

Deep Learning Tools for Yield and Price Forecasting Using Satellite Images

by

Mohamed Sadok Gastli

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2021

© Mohamed Sadok Gastli 2021

Author's Declaration

I declare that this thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Four publications have resulted from the work presented in the thesis:

1. M. Chaudhary, M. S. Gastli, L. Nassar, and F. Karray. Deep Learning Approaches for Forecasting Strawberry Yields and Prices Using Satellite Images and Station-Based Soil Parameters. In *Association for the Advancement of Artificial Intelligence (AAAI) Spring Symposium 2021*, 2021, CEUR-WS.org, <http://ceur-ws.org/Vol-2846/paper10.pdf>
2. M. S. Gastli, L. Nassar, and F. Karray. Satellite Images and Deep Learning Tools for Crop Yield Prediction and Price Forecasting. (Accepted by International Joint Conference on Neural Networks 2021)
3. M. Chaudhary, M. S. Gastli, L. Nassar, and F. Karray. Transfer Learning Application for Berries Yield Forecasting using Deep Learning. (Accepted by International Joint Conference on Neural Networks 2021)
4. M. S. Gastli, L. Nassar, and F. Karray. Deep Learning Models for Strawberry Yield and Price Forecasting Using Satellite Images. (Submitted to IEEE International Conference on Systems, Man, and Cybernetics (SMC) 2021, submission #728)

In paper 1, I was responsible for satellite images data preprocessing and building the models and using them to forecast both yield and price. The work in this paper forms part of Sections 4.3 and 4.4.

For paper 2, I was responsible for the entire work including the literature review, data preprocessing, and experiments. This paper's content forms Sections 4.1 and 4.2.

For paper 3, I performed data preprocessing and TL implementation using satellite images to forecast both yield and price. The content of this paper forms Section 4.6.

For paper 4, I was responsible for the entire work including the literature review, data preprocessing, and experiments to forecast both yield and price. The content of this paper forms part of Sections 4.3 and 4.4.

Abstract

The ability to forecast crop yields and prices is vital to secure global food availability and provide farmers, retailers, and consumers with valuable information to maximize effectiveness. Conventional approaches used to tackle this often use localized methods that are expensive and limited in generalizability. To tackle some of these known issues and to benefit from recently developed advanced tools of machine learning, this thesis explores the use of deep learning models as well as satellite images to forecast various crop yields and prices across the USA. The special case of the USA was chosen given the abundance of datasets pertaining to weather and agricultural information. Moreover, the thesis explores Transfer Learning (TL) and incremental learning applications in the field for generalizability. In addition, a web application along with a user-friendly interface are designed and implemented to facilitate the ease of user application of the proposed models and approaches.

Multiple machine learning models, specifically those based on artificial neural networks, are deployed and tested, along with several voting regressor ensembles. The models are tested using satellite images for California and the Midwest in USA to predict soybean yield and forecast strawberry and raspberry yield and price. Dimensionality reduction is applied by converting those satellite images into histograms that represent the pixel value frequency count. To gauge the performance of the deployed models, several evaluation metrics are used including Mean Absolute Error (MAE), Root Mean-Squared Error (RMSE), R-Squared Coefficient (R^2), as well as Aggregated Measure (AGM) and their Average Aggregated Measure (AAGM).

The potential of using deep learning based models in real-life applications which provides crucial insight for all stakeholders in the field of agriculture is demonstrated in this work. The deployed multi-module based models and voting regressors ensembles proved to have higher performance compared to the single module models. The proposed CNN-LSTM is found to outperform Convolutional Neural Network (CNN) models proposed in the literature by an average RMSE percentage improvement of 31% while the inclusion of the satellite images of surface and subsurface moisture levels enhances the prediction performance. In addition, it is observed that all deployed models consistently lose forecasting performance the further they forecast in the future, with the CNN-LSTM Ensemble outperforming each of its components as well as the LSTM in yield forecasting while the CNN-LSTM outperforms the LSTM in price forecasting. Moreover, the proposed CNN-LSTM-SAE Ensemble outperforms the deployed CNN-LSTM, VAE, and SAE models including the literature CNN model by 70% AGM improvement for yield forecasting and 66% for price forecasting. The deployment of incremental learning with the CNN-LSTM

Ensemble for yield forecasting without drastic loss in performance is achieved. Finally, based on the AGM metric, it is found that the TL CNN-LSTM outperforms the non-TL CNN-LSTM model by almost 28% AGM with reduction of 49% in computational time. For future work, there is potential in expanding the utilized datasets and models to verify and improve the obtained results as well as investigating the performance on additional fresh produce and counties to better gauge and enhance the effectiveness of the models and application.

Acknowledgements

I would like to thank my supervisor Professor Fakhri Karray for all the guidance and support given to me throughout my Master's degree as well as the research opportunities. In addition, I want to thank my colleagues and team members of this project: Dr. Lobna Nassar and Mohita Chaudhary. I would like to acknowledge Dr. Nassar's contribution to experiment ideas, result compilation, application design, documentation, and revision. I would also like to thank Mohita Chaudhary for her help with experiment ideas, documentation and revision.

I would also like to acknowledge and thank the members of my thesis committee Prof. Kumaraswamy Ponnambalam and Dr. Apurva Narayan for taking time to provide a detailed review and vital feedback of the thesis. Their assessment and evaluation are very appreciated.

I acknowledge and am thankful for the financial support from Loblaw Companies Limited, University of Waterloo, and its Electrical and Computer Engineering department.

I am most grateful for my parents, Mrs. Ilhem Sassi Gastli and Prof. Adel Gastli, for their unlimited and unfaltering love, support, guidance, and sacrifices that allowed me to be where I am today. I am also grateful for the love and support of my sisters Leila Gastli and Kaouther Gastli.

Finally, I present my sincere gratitude for all of my friends who were there for me when times were tough; I would not be here without them. I am obliged to mention some by name: Yasmine, Lina, Melek, Safiya, and Abdelrahman.

Dedication

This thesis is dedicated to my parents, friends, and those who helped me push through until the end.

Table of Contents

List of Figures	xii
List of Tables	xv
List of Abbreviations	xvi
1 Introduction	1
1.1 Problem Definition	1
1.2 Motivation	2
1.3 Scope	2
1.4 Objective and Deliverables	3
1.5 Thesis Organization	3
2 Background and Literature Review	4
2.1 Deep Learning Tools	4
2.1.1 Gaussian Process (GP)	5
2.1.2 Long Short-Term Memory (LSTM)	5
2.1.3 Convolutional Neural Networks (CNN)	8
2.1.4 CNN-LSTM	10
2.1.5 Stacked Autoencoder (SAE)	10
2.1.6 Variational Autoencoder (VAE)	11

2.1.7	Voting Regressors	12
2.1.8	Incremental Learning	13
2.1.9	Transfer Learning	14
2.2	Literature Review	14
2.2.1	Time Series Modelling	15
2.2.2	Time Series Forecasting	17
2.2.3	Satellite Images in Forecasting	19
2.2.4	Work in Incremental Learning	20
2.2.5	Transfer Learning	21
2.3	Evaluation Metrics	21
2.3.1	Mean Absolute Error (MAE)	22
2.3.2	Root Mean Squared Error (RMSE)	22
2.3.3	Coefficient of Determination (R^2)	22
2.3.4	Aggregated Measure (AGM)	23
2.3.5	Average AGM (AAGM)	24
2.4	Chapter Summary	24
3	Methodologies and Proposed Solutions	25
3.1	Datasets	26
3.1.1	Midwestern USA Dataset	26
3.1.2	California Dataset	28
3.2	Data Preprocessing	30
3.2.1	Preprocessing and Masking the Satellite Images	30
3.2.2	Conversion to Histograms	32
3.2.3	Forming the Final Data Structure	34
3.3	The Proposed Models	37
3.3.1	LSTM	37
3.3.2	GP	37

3.3.3	CNN-LSTM	38
3.3.4	CNN-LSTM Ensemble (CNN-LSTM_Ens)	39
3.3.5	SAE	39
3.3.6	VAE	40
3.3.7	SAE-CNN-LSTM Ensemble (SAE-CNN-LSTM_Ens)	41
3.4	Evaluation Metrics Units	42
3.5	Hyperparameter Tuning	42
3.6	Chapter Summary	43
4	Experiments and Results Analysis	44
4.1	Annual Prediction in Multiple Counties	44
4.1.1	Predicting Yield with Surface Reflectance and Temperature using data from 2003 to 2015	45
4.1.2	Predicting Yield with Surface Reflectance, Temperature, and Moisture using data from 2010 to 2015	46
4.2	Daily Forecasting in a Single County	49
4.2.1	Exploring Daily Strawberry Yield Forecasting Windows	49
4.2.2	Exploring the Parameters' Significance for Price Forecasting	50
4.3	Daily Yield Forecasting in Multiple Counties	52
4.4	Daily Price Forecasting in Multiple Counties	55
4.5	Incremental Learning for Yield Forecasting	58
4.6	Transfer Learning for Yield Forecasting	60
4.7	Chapter Summary	63
5	Web Application	65
5.1	Design and Implementation	65
5.2	Testing	66
5.3	Chapter Summary	74

6 Conclusions and Future Work	75
References	76

List of Figures

2.1	Illustration of an RNN unrolled over time steps [71]	6
2.2	Comparative diagram of (a) a basic RNN and (b) an LSTM [45]	7
2.3	Schematic diagram of a CNN architecture [79]	9
2.4	Schematic diagram of the architecture of an autoencoder[97]	11
2.5	Schematic diagram of the incremental learning approach	13
2.6	Schematic diagram of the network-based transfer learning approach	15
2.7	Illustration of converting an image into its histogram representation	20
3.1	Sample satellite images of soil surface reflectance (left) and temperature (right) in Douglas, Kansas	27
3.2	Sample satellite images of soil moisture in Douglas, Kansas	28
3.3	Sample satellite images of soil surface reflectance (left) and temperature (right) in Santa Barbara, California	29
3.4	Sample satellite images of soil moisture in Santa Barbara, California	30
3.5	Strawberry yield time series in Santa Barbara, California	31
3.6	Sample of land cover used for masking of Santa Barbara, California	32
3.7	3D histogram collection per sample for a given year at a given county in the Midwest	33
3.8	3D histogram collection per sample for daily forecasting in California	34
3.9	Mapping 3D county histogram collection per year (Input) to the corresponding county yield of same year (Output)	35

3.10	Mapping 3D histogram collection (Input) to the corresponding daily yield or price (Output) in California	36
3.11	LSTM architecture deployed in [112]	37
3.12	LSTM architecture for the California experiment	38
3.13	CNN-LSTM architecture for yield forecasting in the Midwest	39
3.14	CNN-LSTM architecture for yield and price forecasting in California	39
3.15	Block diagram of CNN-LSTM_Ens	40
3.16	SAE architecture for yield & price forecasting using satellite images	40
3.17	VAE architecture for yield & price forecasting using satellite images	41
3.18	Block diagram of the proposed CNN-LSTM-SAE voting regressor model	42
4.1	AAGM of tested models with & without moisture	48
4.2	Performance based on AGM of LSTM, CNN-LSTM, and CNN-LSTM_Ens models for forecasting yield in Santa Barbara for multiple days ahead	50
4.3	Forecasted LSTM, CNN-LSTM, and CNN-LSTM_Ens vs. true yield values	51
4.4	Forecasted values of price using LSTM and CNN-LSTM models versus true price values	52
4.5	AGM percentage improvement in yield forecasting of the CNN-LSTM-SAE Ensemble to the CNN benchmark model, VAE, and its two component models: SAE and CNN-LSTM	55
4.6	AGM percentage improvement in forecasting price of the CNN-LSTM-SAE Ensemble compared to the CNN benchmark model, VAE, and its two component models: SAE and CNN-LSTM	58
4.7	AGM forecasting scores of CNN-LSTM_Ens model in 2019 when trained using the incremented yearly data of 7 years from mid 2011 to 2018	59
4.8	Forecasted yield values by CNN-LSTM_Ens model and its two CNN-LSTM components compared to true yield values in 2019	60
4.9	Comparison between strawberry and raspberry yields from 2011 to 2019	61
4.10	The true raspberry yield values versus the yields forecasted by (a) model pretrained on strawberry yield, (b) TL model, and (c) model trained on raspberry yield without TL	63

5.1	Web application design flowchart	66
5.2	Application screenshot showing the user input for test case 1	67
5.3	Screenshot showing the application output for test case 1	67
5.4	Application screenshot showing the user input for test case 2	68
5.5	Screenshot showing the application output for test case 2	68
5.6	Application screenshot showing the user input for test case 3	69
5.7	Screenshot showing the application output for test case 3	69
5.8	Application screenshot showing the user input for test case 4	70
5.9	Screenshot showing the application output for test case 4	70
5.10	Application screenshot showing the user input for test case 5	71
5.11	Screenshot showing the application output for test case 5	72
5.12	Application screenshot showing the user input for test case 6	73
5.13	Screenshot showing the application output for test case 6	73

List of Tables

3.1	Number of counties in the Midwest with available yields from 2003 to 2015	35
4.1	AGM and AAGM results for models tested on years 2003 to 2015	45
4.2	Models' performance based on the AGM and AAGM for years 2010 to 2015 without considering the moisture parameter	46
4.3	Models' performance based on the AGM and AAGM for years 2010 to 2015 after considering the moisture parameter	47
4.4	AGM and AAGM percentage improvement compared to worst performing model	47
4.5	RMSE improvement compared to literature model	48
4.6	Results for forecasting prices using LSTM and CNN-LSTM	51
4.7	Yield forecasting performance of CNN, VAE, CNN-LSTM, SAE, & CNN-LSTM-SAE Ensemble models	53
4.8	CNN-LSTM-SAE Ensemble yield forecasting AGM percentage improvement to the state-of-the-art models across three counties	54
4.9	Price forecasting performance of CNN, VAE, CNN-LSTM, SAE and CNN-LSTM-SAE Ensemble	56
4.10	CNN-LSTM-SAE Ensemble price forecasting AGM percentage improvement to the state-of-the-art models across three counties	57
4.11	Evaluation metrics scores of CNN-LSTM using TL compared to non-TL approaches	62

List of Abbreviations

- AAGM** Average Aggregated Measure [24](#)
- AGM** Aggregated Measure [23](#)
- ANN** Artificial Neural Network [4](#)
- CNN** Convolutional Neural Network [9](#)
- DL** Deep Learning [2](#)
- FIPS** Federal Information Processing Standards [66](#)
- FP** Fresh Produce [1](#)
- GP** Gaussian Process [5](#)
- LSTM** Long Short-Term Memory [7](#)
- MAE** Mean Absolute Error [22](#)
- ML** Machine Learning [2](#)
- MODIS** Moderate Resolution Imaging Spectroradiometer [26](#)
- R²** Coefficient of Determination [23](#)
- RMSE** Root Mean Squared Error [22](#)
- RNN** Recurrent Neural Network [6](#)

SAE Stacked Autoencoder [11](#)

TL Transfer Learning [14](#)

USDA United States Department of Agriculture [34](#)

VAE Variational Autoencoder [12](#)

Chapter 1

Introduction

1.1 Problem Definition

Agricultural industries, specifically Fresh Produce (FP), are one of the most vital sectors for any country in terms of both food security and economic growth. On a global scale, the United Nations World Food Programme reported that in 2019 approximately 821 million people around the world have insufficient access to food for consumption [101]. As a result, the United Nations has made ending hunger and improving food security one of their top priorities in their 2030 Agenda for Sustainable Development [102].

