S. Chan, "Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community," *Journal of Applied Remote Sensing*, vol. 11, Sep. 2017.

[14] NASA. Modis: Moderate-resolution imaging spectroradiometer. [Online]. Available: https://modis.gsfc.nasa.gov/

[15] J. Ruiz-Vega, "Soybean phenology and yield as influenced by environmental and management factors," Ph.D. dissertation, Iowa State University, 1984.

[16] L. C. Purcell, M. Salmeron, and L. Ashlock, *Arkansas Soybean Production Handbook*. Little Rock, AR: University of Arkansas Cooperative Extension, 2014.

[17] USDA. (2018) Usda national agricultural statistics service. [Online]. Available: https://www.nass.usda.gov/Quick_Stats/index.php

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] R. Quiza and J. P. Davim, "Computational methods and optimization," in *Machining of hard materials*. Springer, 2011, pp. 177–208.

[20] E. Gavves, K. Gavrilyuk, B. Kicanaoglu, and P. Putzky, "Uva deep learning course 2017," University of Amsterdam, 2017. [Online]. Available: http://uvadlc.github.io/

[21] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[22] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[23] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[24] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models."

[25] R. Fergus, "Neural networks," MLSS 2015 Summer School, Facebook AI Research, 2015. [Online]. Available: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf

[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[27] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4489–4497.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[29] M. Fauvel, J. Chanussot, and J. A. Benediktsson, "Evaluation of kernels for multiclass classification of hyperspectral remote sensing data," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 2. IEEE, 2006, pp. II–II.

[30] D. McIver and M. Friedl, "Using prior probabilities in decision-tree classification of remotely sensed data," *Remote sensing of Environment*, vol. 81, no. 2-3, pp. 253–261, 2002.

[31] T. Kavzoglu and P. M. Mather, "The use of backpropagating artificial neural networks in land cover classification," *International journal of remote sensing*, vol. 24, no. 23, pp. 4907–4938, 2003.

[32] B. D. Wardlow, S. L. Egbert, and J. H. Kastens, "Analysis of time-series modis 250 m vegetation index data for crop classification in the us central great plains," *Remote Sensing of Environment*, vol. 108, no. 3, pp. 290–310, 2007.

[33] X. Xiao, S. Boles, J. Liu, D. Zhuang, S. Frolking, C. Li, W. Salas, and B. Moore III, "Mapping paddy rice agriculture in southern china using multi-temporal modis images," *Remote sensing of environment*, vol. 95, no. 4, pp. 480–492, 2005.

[34] C. Conrad, R. R. Colditz, S. Dech, D. Klein, and P. L. Vlek, "Temporal segmentation of modis time series for improving crop classification in central asian irrigation systems," *International Journal of Remote Sensing*, vol. 32, no. 23, pp. 8763–8778, 2011.

[35] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 5, pp. 778–782, 2017.

[36] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[37] S. Ji, C. Zhang, A. Xu, Y. Shi, and Y. Duan, "3d convolutional neural networks for crop classification with multi-temporal remote sensing images," *Remote Sensing*, vol. 10, no. 1, p. 75, 2018.

[38] D. Egli, "Comparison of corn and soybean yields in the united states: Historical trends and future prospects," *Agronomy journal*, vol. 100, no. Supplement_3, pp. S–79, 2008.

[39] E. Vermote, "Mod09a1 modis/terra surface reflectance 8-day l3 global 500m sin grid v006," NASA EOSDIS LP DAAC, 2015. [Online]. Available: https://lpdaac.usgs.gov/node/804

[40] E. Vermote, S. Kotchenova, and J. Ray, "Modis surface reflectance users guide," *MODIS Land Surface Reflectance Science Computing Facility, version 6*, vol. 1, 2011.

[41] Z. Wan, S. Hook, and G. Hulley, "Myd11a2 modis/aqua land surface temperature/emissivity 8-day l3 global 1km sin grid v006," NASA EOSDIS LP DAAC, 2015. [Online]. Available: https://lpdaac.usgs.gov/node/829

[42] Z. Wan *et al.*, "Modis land surface temperature products users guide," *Institute for Computational Earth System Science, University of California: Santa Barbara, CA, USA*, 2006.

[43] D. Sulla-Menashe and M. Friedl, "Mcd12q1 modis/terra+aqua land cover type yearly l3 global 500m sin grid v006," NASA EOSDIS LP DAAC, 2015. [Online]. Available: https://lpdaac.usgs.gov/dataset_discovery/modis/modis_products_table/mcd12q1_v006

[44] D. Sulla-Menashe and M. A. Friedl, "User guide to collection 6 modis land cover (mcd12q1 and mcd12c1) product," *Department of Geography and Environment, Boston University*, 2018.

[45] W. J. Murphy. (1993) Tables for weights and measurements: Crops. MU Extension. [Online]. Available: https://extension2.missouri.edu/G4020

[46] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[47] M. Hutson, "Artificial intelligence faces reproducibility crisis," 2018.

[48] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[49] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[50] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[51] J. M. Omernik, "Ecoregions of the conterminous united states," *Annals of the Association of American geographers*, vol. 77, no. 1, pp. 118–125, 1987.

[52] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[54] R. Socher, C. Xiong, and K. S. Tai, "Three-dimensional (3d) convolution with 3d batch normalization," Feb. 16 2017, uS Patent App. 15/237,575.

[55] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[56] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, pp. 462–466, 1952.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[58] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[59] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.