The issue with food security goes further beyond the insufficient production of crops. Studies conducted by the Food Marketing Institute (FMI) and the Retail Control Group reveal that about 64% of food store items in the USA end up not being sold due to inefficient store operations, 14% of which are due to ordering inefficiencies, while 11% are due to inadequate production planning [85]. To investigate such issues, a survey was conducted in [58] to explore FP supply chain management. One of their main conclusions is that the lack of reliable forecasting tools plays a major role in hindering the goal of achieving food security. This is because unreliable conclusions regarding crop yields lead to unreliable FP procurement plans, which often leads to the underestimation of production where excess FP is not accounted for and is disposed of, hence food wastage. The limitations of forecasting tools are largely due to the lack of data as well as the complicated and often non-linear dependencies of real-life variables that are difficult to quantify and isolate for accurate forecasting.

1.2 Motivation

In Canada, the agricultural industry is one of the largest industry sectors with a realized net income of \$4.9 billion in 2019 as reported by Statistics Canada; this is a 10.4% increase compared to 2018 [96]. In addition, the industry is continuously growing as noted by the report of the Canadian Census of Agriculture in 2016 which indicated a tripling of the average field crops acreage over the previous 50 years [95]. This raises the need to have reliable, efficient, scalable, and generalizable forecasting models to minimize losses and sustain food security.

Machine Learning (ML) models can help tackle this problem, as they excel in using the available data to extract relationships with the output that are not explicitly modelled. They use historical input data to extract useful features and patterns from them, which they then use to forecast the required output. The input variables are presumed to be independent, allowing the dependent variable to be predicted as an output. Within ML, Deep Learning (DL) models are capable of capturing non-linear relationships among variables allowing for more accurate forecasting capabilities.

For such models to reach their full potential, sufficient data must be used for training. Data availability is a major limitation in ML, and especially for FP forecasting since agricultural data collection tends to be localized. This can be addressed by leveraging remote sensing data, specifically satellite imagery, that can cover large geographic areas. Satellite images provide a global data source that ML models can train on to learn the patterns in a crop's life cycle, which is essential for forecasting its yields and prices. Moreover, the use of remote sensing data, such as satellite images, minimizes the need for costly, hand-crafted, localized, and often time-consuming data collection methods.

1.3 Scope

This work focuses on developing effective DL based models for forecasting yields and prices. For this research, soybeans, strawberry and raspberry crops are investigated in the Mid-western and Californian counties since they are the leading producers of mentioned crops, and their data for yields, prices, and soil parameters are publicly available from online sources.

1.4 Objective and Deliverables

The main objective of this work is on developing complex and accurate computational models to forecast both yields and prices of crops using satellite images. To reach this main objective, the following tasks are performed:

1. Exploring existing literature to find the most suitable approach.
2. Identifying, collecting, and processing yields, prices, and satellite images data relevant to the crops being investigated.
3. Designing, tuning, and implementing several DL models for forecasting yield and price.
4. Evaluating models by comparing them to existing models proposed in literature.
5. Exploring the generalization and scalability of the built models.
6. Building a user friendly web application to automate the forecasting process for the main users such as the FP procurers.

1.5 Thesis Organization

This thesis is divided into five chapters. The current chapter provides an overview of the problems faced in the FP procurement process and the motivation behind tackling them. It then outlines the scope as well as the objectives of this work. Chapter 2 provides the background information and literature review necessary for the work. It describes FP procurement, time series modelling, satellite images, the deep learning models, and the literature review. Chapter 3 presents the proposed solution, including the datasets used, preprocessing techniques applied, models proposed, evaluation metrics used, as well as the process of choosing the most effective parameters. In Chapter 4, the conducted experiments conducted are described and the results are presented and analyzed. Chapter 5 presents the work done to design and implement a web application that forecasts similar crops at different counties using the approaches investigated. Finally, Chapter 6 summarizes the main findings obtained from the work and proposes potential improvements and future work.

Chapter 2

Background and Literature Review

This chapter explores the existing background and literature concerning the use of remote sensing and DL tools in forecasting FP yield and price. Moreover, it presents conventional approaches used in modelling different time series classes. The chapter also explores techniques used for preprocessing satellite images, incremental learning, and Transfer Learning (TL). In addition, it presents a description of different evaluation metrics used for time series forecasting.

2.1 Deep Learning Tools

DL is a branch of ML that is broadly inspired by the neural structures of the brain. DL models utilize a hierarchical structure of nodes that are interconnected, similar to brain neurons, allowing them to process and pass data thus learning data representations. While non-deep ML approaches are effective in many applications, they typically falter when the data representation is complex and not optimized. They typically require raw data to be engineered beforehand, which needs expertise in the applied domain, to optimize the extraction of features; and that is often a time-consuming and expensive process. The hierarchical structure of DL models allows them to transform raw data into their own optimized representations, making them more adaptive to unstructured and raw data; provided the dataset is large enough and the models are deep enough.

Research into DL techniques is a broad and ongoing field, but the main foundation of the models are Artificial Neural Networks (ANN). ANNs are made up of nodes that form a directed hierarchical structure that takes input data and outputs the target variable.

2.1.1 Gaussian Process (GP)

The Gaussian Process (GP) is defined as a collection of random variables of a non-parametric probability distribution over certain functions, such that each finite subset of said functions follows a Gaussian distribution [83]. It was first introduced to solve regression problems by C. Williams and C. Rasmussen in [107]. GP operates by taking a prior distribution and updating it as it observes more data points, producing the posterior distribution over functions. The linear GP model used is defined in Equation 2.1.

$$g(x) = f(x) + h(x)^T \beta \quad (2.1)$$

where $x \in \mathbb{R}^d$, $f(x) \sim GP(0, k(x, x'))$ is a GP that models the residuals of the linear model, $h(x)$ is the feature vector of the deep model that represents a fixed set of basis functions, and β is an independent random variable with a Gaussian prior. The squared exponential kernel is typically used for the kernel function $k(x, x')$ as described in [112], and is defined in Equation 2.2.

$$k_{SE}(x, x') = \sigma^2 \exp\left(-\frac{\|x - x'\|_2^2}{2r^2}\right) \quad (2.2)$$

where $\|\cdot\|_2$ is the L^2 norm whilst σ and r are hyperparameters selected by performing a grid search during training. Since real-life observations usually contain noise; a Gaussian noise term is included in the covariance term as shown in Equation 2.3.

$$k_{SE}(x, x') = k_{SE}(x, x') + \sigma_e^2 \delta_{x, x'} \quad (2.3)$$

where $\delta_{x, x'}$ is the Kronecker delta, defined in Equation 2.4, and σ_e^2 is the variance of the Gaussian noise [83].

$$\delta_{x, x'} = \begin{cases} 0 & \text{for } x \neq x' \\ 1 & \text{for } x = x' \end{cases} \quad (2.4)$$

The GP is typically used to take into account spatio-temporal dependencies between datapoints provided the needed information.

2.1.2 Long Short-Term Memory (LSTM)

While ANNs are suitable for a variety of applications, their effectiveness is limited when dealing with sequence dependent data. Recurrent Neural Networks (RNN) are a type of models characterized by their ability to retain an internal state and use it to process data,

behaving like memory. It allows this class of ANNs to learn features that are dependent on a sequence, typically a sequence in time or words, making it suitable for time series modelling applications [94] as well as text [67, 62] and speech recognition [27].

Given an input x_t at a current time t and a node hidden state h_{t-1} from the previous iteration $t-1$, a recurrent node calculates the current hidden state h_t as shown in Equation 2.5.

$$h_t = \sigma_h(W_{hx}x_t + W_{hh}h_{t-1} + b_h) \quad (2.5)$$

where σ_h is an activation function, specifically a sigmoid function, which maps its input into a value between 0 and 1, W_{hx} is the matrix of weights between the input and hidden state, W_{hh} is the matrix of weights between the hidden layer and itself at an adjacent time step, and b_h is the bias parameter used to control the network offset to better fit the data [51].

After the hidden state is obtained, it is used to calculate the output y_t as presented in Equation 2.6.

$$y_t = \sigma_y(W_{yh}h_t + b_y) \quad (2.6)$$

where σ_y is an activation function, W_{yh} is the matrix of weights between the output and the hidden layer state, and b_y is the bias parameter used to offset the network to best fit the data [51].

The weights and parameters of the network are updated using backpropagation across the different time steps, this algorithm is termed Backpropagation Through Time (BPTT). A great visualization of the unrolled structure of an RNN is presented in Figure 2.1, with the recurrent learning across time steps occurring as information travels from left to right.

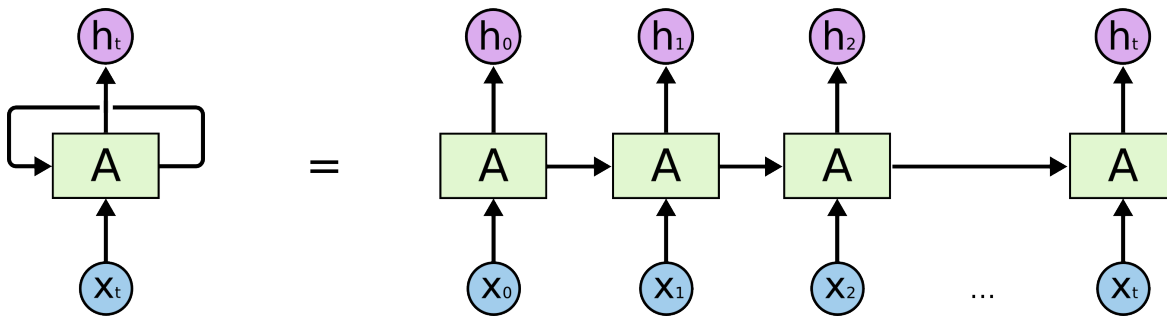


Figure 2.1: Illustration of an RNN unrolled over time steps [71]

An issue that arises with using backpropagation in general is that if the gradient is small, the Chain Rule multiplication would compound the effect. This causes the gradient

to shrink to the point of vanishing, stopping the network from learning any further. This phenomenon is commonly referred to as the vanishing gradient problem, as is common in many neural network architectures. A large gradient would also cause a compounding effect, causing the gradient to explode and become unstable resulting in what is referred to as the exploding gradient problem. It is also apparent with RNNs where each node has deep recurrent iterations through time. A specific subclass of RNNs was made to combat this issue, suitably named Long Short-Term Memory (LSTM). The LSTM network was first introduced by S. Hochreiter and J. Schmidhuber in [31]. It is a type of RNNs that allows for longer short-term memory storage, making it capable of capturing dependencies over longer sequences. Figure 2.2 presents a comparison between the structure of RNNs and LSTM.

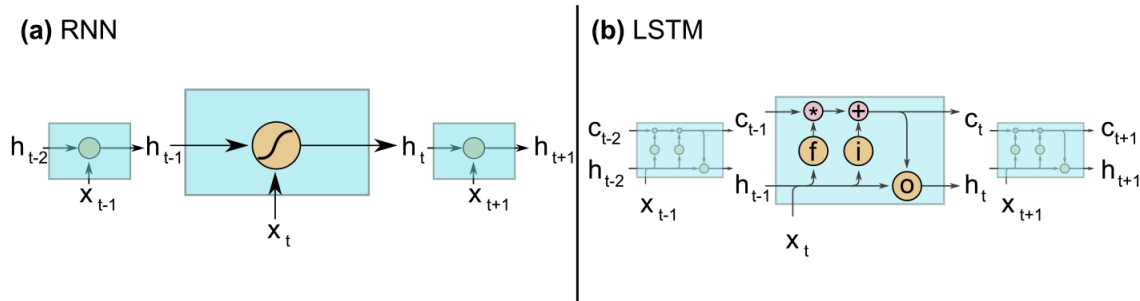


Figure 2.2: Comparative diagram of (a) a basic RNN and (b) an LSTM [45]

Compared to RNNs, LSTM networks contain an additional cell memory termed c_t , which stores information, and three gates: the input gate i , the forget gate f , and the output gate o . These gates control the flow of information within the LSTM cell. The forget gate controls how much information is forgotten from the previous cell state c_{t-1} , and is often applied using a sigmoid function. It is mathematically represented in Equation 2.7.

$$f_t = \sigma_f(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \quad (2.7)$$

where σ_f is the sigmoid activation function, W_{fx} is the matrix of weights between the input and forget gate, W_{fh} is the matrix of weights between the forget gate and its previous hidden state, and b_f is the bias parameter used to control the forget gate offset.

Then, the potential update vector for the cell state \vec{c}_t is calculated from the current input and previous hidden state using Equation 2.8.

$$\vec{c}_t = \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \quad (2.8)$$

where \tanh is the hyperbolic tangent activation function and b_c is the bias parameter used to control the offset of the update vector.

The input gate is then used to control how much of the update vector \vec{c}_t is used based on the input vector. It is computed as presented in Equation 2.9.

$$i_t = \sigma_i(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \quad (2.9)$$

where σ_i is the activation function, W_{ix} is the matrix of weights between the input and input gate, W_i is the matrix of weights between the input gate and its previous hidden state, and b_i is the bias parameter used to control the input gate offset.

The update vector \vec{c}_t , tuned by the input gate, is then combined with the previous cell state c_{t-1} , tuned by the forget gate, to produce the cell state c_t as shown in Equation 2.10.

$$c_t = f_t * c_{t-1} + i_t * \vec{c}_t \quad (2.10)$$

Finally, the output gate is used to control the current hidden state as in Equation 2.11.

$$o_t = \sigma_o(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \quad (2.11)$$

where σ_o is the activation function, W_{ox} is the matrix of weights between the input and output gate, W_o is the matrix of weights between the output gate and its previous hidden state, and b_o is the bias parameter used to control the output gate offset.

The current hidden state h_t is then computed using Equation 2.12.

$$h_t = o_t * (\tanh c_t) \quad (2.12)$$

The output y_t is lastly calculated using the current hidden state as in Equation 2.13.

$$y_t = \sigma_y(W_{yh}h_t + b_y) \quad (2.13)$$

where σ_y is an activation function, W_{yh} is the matrix of weights between the output and the hidden layer state, and b_y is the bias parameter used to offset the output [45].

2.1.3 Convolutional Neural Networks (CNN)

Convolutional Neural Networks are a class of ANNs that specialize in extracting features from images. They were first introduced by K. Fukushima in [24], who was inspired

by the biological behavior of receptive fields. This class of models utilizes layers that perform convolution on input data by extracting spatial dependencies between different pixels. Moreover, CNNs are considered a regularized version of fully connected networks, which are characterized by having each neuron in one layer connected to all neurons in the next layer. This excessive web of connections makes fully connected networks prone to losing their generalization of data by fixating on fitting perfectly to the training data, a problem termed overfitting. CNNs tackle this issue by using filters that simplify the data representation the deeper the network goes, forcing the network to discard information that is too specific to the training data [26]. CNN applications span various fields, including image classification [111, 65], time series analysis [89], and recommendation systems [21]. Figure 2.3 clearly illustrates the typical structure of a CNN; which is used for object recognition in [79].

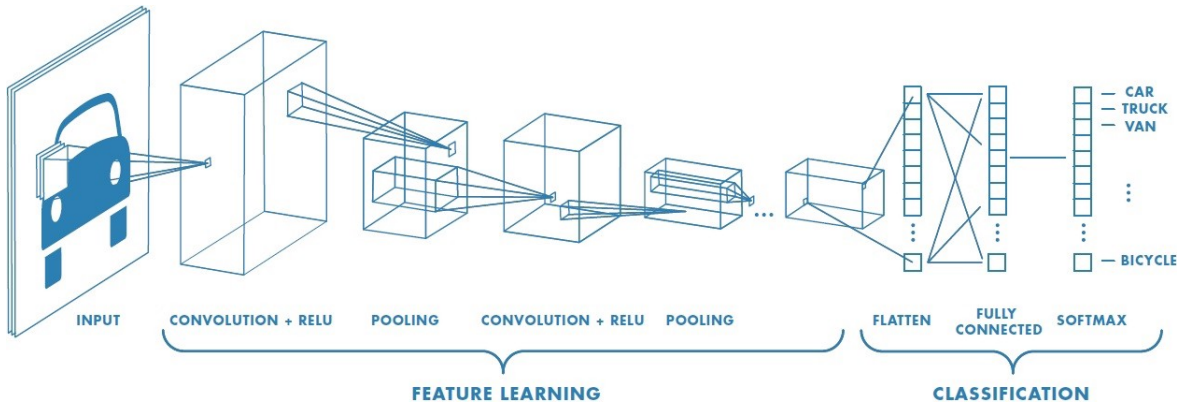


Figure 2.3: Schematic diagram of a CNN architecture [79]

CNNs, like most networks, consist of an input and output layer, as well as multiple hidden layers. These hidden layers, as shown in Figure 2.3, are typically convolutional layers, pooling layers, fully connected layers, and normalization layers. The layers specific to CNNs are described as follows:

- Convolution: It is considered the building block of CNNs. This layer uses mathematical convolution instead of general matrix multiplication used by other networks. A matrix of values, referred to as a filter or kernel, is convoluted with the input data to transform it according to its values. The filter is typically smaller in dimensions than the input layer, thus it requires shifting after convolution with a region of the image; this shifting is known as the stride. The typical dimensions of the input images are (number of images x image length x image width x image depth), and

after convolution the output of the layer is (number of images x feature map length x feature map width x feature map depth x feature map channels) which is then passed through a nonlinear activation function, usually a ReLu function [84]. These dimensions vary depending on the dimensions of the input and the convolutional layer, as for 1-dimensional convolution layers the filter strides through a single dimension.

- Pooling: This type of layers is used to further reduce the dimensions of the convolution layers for regularization. They achieve that by taking a region of nodes in the feature map and combining them into a single value. This loss of information helps generalize the model and avoid overfitting. The combination is typically done by choosing either the maximum value in the region, known as max pooling, or the average value of the region, known as average pooling [84]. Furthermore, if the pooling region is smaller than the feature map it is called local pooling, otherwise it is called global pooling where the pooling region spans the entire feature map space [50]. Their uses depend on the nature of the data and how much regularization is needed in the model.

2.1.4 CNN-LSTM

LSTM networks excel at extracting time dependent features from data, but they fail in detecting spatial relations. On the other hand, CNNs are proficient in extracting spatially dependent features from data, but they fail in detecting temporal relations. A hybrid network was modelled to obtain the capabilities of both classes of ANNs, while offsetting their shortcomings, named CNN-LSTM. This class of models is specifically designed for sequence prediction problems that involve spatial inputs. A CNN-LSTM typically starts with convolution and pooling layers to capture spatial features and produce efficient vector representations, which are then fed into the LSTM layers for extracting the temporal features. CNN-LSTM models are successful in time series forecasting [52, 33], sentiment analysis [105] and speech recognition [114].

2.1.5 Stacked Autoencoder (SAE)

Autoencoders are a class of neural networks that focuses on finding an optimal feature representation. One of their early implementations was by Y. LeCun and F. Soulie Fogelman in [49] and by D. Ballard in [6]. The traditional use of autoencoders was in feature learning and dimensionality reduction, as they are characterized by having both an encoder and a decoder. The encoder is a block of layers that learns how to encode the input data

into a more concise representation. It achieves this by gradually decreasing the number of nodes as the network goes deeper, finally reaching the desired code vector. On the other hand, decoders operate in reverse, taking the encoded feature space and decoding it to replicate the original data to the best of its ability. This approach ensures the model learns to interpret the most important features of the data by restricting it to represent the training data in a limited feature space. An autoencoder that contains multiple hidden layers stacked one after the other is called a Stacked Autoencoder (SAE). The application of SAEs include medical imaging classification [100], remote sensing classification [115, 56], and time series forecasting [5]. Figure 2.4 illustrates the structure described earlier.

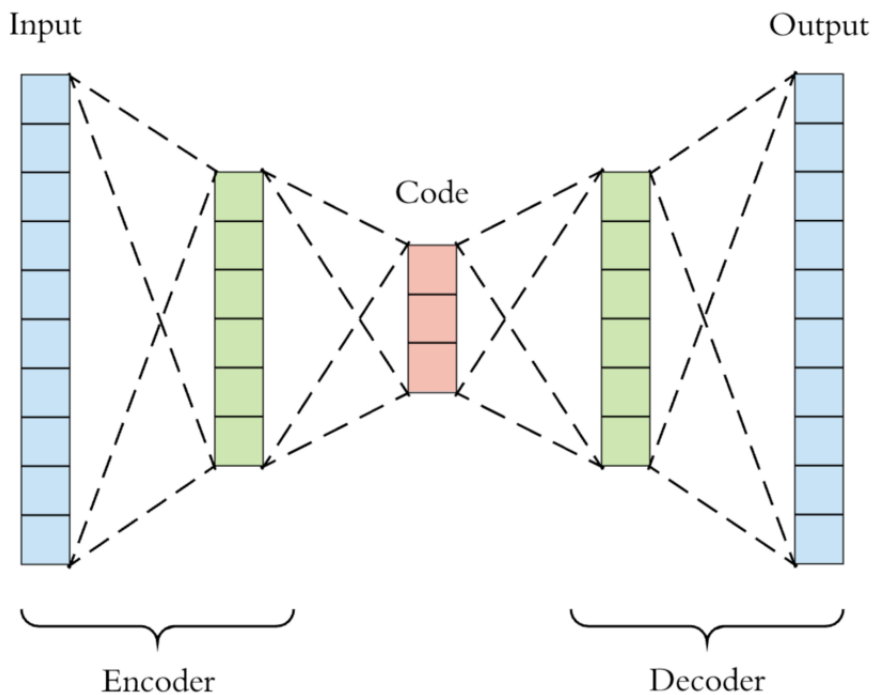


Figure 2.4: Schematic diagram of the architecture of an autoencoder[97]

2.1.6 Variational Autoencoder (VAE)

Conventional autoencoders map the input data into a fixed feature vector that are discrete in nature and might not be able to accurately describe the data. To tackle this, Variational Autoencoders (VAE), first introduced by D. Kingma and M. Welling in [44], try to map the input to a Gaussian distribution. They achieve this by using an encoder to try and

approximate the posterior distribution $p(z|x)$ defined in Equation 2.14, which is then used by the decoder to try to regenerate the original data from the learned distribution.

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} \quad (2.14)$$

where $p(x|z)$ is the likelihood of observing x given z , $p(z)$ is the prior probability distribution, and $p(x)$ is the evidence of data that was unobserved when computing the prior distribution.

In order to learn the behavior of $p(z|x)$, the distribution can be described solely through its mean μ and standard deviation σ . These parameters are what is learned in the bottleneck code vector presented in Figure 2.4. It allows for the model to reconstruct the data from continuous distributions, provided the assumption that the features follow a Gaussian distribution. VAEs are used in remote sensing classification [92] and anomaly detection [60].

2.1.7 Voting Regressors

Even though ML models are praised for their learning, many classes have a certain level of randomness in their behavior. Specifically for ANNs, this can be attributed to the stochastic nature of the algorithms used by the models to extract features. Despite its advantage of reducing the amount of processing required, it makes the models susceptible to converging to different solutions, often due to issues like overfitting. To reduce this variance in results, multiple instances of the models can be trained then their result are combined in a process called ensemble learning. Moreover, combining the effort of different models helps in improving the forecasting capabilities of the models as each model would extract different features, thus incorporating their efforts leads to a compounded enhancement in performance. The individual models used in an ensemble are typically termed the “weaker” models in relation to their more effective ensemble performance. Common ensemble methods include voting, bagging, and boosting for various applications [10]; the voting technique is deployed in this work to achieve better and more consistent results.

For classification problems, the voting method is simply choosing the majority voted label of the weaker models as the final output. For regression, it is called a voting regressor and is implemented by averaging the predicted results of the weaker models to compute the final output.

2.1.8 Incremental Learning

When forecasting time series, it is common for new data to be available over time, necessitating that the models keep up through periodic updates. This allows the models to stay up to date with the trends of the modelled data for more accurate forecasting. There are several approaches to update a model after training, depending on the frequency of data. Feeding a continuous and ordered stream of data for the continuous learning of a model is known as online learning, while a batch-based approach that requires less frequent learning is called incremental learning [25]. Due to the noncontinuous frequency of the data used in this work as well as memory limitations, incremental learning is investigated. Incremental learning can be achieved by sequentially feeding new batches of data into the model at a certain frequency, e.g. every 4 weeks. Figure 2.5 illustrates the approach, with the model instances being fed with a new batch of data every t time interval, while the previous batches are discarded.

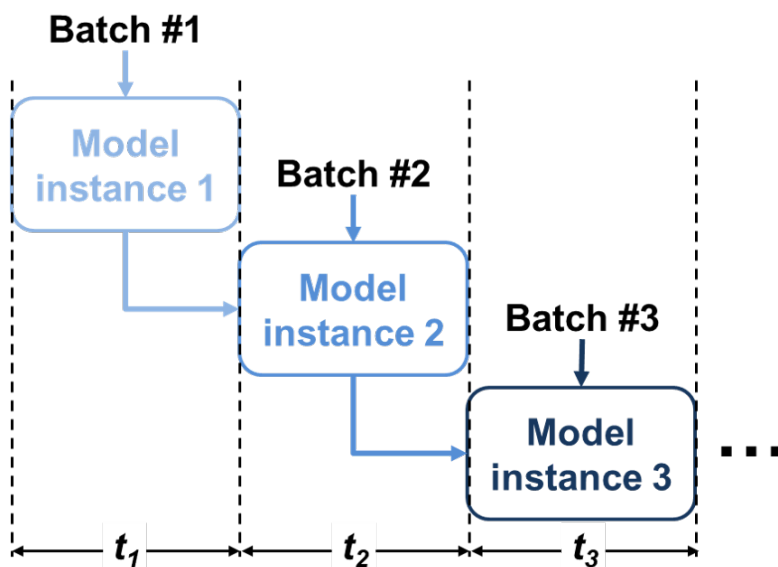


Figure 2.5: Schematic diagram of the incremental learning approach

A major drawback of this approach is that, given a new batch of data, over time the model could potentially start overwriting weights learned from a previous batch which can be crucial for forecasting. As neural networks have the freedom of prioritizing their own parameters, the learning is not directed; this phenomenon is referred to as catastrophic forgetting. Incremental learning is used for image classification [11, 78, 110, 53, 59], semantics segmentation [15], and automated annotation in video and speech tagging [34, 8, 14].

2.1.9 Transfer Learning

To create a generalized framework for different crops, forecasting models trained on one crop can be adapted to forecast related crops. This can be achieved by taking a pretrained model, or part of it, that is trained on one task and fine-tuning it to work on another related task; a technique termed Transfer Learning. TL approaches can be divided into 4 categories: First is the instances-based approach which chooses partial instances from the source domain as additions to the training set in the target domain by appointing suitable weights to the chosen instances. Second is the mapping-based approach which maps instances from two domains into a new feature space that enhances the similarity between them; this ensures the new feature space has combined knowledge from the original domains. Third is the network-based approach which partly reuses the network pretrained with the source domain in the target domain in an effort to transfer useful learned knowledge to it. Finally, the adversarial-based approach that uses adversarial technology, such as Generative Adversarial Networks (GAN), to locate features that are transferable and applicable for two domains [99]. For this thesis, the network-based approach is chosen for its effectiveness with DL models as suggested in [99]. Figure 2.6 illustrates the approach of network-based TL, with a section of the model pretrained in source domain A used in target domain B for knowledge transfer.

TL has been abundantly used for image classification [93, 98]; that is aided by the fact that ANNs do not necessarily assume the data is independent and identically distributed (i.i.d.), which is an assumption made by the stochastic approaches [54]. Moreover, TL has been used in various time series forecasting applications including flood prediction [43], traffic forecasting [46], and energy consumption forecasting [48].

2.2 Literature Review

The three main ML applications are investigated in this work: the time series modelling and forecasting, incremental learning, and transfer learning. Details of each of the three

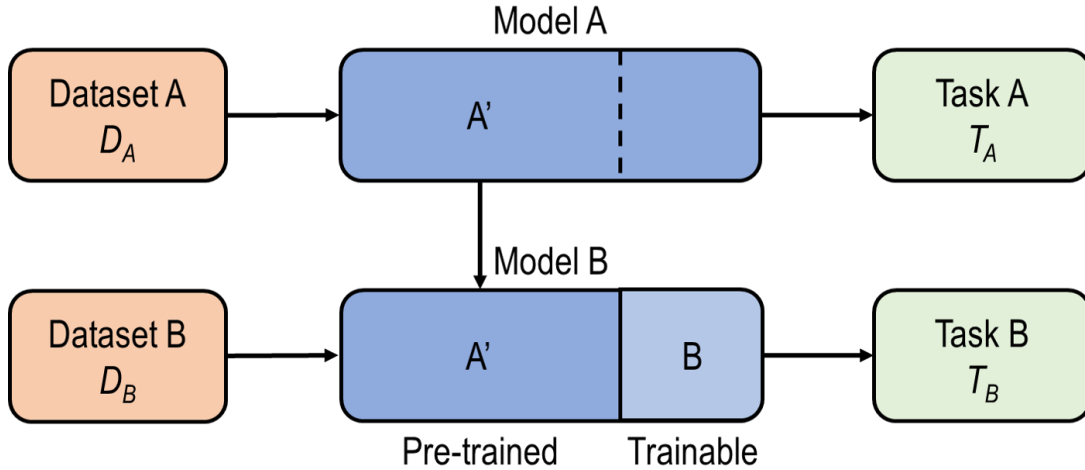


Figure 2.6: Schematic diagram of the network-based transfer learning approach

applications are covered in this section.

2.2.1 Time Series Modelling

A major class of data structures is time series, which is any sequence of observations x_t that are recorded at equally spaced time intervals over a period of time t [66]. Time series are used for representing data in almost every field that contains data variation through time. Two common conventional methods of modelling time series are Auto-Regressive Moving Average (ARIMA) and Seasonal ARIMA (SARIMA), which are classes of models that use past observations to model the time series to be able to forecast future observations. They use statistical characteristics of a time series to learn its behavior, such characteristics include the trend, seasonality, and stationarity. The trend of a time series describes how the average of the time series behaves over time, whether the time series is increasing or decreasing in the long-term. The seasonality describes the fixed periodic fluctuations in a time series, typically occurring in a daily, weekly, monthly, or yearly cycle. The stationarity describes whether the statistical properties, namely the mean and variance, of the time series are constant or varying; this usually describes whether the time series is converging or diverging [29]. These attributes of a time series help describe its nature, which in turn helps in forecasting its behavior in the future. Applications of ARIMA include forecasting disease outbreak trajectory [7], national inflation [68], and energy consumption [20]. Moreover, time series can be divided into two subclasses: univariate and multivariate time series.

Univariate Time Series

Univariate time series are characterized by containing single scalar observations. They are the simplest form of time series, where each observation x_t depends solely on a set of previous observations $x_{t-1}, x_{t-2}, \dots, x_{t-n}$ where n is the number of samples it depends on. ARIMA, SARIMA, and their component approaches are directly used for univariate time series since their trend, seasonality, and stationarity are easily captured and modelled. An example of an ARMA model for a univariate time series is described in Equation 2.15.

$$\hat{y}_t = \mu + \phi_1 y_{t-1} + \dots + \phi_p y_{t-p} - \theta_1 e_{t-1} - \dots - \theta_q e_{t-q} \quad (2.15)$$

where \hat{y}_t is the forecasted value, μ is the mean of the series, ϕ is the autoregression term with a maximum of p terms, θ is the error term with a maximum of q terms for a specified nonseasonal order of differencing d [30].

Multivariate Time Series

Multivariate time series are characterized by having multiple time dependent variables, meaning each variable has a dependency on other variables as well as itself. Since many natural, social, and economic behaviors are affected by multiple variables, multivariate time series provide a more realistic and accurate representation [30]. Multivariate time series are more challenging to model compared to univariate time series since multivariate dependencies are much more complex to model compared to describing an independent univariate time series based on itself. Moreover, the parameters needed to describe their relationships increase even further if the data contains nonlinear dependencies. This complexity makes it difficult to determine the best approaches in handling multivariate time series without experimental results. Conventional methods to tackle multivariate time series include Transfer Function-Noise (TFN), Contemporaneous ARMA (CARMA), and Vector Auto-Regression (VAR) [30, 29]. A single lag VAR(1) model can be defined as shown in Equation 2.16, with each equation modelling an input vector using the univariate autoregressive model.

$$\begin{aligned} \hat{y}_{1,t} &= \mu_1 + \phi_{11} y_{1,t-1} + \phi_{12} y_{2,t-1} + e_{1,t} \\ \hat{y}_{2,t} &= \mu_2 + \phi_{21} y_{1,t-1} + \phi_{22} y_{2,t-1} + e_{2,t} \end{aligned} \quad (2.16)$$

where $\phi_{ii,l}$ presents the influence of the l th lag of y_i on itself, $\phi_{ij,l}$ presents the influence of the l th lag of y_j on y_i , and $e_{1,t}$ and $e_{2,t}$ model the white noise process that may be correlated [36].

2.2.2 Time Series Forecasting

For yield forecasting, extensive research is done within literature. First, a survey conducted in [41] assessed the efficacy of different remotely sensed variables to forecast corn and soybean yields, both pre-season and in-season, within the USA. The author concluded that the Normalized Difference Vegetation Index (NDVI) and the land surface temperature have high correlation with the crop yields tested. The authors of [61] explored the capabilities of forecasting yield in Algeria using remote sensing data. The scope of their work was comparing machine learning approaches such as Random Forest and Multi-Layer Perceptron (MLP) to more non-machine learning (non-ML) approaches. Their results revealed that machine learning approaches consistently outperform Non-ML ones. Another notable work using remote sensing for yield prediction is authored by J. You et al in [112]. They use satellite images to predict yearly soybean yields for some counties in Midwestern states in the USA. The images utilized contain 7 bands for surface reflectance at different wavelengths and 2 bands for temperature at daytime and nighttime. They proposed and applied the dimensionality reduction approach described in Section 2.2.3, after which they fed the computed histograms into their tested models. The evaluated models are LSTM, CNN, LSTM with GP, and CNN with GP. The authors investigated the effect of adding GP to the ANN models. Their results indicate that the CNN outperforms the LSTM in predicting annual soybean yields. Moreover, they conclude that using GP helps further improve the overall performance of the CNN and LSTM. This work is further extended in [77], where the author investigated the addition of moisture data to the satellite image bands in addition to surface reflectance and temperature to predict yearly corn yields. The conversion to histograms is implemented in that work for tractability, and the models implemented are ConvLSTM, Separable CNN-LSTM, Custom CNN-LSTM, 3D-CNN, and CNN-LSTM-3D. The author concludes that ConvLSTM outperforms the other models in predicting annual corn yields. The approach's success is further cemented by the work in [90], where the authors explored the application in southern Brazil for the prediction of annual soybean yields. This work incorporated precipitation data acquired from satellite images as well as weather stations, combining remote and local data sources. The models they implemented are multivariate Ordinary Least Squares (OLS) linear regression, Random Forest, and LSTM. Their results show that the LSTM model is the best performing model in predicting annual soybean yields. From reviewing literature, the general sentiment is that ANNs, specifically LSTM, CNN, and their combinations, are well-suited for yield forecasting, which is supported by a literature survey conducted in [103]. Nevertheless, a notable limitation in [112], [88], [91], [77], [40], and [90] is that the implemented models are relatively simple and exploration of different classes of ANN is limited. In addition, they mostly focus on yearly yield prediction; this limits the scope of yield analysis

that could be applied which is often needed on a monthly, weekly, or even daily basis. Lastly, the literature extensively explores grain produce, while works on FP are limited. This work aims to tackle that by investigating new models for daily FP yield forecasting.

For price forecasting, Y. Peng et al. designed and implemented a service for FP price forecasting in [75] using historical prices. They compared several forecasting techniques including Partial Least Square (PLS), ARIMA, ANN, and the Response Surface Methodology (RSM). Moreover, the crops investigated are bok choy, cabbage, cauliflower, and watermelon. PLS and ANN are found to perform best with the lowest percentage errors out of the tested models. The authors suggested the use of PLS for short-term forecasting and ANN for long-term forecasting. A similar study is carried out in [82] where the authors utilized rainfall, crop yield, minimum support price, maximum trade value, seed cost, and cultivation cost as features for forecasting both price and profit. The models used are Naïve Bayes algorithm for price forecasting and K-Nearest Neighbor (KNN) for profit forecasting; promising results are found for both price and profit forecasting. It should be noted that their experiment is set up as a classification problem rather than a regression problem. More recent examinations of using DL for crop price forecasting are presented in [63] and [64], where strawberry prices in California are predicted using weather data collected from local stations. The models tested by the authors include ARIMA, XGBoost, CNN, LSTM, CNN-LSTM, and Attention-based CNN-LSTM. Their results indicate that DL models outperform the nondeep machine learning and time series analysis models and that Attention-CNN-LSTM outperforms its simpler DL counterparts in forecasting strawberry price. When reviewing the literature, several limitations arise. In both [75] and [3], the authors take into account only previous prices when forecasting price; assuming that the prices are univariate which is not necessarily true as various variables can affect the price. Moreover, the authors of [63] and [64] utilize localized data from stations to forecast prices; which runs the risk of limiting data availability to only locations that have stations, limiting the generalizability of the models to more global applications. This work aims to overcome that by utilizing remote sensing data in the form of satellite images.

Autoencoders are also used in forecasting time series, with literature supporting its efficacy. The authors of [113] tested a VAE model along with a simple RNN, LSTM, Bidirectional LSTM (BiLSTM), and a Gated Recurrent Unit (GRU) to forecast the number of new and recovered COVID-19 cases. The study focused on daily recorded cases in six countries which are China, Australia, Italy, Spain, USA, and France. Their conclusion is that VAE outmatches the other tested models in forecasting new and recovering COVID-19 cases. Another work in [42] investigates the effectiveness of VAE in forecasting renewable energy. The authors test several variations of VAE architecture with a combination of other DL models which are: Bayesian ANN, Bayesian BiLSTM, VAE-Bayesian

RNN, VAE-Bayesian LSTM, VAE-Bayesian BiLSTM, VAE-BiLSTM, and VAE-LSTM. Their results show that the VAE variants outperform the other models, which showcases the proficiency of VAE models in forecasting renewable energy. As for SAE models, the study conducted in [74] explores the ability of SAEs in using soil and weather parameters to predict yield and protein content of winter wheat. The authors implemented linear regression, nonlinear regression, ANN, and SAE, with the results suggesting that the best prediction is achieved by using SAE with spatial sampling. On top of that, Hu et al. and Preira et al. implemented SAE in [32] and [76] to forecast confirmed COVID-19 cases in both China and Brazil, showcasing encouraging results. Moreover, the application of SAE in forecasting wind speeds was explored in [39], where the authors proposed and tested a hybrid LSTM-SAE model. The proposed model was compared to Support Vector Regression (SVR), ANN, RNN, and LSTM. The major limitation in the investigated literature is the lack of work in using autoencoders in forecasting multivariate time series, specifically crop yield and price, which provides a lucrative venue for exploration tackled in this work.

2.2.3 Satellite Images in Forecasting

Satellite images typically provide abundant amounts of data for learning, however with that the challenge of processing such large amounts of data arises. ML models in general, and DL models in particular, are usually limited by their large computational demand which scales up when more data is used. Moreover, due to the remote and global nature of satellite images, their information can be spatially scarce. The most direct method of utilization is to feed those satellite images into the forecasting models to train on without modification. However, this approach is computationally expensive and not tractable with existing resources.

Therefore, methods of dimensionality reduction can help achieve tractability without sacrificing retained information. In the field of yield forecasting using remote sensing, the authors of [112] proposed a dimensionality reduction technique that was adapted by later publications in the field. Their approach is to convert each image from a two dimensional matrix into a single dimension histogram that counts the frequency of pixel values. This reduction is done under the assumption of permutation invariance, which means that the positions of the pixels within an image are assumed to carry little or no information relevant to yield and price forecasting, thus they can be discarded without losing important features. It is acknowledged that the forecasted yields can have some dependency on the positions of the pixels, but for tractability they must be dropped. Figure 2.7 visualizes the reduction for better clarification; given an image with values ranging from 0 to 0.5, the bins are divided into equidistant value ranges and each bin holds the number of pixels that lie within its

range. It should be noted that the values used in Figure 2.7 are only for clarification purposes, the actual values used in this work are described in Chapter 3.

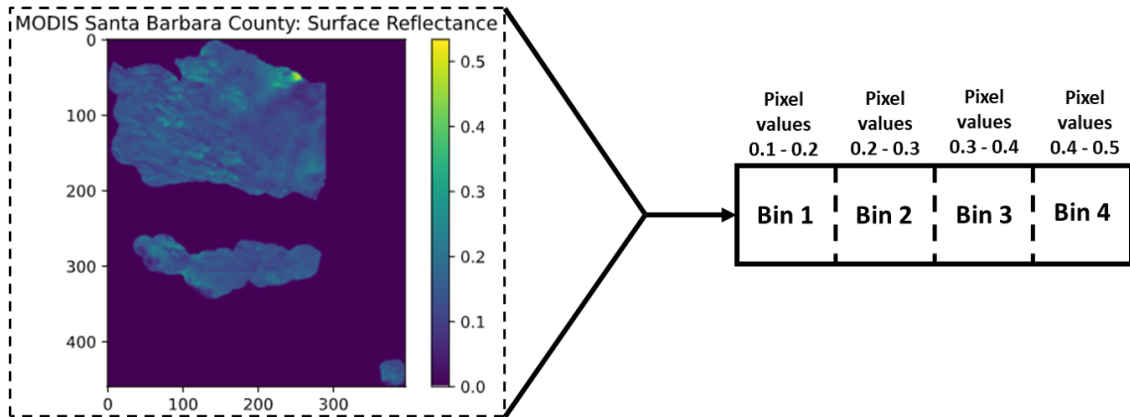


Figure 2.7: Illustration of converting an image into its histogram representation

2.2.4 Work in Incremental Learning

For incremental learning, the authors of [81] use incremental learning to train a Random Vector Functional Link (RVFL) network, a class of ANN, to forecast short-term electric load. They compared their proposed model to benchmark models like non-incremental ANN and random forest, among others, and one of their conclusions is that incremental learning is beneficial for short-term electric load forecasting. Moreover, a study of the effectiveness of incremental learning in stock price forecasting is presented in [80]. Their hybrid incremental RVFL is compared to similar benchmark non-incremental ANN, RVFL, SVR, and other models, with the results indicating that incremental learning provides significant improvement to the short-term stock price forecasting performance. In addition, the authors of [53] compare the performance of several incremental learning approaches on CNN networks used for two classification problems. Their results present successful adaptation of incremental learning on both datasets investigated. Several applications of incremental learning, such as in [11] and [78], modify the algorithms of machine learning models to adapt to an incremental feed of inputs which is typically used to add new classes to a model in a classification problem. Moreover, it is indicated in literature that incremental learning was not explored much for crop yield and price forecasting, which presents a lucrative opportunity for exploration.

2.2.5 Transfer Learning

The effectiveness of transfer learning was explored by Anna Wang et al. in [104] for forecasting annual soybean yield in Argentina using TL from a model pretrained on forecasting annual soybean yield in Brazil. Their experiment showcased promising results, and the authors discussed the prospect of extending their application of TL to include other crops. The authors of [86] explore using TL in forecasting cross-building energy consumption. They test their model, named Hephaestus, on energy consumption times series of multiple schools, applying TL for schools with small datasets. Their results conclude that using TL features from the energy consumption of other schools enhances the energy consumption forecasting of a single school compared to only training using the individual school's data. Further, TL is applied for image classification in the medical field, with one study presented in [93] applying TL from large, publicly available pretrained models, such as CifarNet, AlexNet, and GoogLeNet, to classify medical images. The authors' implementation fine-tunes CNN layers and uses off-the-shelf pretrained feature extractors that are fed into fully connected layers for classification. They conclude that TL provides improvement in classification, which indicates potential for learning across datasets. Another study in [98] applied a similar TL methodology, concluding that, compared to a CNN trained without any TL, the use of a pretrained and fine-tuned CNN achieves better results and is more robust to the size of the training dataset. Moreover, the layer-wise fine-tuning of CNNs provides a potent method to achieve the best performance in their application, depending on how much data is available. The reviewed literature provides insight on applying TL among similar types of time series beside presenting methods that could help in enhancing forecasting capabilities even with limited data while decreasing the computational costs of model training by utilizing existing models. Although works in the applications of TL in image classification problems are abundant, not much literature exists for crop yield forecasting and existing work in the field such as [104] focuses on predicting annual yields. Thus, further research is needed for exploring the application of TL in crop yield forecasting which this work aims to contribute.

2.3 Evaluation Metrics

An evaluation metric is crucial for determining the effectiveness of any model. For time series, commonly used evaluation metrics as suggested in literature are Mean Absolute Error (MAE), Root Mean-Squared Error (RMSE), as well as the R-Squared Coefficient (R^2) [112, 77, 90, 75].

2.3.1 Mean Absolute Error (MAE)

The MAE is a basic measure of the error between pairs averaged across all observations. The absolute difference between the true and predicted values is first calculated, then the average is computed for all observations. Since MAE does not involve squaring the difference, it is more robust to outliers compared to RMSE [108]. Mathematically, it is defined as in Equation 2.17.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (2.17)$$

where n is the total number of observations, y_i is the true value of observation i , and \hat{y}_i is the predicted value at observation i [9].

2.3.2 Root Mean Squared Error (RMSE)

The RMSE is another widely used evaluation metric, it is a measure of the standard deviation of the error. The difference between the true and predicted observations is first calculated and then squared, which is then averaged across all observations and finalized by computing its square-root. The squaring of the difference allows the metric to be sensitive to large error, making it suitable when penalizing larger errors is required. Mathematically, it is defined as in Equation 2.18.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (2.18)$$

where n is the total number of observations, y_i is the true value of observation i , and \hat{y}_i is the predicted value at observation i [16].

2.3.3 Coefficient of Determination (R^2)

The R-Squared coefficient, also named the coefficient of determination, is another commonly used metric in regression. It describes the percentage of variance of the dependent variable that is expressed by the independent variable. In other words, it is a ratio that describes how much of the difference in one variable is explained by the difference in another variable, reflecting how well the model replicates true values. As a ratio, its range is typically between 0 and 1, however it can be negative if the predicted regression line of the

model is worse than the mean of the dataset [47]. The closer the value is to 1, the better the model is at predicting the target variable, with the optimal value 1 indicating that the model replicates true values identically, while a value of 0 indicates that the model predicts the mean. It is calculated as the sum of the difference between the true values and the predicted values squared, divided by the sum of the difference between the true values and the mean value squared, subtracted from 1 as in Equation 2.19. While it is very intuitive to gauge the effectiveness of a model, its value increases as more predictors, or explanatory variables, are added to the model, making it susceptible to flawed indication [22].

$$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2} \quad (2.19)$$

where y_i is the true value of observation i , \hat{y}_i is the predicted value at observation i , and \bar{y} is the mean value of the observations [87].

2.3.4 Aggregated Measure (AGM)

Determining the effectiveness of a model in forecasting through multiple metrics can be challenging; given the varied merits and drawbacks of each metric. That is why the authors of [63, 64] proposed a new evaluation metric that attempts to combine the information captured by the MAE, RMSE, and R^2 described in Equations 2.17, 2.18, and 2.19. This allows for more comprehensive gauging of the best performing models over multiple metrics. Both the MAE and RMSE measure prediction errors and share the same units. Moreover, both metrics are negatively oriented, meaning that lower metric values indicate a better performance. Therefore, the average of both metrics is taken to integrate information carried by both. For the R^2 score, it is a unitless metric that is positively-oriented, which means the higher the value, the better the performance. Because the R^2 value range is typically between 0 and 1, taking its complement using $(1 - R^2)$ provides a negatively-oriented score, which is then multiplied by the average of the MAE and RMSE to produce a measure better indicative of the performance of a model. Mathematically, the **AGM** is defined as in Equation 2.20.

$$AGM = \frac{RMSE + MAE}{2} \times (1 - R^2) \quad (2.20)$$

2.3.5 Average AGM (AAGM)

In addition to the AGM, the **AAGM** is used when assessing forecasting results that span multiple varied instances per model, and one value is needed per model to summarize its performance; in this work the instances are different forecasting years. It is the average of the AGM values of a model across all instances [63] and is defined as in Equation 2.21.

$$AAGM = \frac{\sum_{i=1}^n AGM}{n} \quad (2.21)$$

where n is total number of years observed.

2.4 Chapter Summary

This chapter starts by describing, in detail, the background behind the deployed DL models. Then, it explores the nature of univariate and multivariate classes of time series, along with the conventional methods used to model them. It proceeds to describe how satellite images are typically preprocessed before being fed to the ML models by applying dimensionality reduction to achieve tractability. This is followed by a detailed description of ML techniques: voting regression, incremental learning, and transfer learning. In this, the literature review of time series forecasting, incremental learning, and TL is presented in detail along with the observed limitations this work aims to tackle. Finally, the evaluation metrics used in the assessment of time series modelling are described, along with their advantages and disadvantages.

Chapter 3

Methodologies and Proposed Solutions

To tackle the challenging task of forecasting crop yield and price, various elements need to be dealt with. These elements include the deployed datasets and their preprocessing as well as the proposed models along with their evaluation metrics and hyperparameter tuning. The nature of the data utilized should be understood to better exploit it with the models. This includes what the data reflects, the range of the measurements, as well as any limitations that need to be taken into consideration before preprocessing. To optimize the performance of the models, the data must be preprocessed as necessary for precise feature extraction. For the scope of this work, preprocessing includes image masking, normalization, and dimensionality reduction. Various models are proposed and implemented for different experimentation purposes in this work. The complexity of the models varies from simple DL models to compound ensembles that enhance the forecasting performance, as described in Section 2.1. Choosing suitable metrics to determine the effectiveness of these models is very important, since it provides insight to how well the models perform and allows for comparative analysis among them. The metrics described in Section 2.3 are utilized with respect to the datasets used. It is important to showcase the method of choosing the models' hyperparameters through tuning; to better understand the models and improve reproducibility. This chapter aims to provide a detailed description of those elements.

3.1 Datasets

In this work, different datasets are used for both constructing and evaluating the models. A detailed description of each of these datasets is presented in this section.

3.1.1 Midwestern USA Dataset

For a fair comparison with literature implementation in [112], the same datasets are used; where similar crop type and locations are considered. Hence, the satellite images of the soil surface reflectance, temperature, and moisture of Midwestern counties in the US are downloaded then mapped to the crop yield of each county.

MODIS surface reflectance and temperature (Input)

The Moderate Resolution Imaging Spectroradiometer (MODIS) is a vital instrument that records various types of measurements, including surface reflectance and temperature, and is aboard Terra satellite that has been orbiting Earth since 1999 [72]. The surface spectral reflectance is a measure of the percentage of energy that a surface reflects at a specified wavelength; it is unitless since it is a ratio. It is helpful in determining the biochemical composition of a surface, especially soil, which provides insight about its agricultural capabilities [35]. The surface temperature is also measured by MODIS and recorded in Kelvin. In addition, MODIS collects and records measurements for both surface reflectance and temperature on daily basis. The data used for this work spans the years 2003 to 2015, for the same 11 states in the USA considered in [112]: Arkansas, Illinois, Indiana, Iowa, Kansas, Minnesota, Missouri, Nebraska, North Dakota, Ohio, and South Dakota. The surface reflectance data is measured across 7 spectral bands, while the surface temperature is recorded twice a day, once in daytime and once at night, producing 2 bands. Both the surface reflectance and temperature are downloaded from the MODIS Terra Surface Reflectance and Surface Temperature official website [18]. Moreover, MODIS/Terra+Aqua land cover images are provided, downloaded, and used in this work for masking during preprocessing. A sample of the satellite images recording the surface reflectance and temperature in 2014 of Douglas, Kansas is visualized in Figure 3.1, with the image dimensions and value ranges presented.

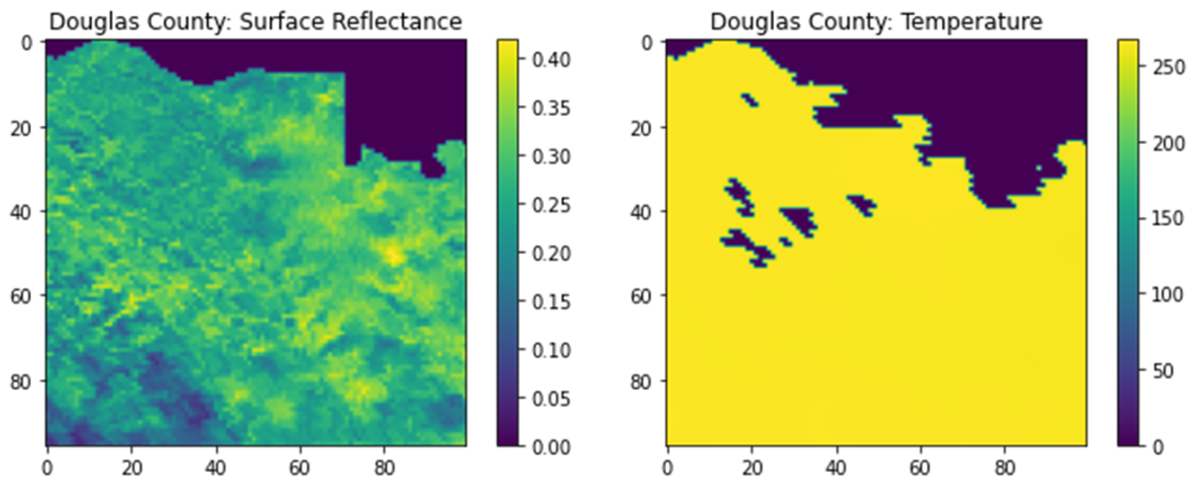


Figure 3.1: Sample satellite images of soil surface reflectance (left) and temperature (right) in Douglas, Kansas

USDA-FAS moisture levels (Input)

Satellite images of moisture data describing the levels of surface and subsurface moisture, spanning 2 bands, are deployed. They are measured in millimeters and collected every 3 days globally from both the Soil Moisture and Ocean Salinity (SMOS) as well as the Soil Moisture Active Passive (SMAP) satellites orbiting Earth since 2009 and 2015 respectively [2, 70]. Furthermore, the dataset is collected and maintained by the US Department of Agriculture - Foreign Agricultural Service (USDA-FAS) and obtained from [1]. The data used for this work spans the years 2010 to 2015 similar to the considered range in [77], for the same 11 states mentioned previously and considered in [112]. Figure 3.2 visualizes a sample of the satellite images measuring the moisture levels in 2014 of Douglas, Kansas; with the image dimensions and value ranges presented.

Annual soybean yield (Output)

Since the models are expected to forecast crop yield values, the ground truth target values used are the annual average soybean yields recorded for each county in bushels per acre. The data spans the years 2003 to 2015 and cover the 11 states described earlier. This data is publicly available and is obtained from the US Department of Agriculture (USDA) official website [69].

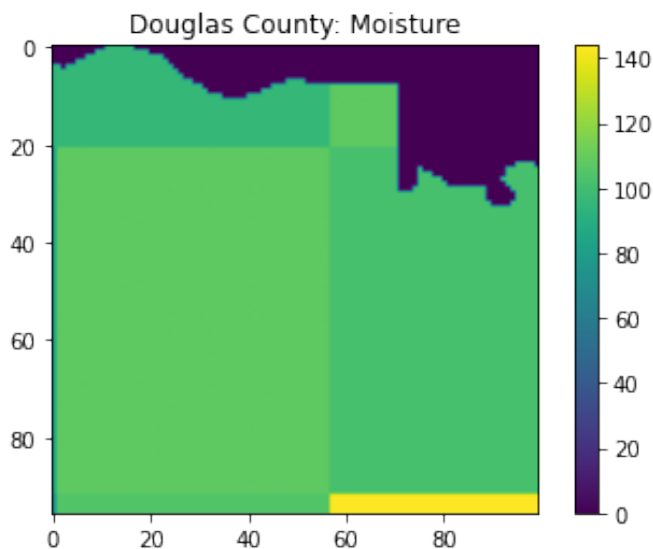


Figure 3.2: Sample satellite images of soil moisture in Douglas, Kansas

3.1.2 California Dataset

The state of California was chosen as a case study since it is the largest producer of strawberries in the USA due to its favorable climate conditions [57], it is also one of the largest producers of raspberries in the USA [4]. Three counties within California are considered: Santa Barbara, Ventura, and Monterey.

MODIS surface reflectance and temperature (Input)

Similar to the Midwestern datasets, the satellite images of surface reflectance and temperature are collected for Santa Barbara, Ventura, and Monterey counties in California [18]. The data spans the years 2011 to 2019 and the land cover is also downloaded per year for every county to be used in masking. Figure 3.3 visualizes a sample of the satellite images measuring the soil surface reflectance as well as temperature in Santa Barbara, California.

USDA-FAS moisture levels (Input)

Satellite images of moisture levels are also obtained from [1] for Santa Barbara, Ventura, and Monterey counties in California, spanning the years 2011 to 2019. Figure 3.4 visualizes

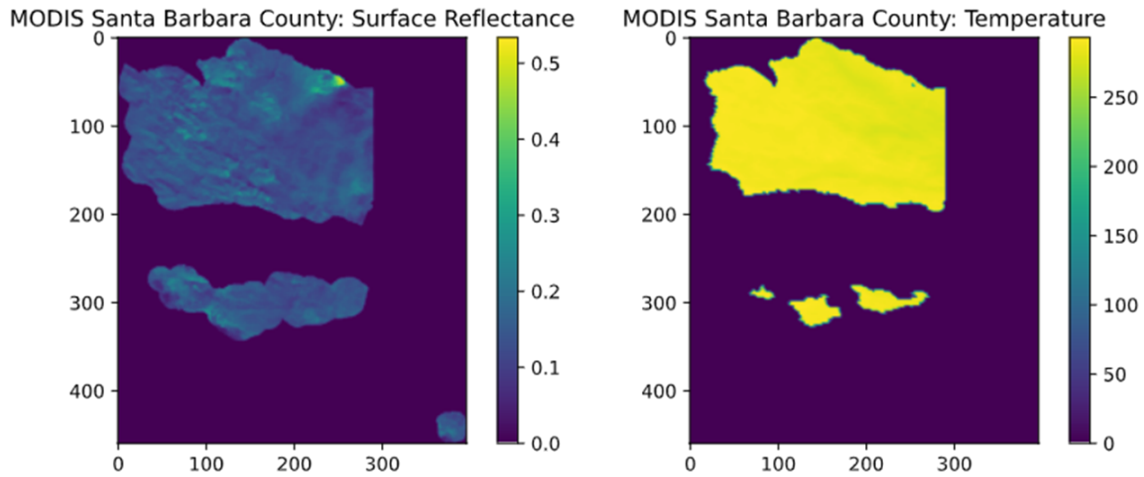


Figure 3.3: Sample satellite images of soil surface reflectance (left) and temperature (right) in Santa Barbara, California

a sample of the satellite images measuring soil moisture levels in Santa Barbara, California. Note that the moisture readings are limited to county’s inland which explains why the coastal areas and islands are not visualized like they are with the surface reflectance and temperature in Figure 3.3.

Daily FP yield and price (Output)

In order to forecast FP yields and prices, the daily strawberry and raspberry yields and prices are used as the ground truth target values. The yield is measured in pounds per acre, while the price is measured in US Dollars. For the scope of this work, the data collected spans the years 2011 to 2019 for Santa Barbara, Ventura, and Monterey counties in California. Moreover, the data for both strawberries and raspberries is publicly available and obtained from the California Strawberry Commission website [13]. Due to the availability of sufficient data for the three mentioned counties, they are specifically chosen for implementing the forecasting models. Figure 3.5 presents the strawberry yield time series in Santa Barbara, California for years 2011 to 2019.

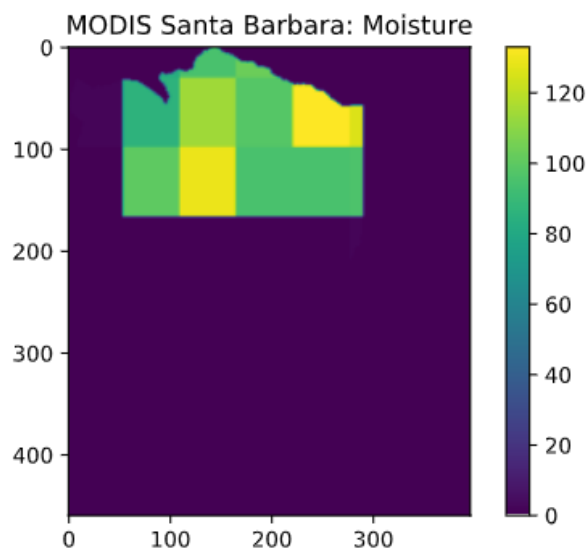


Figure 3.4: Sample satellite images of soil moisture in Santa Barbara, California

3.2 Data Preprocessing

To standardize the labelling of the satellite images for easy access, they are divided by the Federal Information Processing Standards (FIPS) code of their corresponding counties, with each image correlating to a specific county at a specific day. In addition, they require multiple levels of preprocessing before being fed into the models for testing. The images are first scaled and masked, then dimensionality reduction is applied before being concatenated into the final dataset form suitable for ML.

3.2.1 Preprocessing and Masking the Satellite Images

As suggested by the official datasets providers in [18] and [1], the images must be multiplied by a scaling factor as they, in their raw format, do not reflect their units described in Section 3.1.1. After the scaling, the images are passed through a land cover for masking, which is an image similar in dimensions to the other images, but whose pixel values represent the type of land corresponding to that pixel. Figure 3.6 visualizes the land cover of Santa Barbara, California in 2011, with the legend showing different classified types of lands described in [19], specifically label 12 denoting croplands.

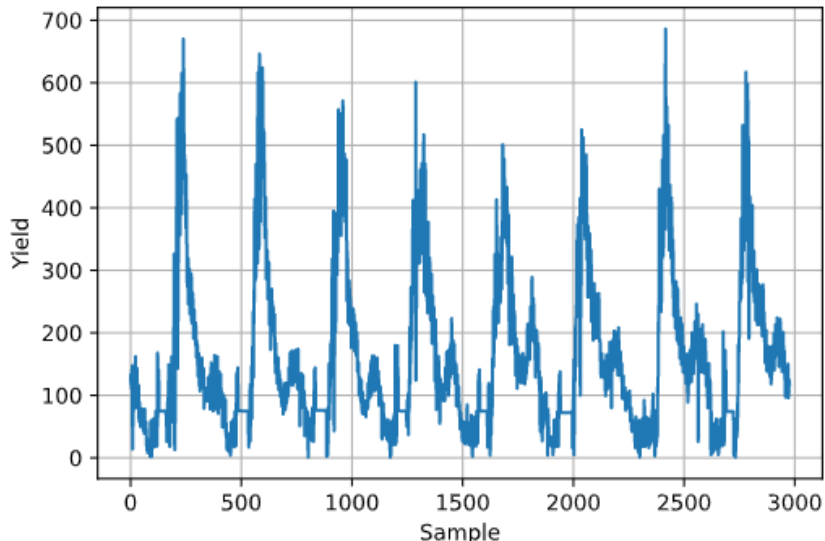


Figure 3.5: Strawberry yield time series in Santa Barbara, California

An image mask is used to selectively maintain desired pixels only, which helps filter out unnecessary data and reduce computational costs. For this work, the models should only be provided with cropland data, thus the mask is applied by setting the land cover pixel values that correspond to cropland locations to a value of 1, while all other pixels that denote non-cropland locations are set to a value of 0. The mask is then applied to the images by multiplying them element by element, maintaining wanted pixel values as they are multiplied by 1 and masking unwanted pixel values as they are multiplied by 0. This ensures that the models train exclusively on pixel values that correspond to croplands and reduces the number of pixels processed by disregarding irrelevant pixels. If pixels that hold desired cropland locations are denoted by p , then the mask is applied as in Equation 3.1.

$$p_{i,j} = \begin{cases} p & \text{for } i, j \in m \\ 0 & \text{for } i, j \notin m \end{cases} \quad (3.1)$$

where $p_{i,j}$ is the value of the image pixel at coordinates (i, j) and m symbolizes the set of pixel coordinates labelled as cropland.

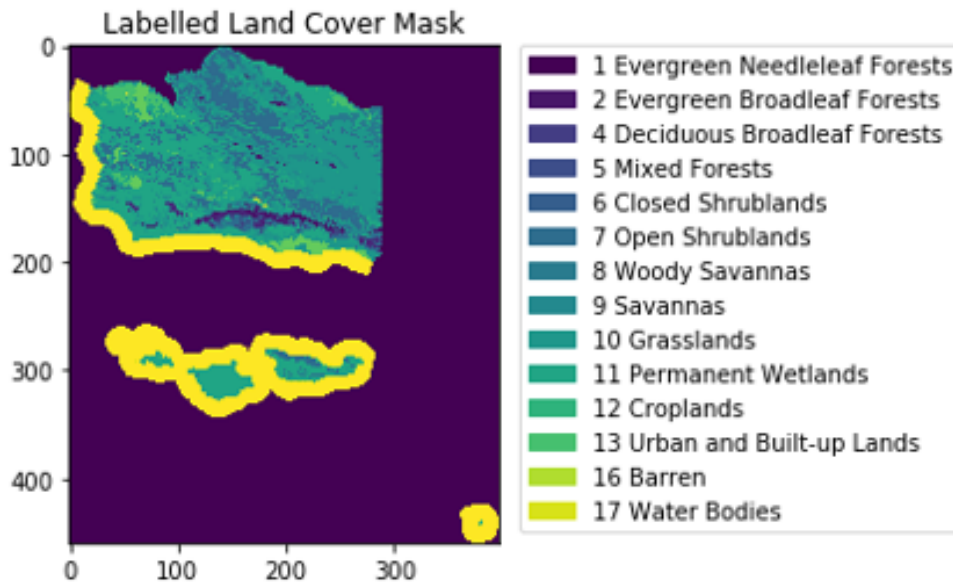


Figure 3.6: Sample of land cover used for masking of Santa Barbara, California

3.2.2 Conversion to Histograms

As described in Section 2.2.3, dimensionality reduction must be applied to the satellite images due to the spatial scarcity and to achieve tractability. This is done through the conversion of the satellite images into 1-Dimensional frequency count histograms of the values of their pixels under the assumption of permutation invariance. The conversion to histograms is defined in Equation 3.2, for all $0 \leq k < K$ where K is the highest possible pixel value.

$$h(k) = \text{card}\{(i, j) \mid p_{ij} = k\} \quad (3.2)$$

where k represents the value, or range of values, of the pixels for each frequency histogram h while card is the cardinality, or set size, of pixels for a specified k [12]. As implemented in [112] and suggested to be a reasonable spread of pixel value bins, each histogram is divided into 32 bins that hold different pixel values ranges.

Afterwards, the histograms are scaled to produce density histograms that have values bounded between 0 and 1 to standardize the value ranges hence optimize the feature extraction. Since the Midwest datasets are of annual yield data, each data point represents a specific county at a specific year. This means that the input must contain histograms for forecasting a year's worth of satellite images. For a fairer comparison with the experiment

conducted in [112], 30 images are used for each year per record. Moreover, the experiment uses 7 bands of surface reflectance, 2 bands of land surface temperature, and 2 bands of moisture levels; thus, an additional dimension is required to accommodate all 11 bands. The structure is visualized in Figure 3.7 for easier understanding, with the histograms stacked to produce the dataset used as input for the models. Dimension b denotes the number of bins per histogram, dimension h denotes the histograms stacked per sample, and dimension d denotes the number of bands used per sample.

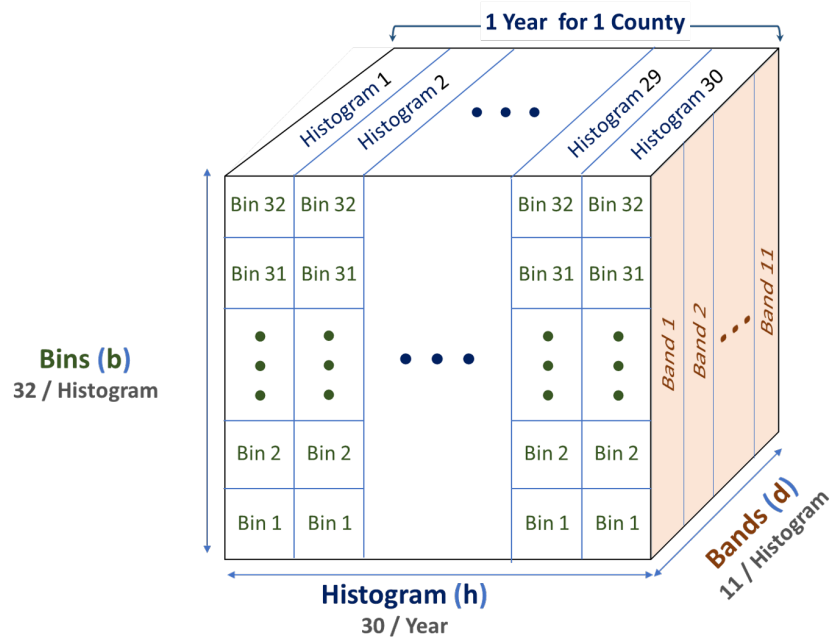


Figure 3.7: 3D histogram collection per sample for a given year at a given county in the Midwest

For the experiments in California, the process is slightly modified since the forecasting is done on a daily basis. Each sample contains images of the last 140 days while forecasting for a given day, which is selected based on the findings in [73]. Moreover, most of the experiments only use the moisture and land surface temperature bands for forecasting, totaling 4 bands. The structure is visualized in Figure 3.8 for easier understanding, with the histograms stacked to produce the 3D histogram collection per sample used as input for the models.

The image preprocessing is implemented using the Geospatial Data Abstraction Library

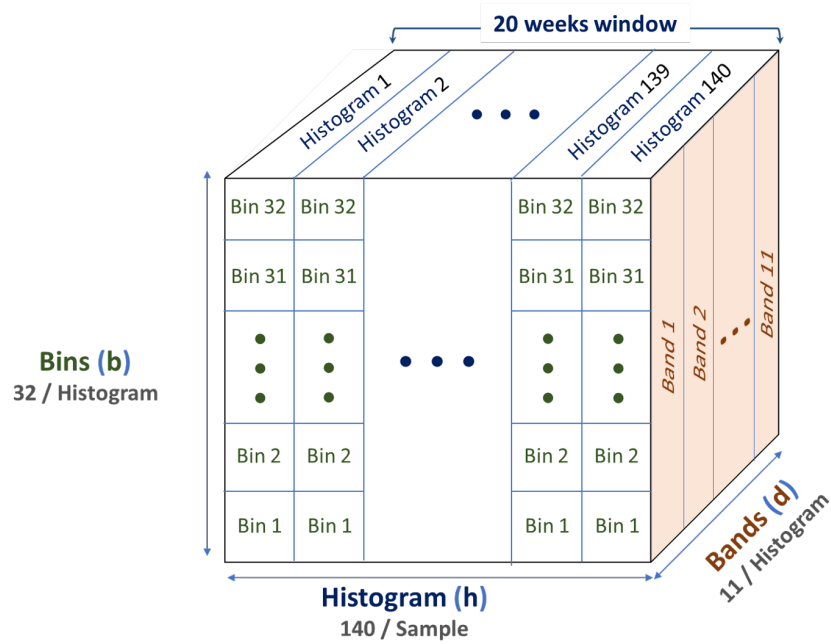


Figure 3.8: 3D histogram collection per sample for daily forecasting in California

(GDAL) in Python to import the images into a readable format that can be processed [106]. In addition, the tested models are implemented using Python’s Tensorflow Keras library [17], which was chosen for its user-friendliness and scalability.

3.2.3 Forming the Final Data Structure

After the 3D histograms are formed, they are appended to the other inputs as well as the mapped output corresponding to them. For the Midwest, the tuple of images year as well as county central longitude and latitude (year, longitude, latitude) is then appended to each input histogram along with the mapped output yield corresponding to that same date and location. The year and coordinates are only used by the GP component to further enhance the models’ forecasting performance. The histograms that are considered are those that have available yields in the USDA website [69], Table 3.1 shows the number of these records for each year as well as the total.

The entire process is visually summarized in Figure 3.9, with the 3D histograms concatenated to form the entire dataset used, along with the year, coordinates, and the output

Table 3.1: Number of counties in the Midwest with available yields from 2003 to 2015

<i>Year</i>	<i>2003</i>	<i>2004</i>	<i>2005</i>	<i>2006</i>	<i>2007</i>	<i>2008</i>	<i>2009</i>
Counties	808	818	807	800	797	741	749
<i>Year</i>	<i>2010</i>	<i>2011</i>	<i>2012</i>	<i>2013</i>	<i>2014</i>	<i>2015</i>	<i>Total</i>
Counties	801	770	754	712	726	667	9950

yield.

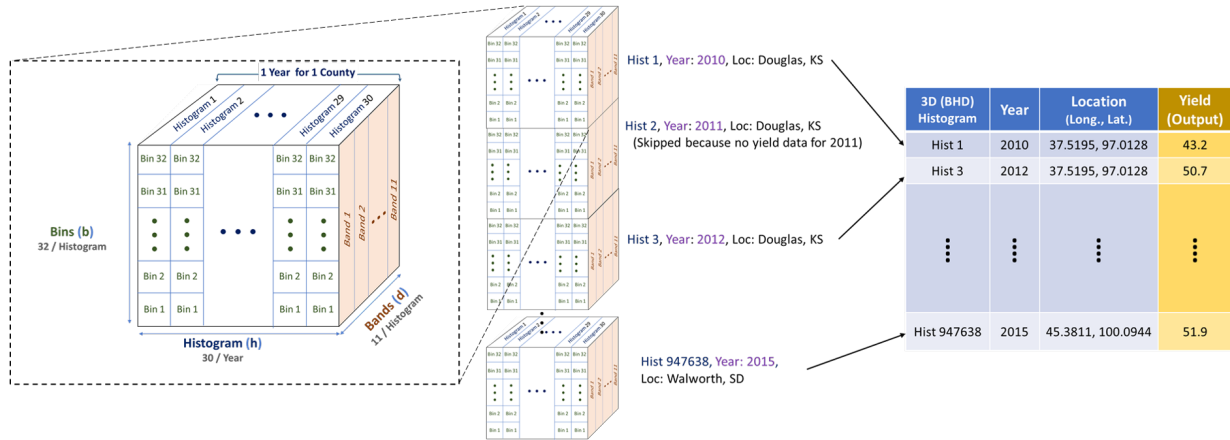


Figure 3.9: Mapping 3D county histogram collection per year (Input) to the corresponding county yield of same year (Output)

For California, since only a single state is considered, the coordinates would not carry much information. Moreover, since the forecasting is on daily basis, the years do not carry much information either. Thus, the GP is not utilized for this dataset since the years and county coordinates are dropped out of the dataset. This process is better visualized in Figure 3.10, with each 3D histogram appended with its corresponding yield or price value as output.

The produced 3D histograms are directly fed into the CNN model since the convolution is across 2 dimensions, but for the other models such as LSTM, CNN-LSTM, SAE, and VAE the structure needs to be further reduced into 2D by flattening dimensions h and d into a single dimension as presented in Equation 3.3, which are then fed into the aforementioned

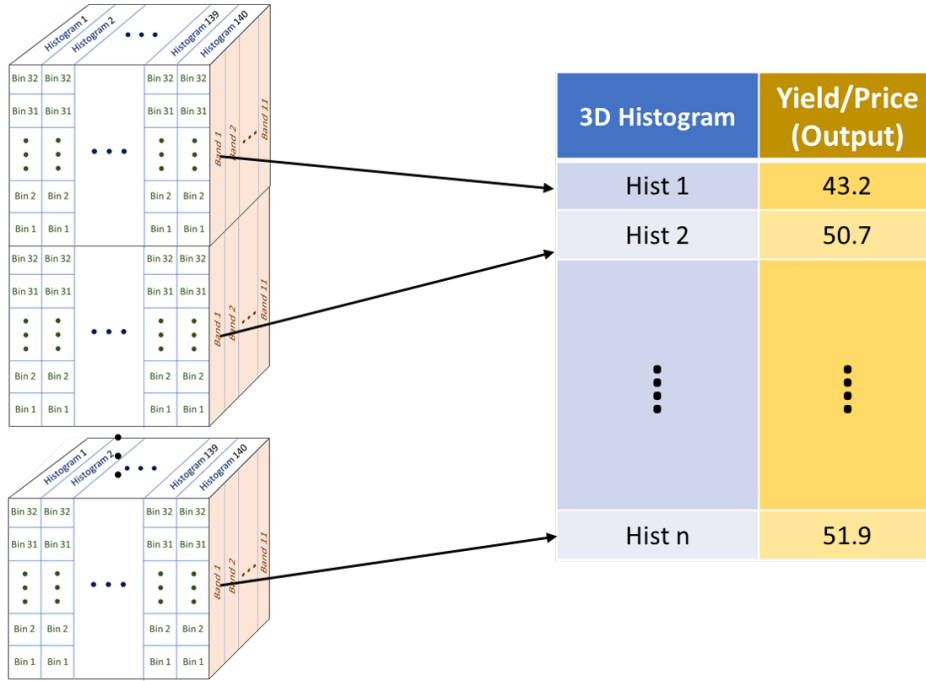


Figure 3.10: Mapping 3D histogram collection (Input) to the corresponding daily yield or price (Output) in California

models.

$$\begin{bmatrix} H_{1,1} & H_{1,2} & \dots & H_{1,d} \\ H_{2,1} & H_{2,2} & \dots & H_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ H_{h,1} & H_{h,2} & \dots & H_{h,d} \end{bmatrix} \rightarrow [H_1 \quad \dots \quad H_{h*d}] \quad (3.3)$$

where each $H_{h,d}$ represents a histogram with b bins and d bands.

3.3 The Proposed Models

The background of the deployed models is given in Section 2.1. These models are adjusted to fit the various forecasting tasks in this work. In this section, details of the deployed models along with the newly proposed ensembles are provided.

3.3.1 LSTM

For Midwestern counties, the utilized LSTM model used is similar to the one described in [112]. As illustrated in Figure 3.11, the model is made up of an LSTM layer comprised of 128 units that is fed to a fully connected dense layer of 256 nodes and a final single node layer for regression.

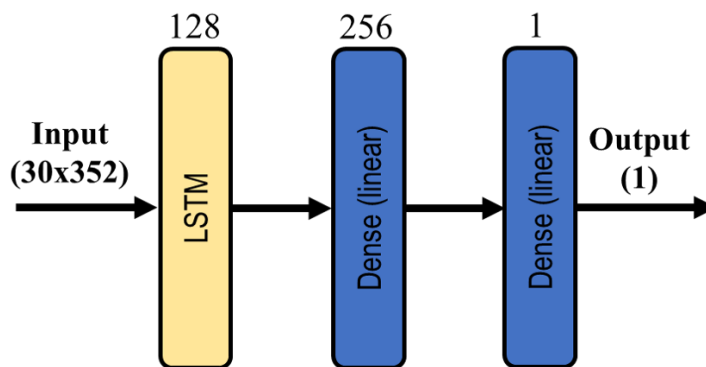


Figure 3.11: LSTM architecture deployed in [112]

For California, the output of the LSTM layer containing 128 units is fed into a fully connected dense layer of 128 nodes, followed by the single node output layer for regression as shown in Figure 3.12.

3.3.2 GP

For the Midwestern counties, the GP is used with the tested models to further enhance their prediction capabilities. It is fed the yields output of the deployed models along with their corresponding years, latitude-longitude coordinates, as well as the feature vector of the DL model. Based on those, the GP models the residuals of the forecasting model to produce a more accurate prediction based on the spatiotemporal dependencies carried

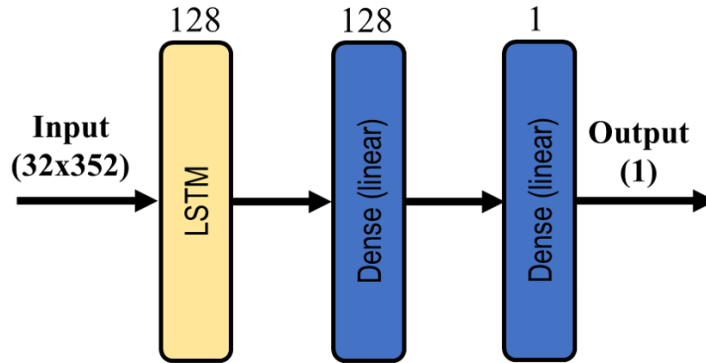


Figure 3.12: LSTM architecture for the California experiment

by the years and coordinates. Because the Midwest dataset is the only one with varying spatiotemporal dependencies, the GP implementation is limited to it. In terms of notation, the term “+GP” is appended to a model to indicate that the model feeds into a GP; for example an LSTM with a GP is denoted by LSTM+GP.

3.3.3 CNN-LSTM

The CNN-LSTM models try to incorporate the capabilities of extracting spatial features of CNNs and temporal features of LSTMs. It should be noted that the deployed models vary slightly in architecture given the dataset and required output variable. For the Midwestern counties, the proposed CNN-LSTM model is as illustrated in Figure 3.13. The data batch is normalized, fed through 2 convolutional layers with a dropout layer of 0.2 for regularization, the output of which is passed through another normalization layer then into an LSTM layer with 128 units. Finally, the output is fed into 3 dense layers of sizes 512, 1024, and 1 respectively for regression. This architecture is chosen through phases of trial and error.

For the California dataset yield and price forecasting, the model architecture is as described in Figure 3.14. The histograms are fed into a normalization layer, followed by 3 convolutional layers and a max pooling layer which feeds into an LSTM layer of 64 unit, succeeded by 3 fully connected layers of sizes 128, 256, and 1 respectively for regression. It should be noted that the max pooling layer is omitted for price forecasting, as that leads to improved forecasting demonstrated through experimentation.

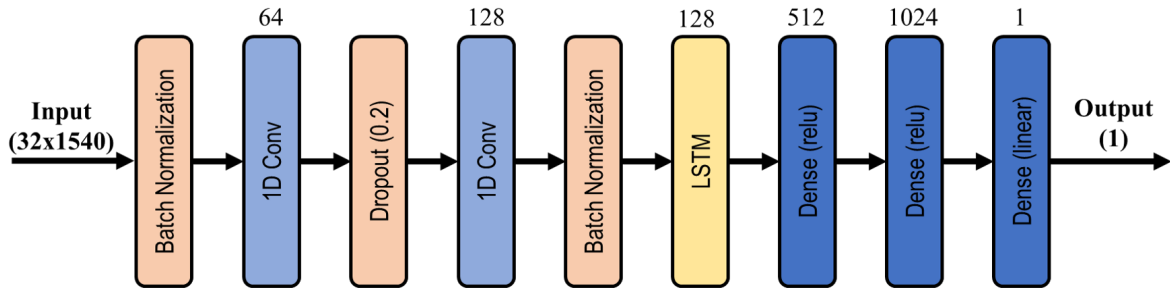


Figure 3.13: CNN-LSTM architecture for yield forecasting in the Midwest

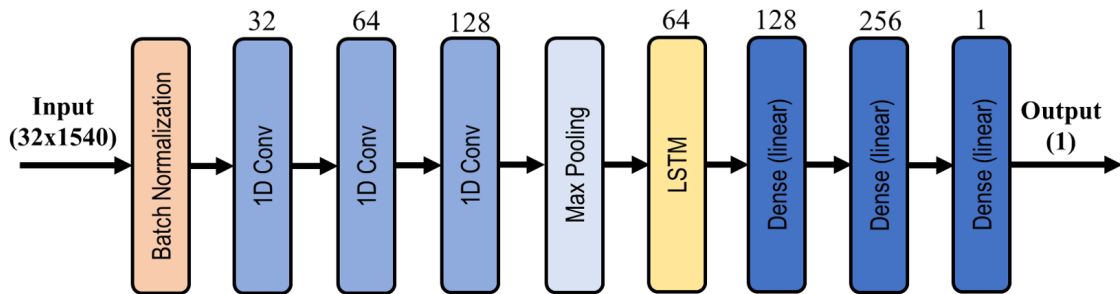


Figure 3.14: CNN-LSTM architecture for yield and price forecasting in California

3.3.4 CNN-LSTM Ensemble (CNN-LSTM_Ens)

The proposed ensemble is of the CNN-LSTM models described in Section 3.3.3, with both weaker models trained on the same data. The voting regressor is used to reduce the variance of weaker models as well as improve their prediction by combining the different trends extracted by the models through averaging. The block diagram of the proposed ensemble is illustrated in Figure 3.15.

3.3.5 SAE

Autoencoders are employed for their ability to extract more global and discriminative features due to the reduced dimensionality in their hidden layers. The proposed SAE is a stacked LSTM autoencoder, which takes advantage of the deep LSTM capabilities for both encoding and decoding. The model encodes the input data into a latent vector which forces it to further optimize the vector representation of the data features. This helps the model better extract trends in the data which improves its learning to forecast the target

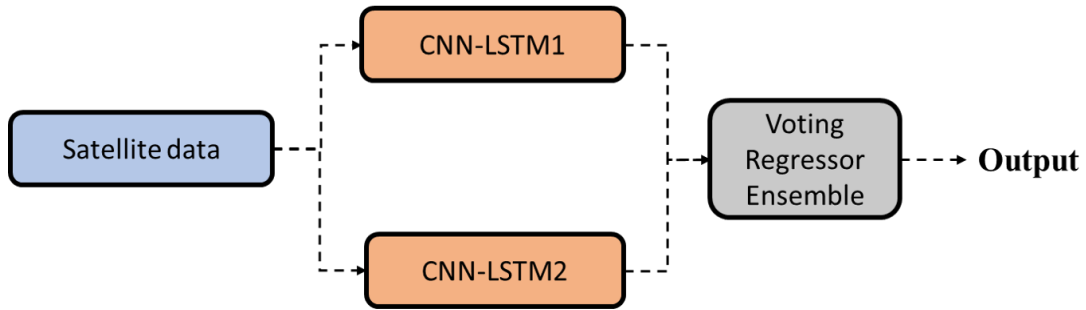


Figure 3.15: Block diagram of CNN-LSTM_Ens

variable. Due to the time-dependent nature of LSTMs, the encoder state has to be passed to the decoder to be used as its initial state. The encoder is usually detached from the model after training and its learned latent vector is used as a feature vector for regression. However, in this proposed version of the model, the step of detaching the encoder is dismissed and the decoder is directly fed into a dense layer for regression. Figure 3.16 visualizes the architecture of the proposed model, where the LSTM encoders and decoders are fed into the fully connected layer for regression.

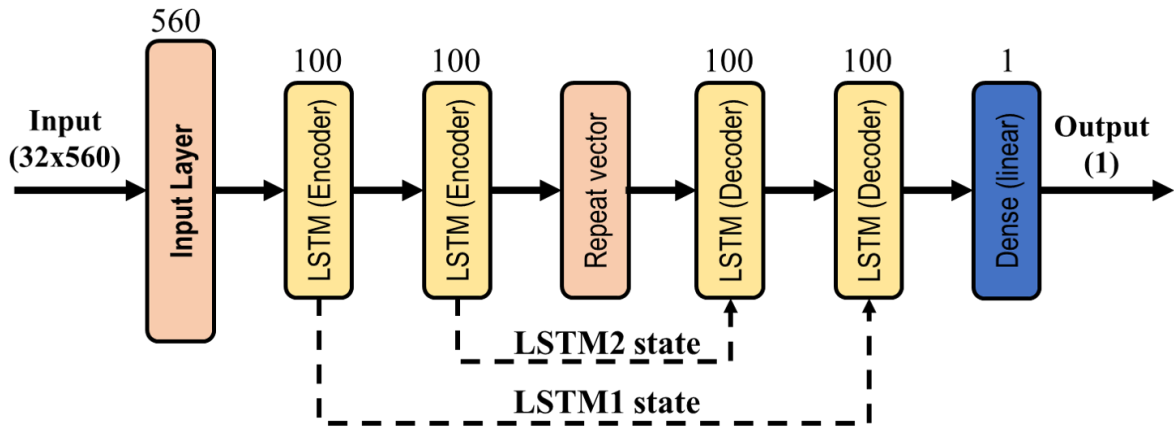


Figure 3.16: SAE architecture for yield & price forecasting using satellite images

3.3.6 VAE

The proposed VAE model takes advantage of the deep convolutional layers for both encoding and decoding the input data. The data is encoded into a latent vector that is sampled

using a learned mean and variance distribution. After training, the encoder is detached from the model and the latent vector is used as a feature vector for regression. Figure 3.17 visualizes the proposed model architecture, with the convolutional encoder and decoder layers learn the latent vector that is then used for regression.

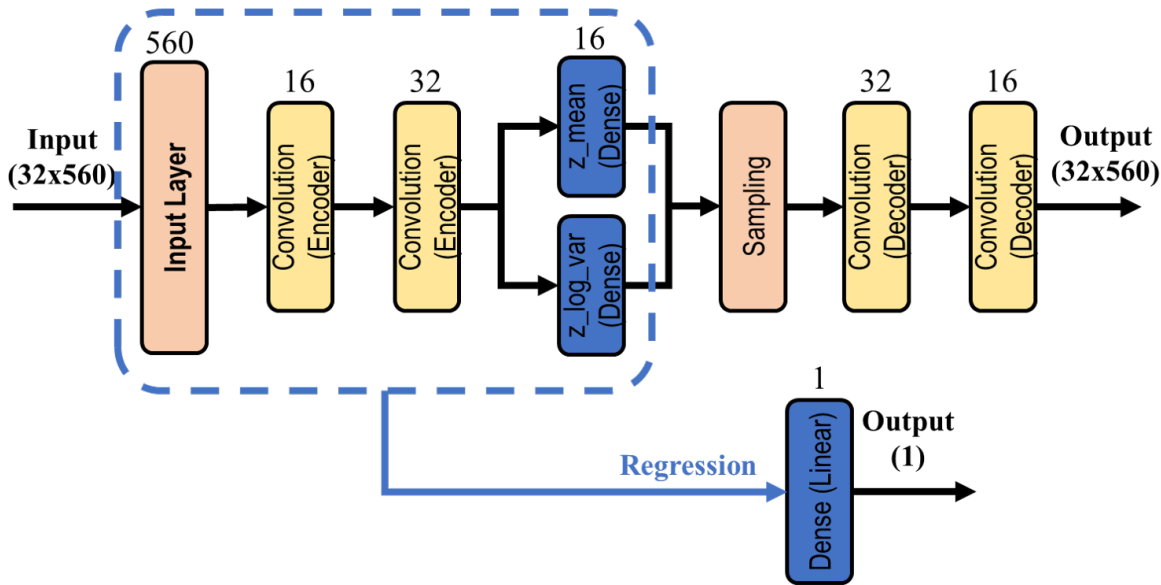


Figure 3.17: VAE architecture for yield & price forecasting using satellite images

3.3.7 SAE-CNN-LSTM Ensemble (SAE-CNN-LSTM_Ens)

To further improve the overall performance, the outputs of both models described in Sections 3.3.5 and 3.3.6 are combined. The proposed voting regressor incorporates the outputs of a CNN-LSTM and SAE, trained with the same data, by averaging them. The block diagram of the voting regressor ensemble is illustrated in Figure 3.18.

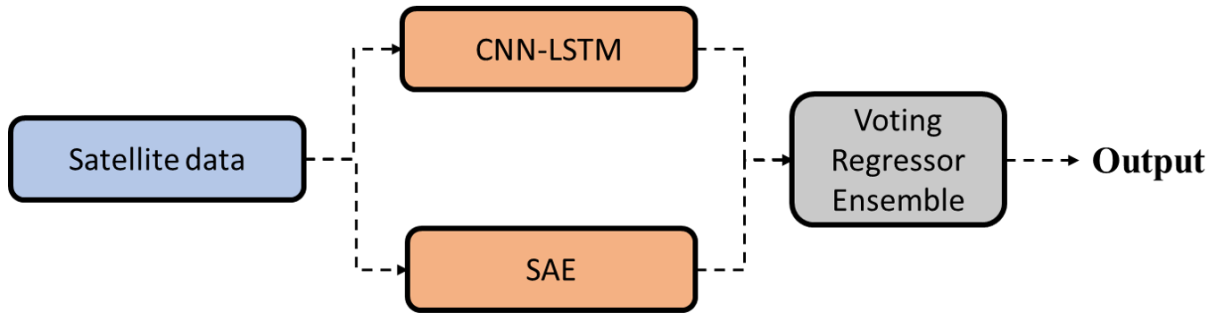


Figure 3.18: Block diagram of the proposed CNN-LSTM-SAE voting regressor model

3.4 Evaluation Metrics Units

The evaluation metrics described in Section 2.3 are deployed while varying their unit depending on the required output. The MAE and RMSE scores are measured in bushels per acre for the Midwest yield dataset, pounds per acre for the California yield dataset, and US dollars for the California price dataset. As for the R^2 , it is unitless since it describes a ratio, while the AGM and AAGM are measured in bushels per acre for the Midwest yield dataset, pounds per acre for the California yield dataset, and US dollars for the California price dataset as it depends on the units used in the obtained datasets.

3.5 Hyperparameter Tuning

A major challenge in using DL models is choosing the hyperparameters to best suit the target task and data. One of the most crucial hyperparameters is the learning rate, this is the step size that determines the increment of change the algorithm should use to update the weights as it approaches a local or a global minimum. The challenge with tuning this parameter is that if it is too large then the model is likely to get stuck in a local minimum and if it is too small then the model takes much longer to converge to a minimum [109]. Thus, ensuring the learning rate is well tuned is crucial to keep the model generalized and avoid underfitting or overfitting to the data. The process of determining the hyperparameters such as the learning rate, optimizer algorithm, model dimensions, and batch size depended heavily on manual selection based on experimentation results.

Another key technique is the use of regularization, which comes in many forms including using dropout and normalization layers as described in Section 3.3. Dropout layers

randomly set weights in the input layer to zero, forcing the model to adapt to a scarcer representation of the data thus generalizing it. Batch normalization layers scale the weights of its input layer which stabilizes and accelerates the training process of DL networks. Analysis conducted in [55] indicates that using batch normalization in DL networks shares the same traits of regularization.

3.6 Chapter Summary

The details of the deployed datasets are highlighted at the start of this chapter including the time period and geographic area they cover along with their sizes. These datasets include:

- MODIS surface reflectance and land surface temperature datasets for both Midwestern counties and California.
- USDA-FAS surface and subsurface moisture levels datasets for both Midwestern counties and California.
- USDA annual soybean yield averages for Midwestern counties.
- California Strawberry Commission daily yields and prices for Santa Barbara, Ventura, and Monterey counties in California

Afterwards, the data preprocessing procedure conducted on the satellite images is explained and the resulting dataset structure is illustrated. Next, the proposed forecasting DL models are described along with their architecture, dimensions and layer sequences. Finally, the units of the evaluation metrics are presented and a description of the different hyperparameters tuning and regularization approaches is provided.

Chapter 4

Experiments and Results Analysis

The preceding chapter covered the proposed solution for prediction and forecasting yield and price along with the datasets, models, and evaluation metrics. This chapter presents the experiments conducted based on the proposed solution, outlining their methodology, results, and subsequent analysis. The experiments conducted are presented in this chapter as follows:

The first set of experiments explores the prediction of annual soybean yields in the USA Midwest using the proposed models compared to literature benchmark in [112] and the effect of adding surface and subsurface moisture bands. The second set of experiments explores the models' performance in Santa Barbara, California for more frequent forecasting of daily strawberry yields and prices with varying the forecast window. To further enhance the performance and generalization, new models are proposed in the third and fourth sets of experiments for yield and price forecasting for a larger number of diverse counties such as Ventura and Monterey. The fifth experiment set explores the use of incremental learning on CNN-LSTM Ensemble for yield forecasting to adapt to more frequent updating provided new data. To efficiently generalize the application of proposed models to other FPs with minimal retraining, in the last set of experiments, the application of TL is explored for forecasting raspberry yields using TL from the built strawberry models.

4.1 Annual Prediction in Multiple Counties

This set of experiments focuses on further improving the prediction performance of yearly average yields of soybean in multiple counties in the US Midwest using the proposed

models. Moreover, it investigates the effect of adding surface and subsurface moisture levels as input parameters on the prediction capabilities of the proposed models.

4.1.1 Predicting Yield with Surface Reflectance and Temperature using data from 2003 to 2015

The aim of this experiment is to explore the performance of the proposed models compared to the literature. To achieve that, the proposed CNN-LSTM+GP model is tested to predict yearly soybean yields in Midwestern counties, against the LSTM, CNN, LSTM+GP, and CNN+GP literature models proposed in [112]. The CNN model proposed in [112] is made up of two pairs of 2D convolutional layers of size 128 and 256 respectively, followed by three 2D convolutional layers of size 512 which are fed into a fully connected layer of 2048 nodes. The models are trained on the histograms ranging from year 2003 to 2010 to predict the average soybean yields across 5 years, 2011 to 2015; the AGM per year and AAGM are used as evaluation metrics. The results of the experiment are presented in Table 4.1, they indicate that the CNN-LSTM+GP outperforms the other tested models with the least yearly AGM as well as the overall AAGM. Moreover, the models utilizing a GP outperform their counterparts without GP.

Table 4.1: AGM and AAGM results for models tested on years 2003 to 2015

<i>Model</i>	<i>AGM</i>					<i>AAGM</i>
	<i>2011</i>	<i>2012</i>	<i>2013</i>	<i>2014</i>	<i>2015</i>	<i>2011-2015</i>
LSTM	1.51	1.90	2.59	3.51	2.94	2.49
LSTM+GP	1.60	1.85	2.13	2.27	1.93	1.96
CNN	1.27	1.21	2.88	1.14	2.02	1.70
CNN+GP	1.14	0.93	2.50	0.95	2.30	1.56
CNN-LSTM+GP	0.48	0.60	0.65	0.57	1.70	0.80

The fact that the CNN-LSTM+GP outperforms its simpler counterparts can be accredited to its ability to capture spatiotemporal feature dependencies by combining the properties of both CNN and LSTM. Furthermore, the addition of the years and coordinates data allows the GP to extract even more features, enhancing the models' performance.

4.1.2 Predicting Yield with Surface Reflectance, Temperature, and Moisture using data from 2010 to 2015

The second experiment in this set explores the effect of considering the moisture input parameter on the prediction performance. Since the moisture dataset starts from 2010, the dataset range considered for this experiment is set for years 2010 to 2015, with the models being trained on years 2010 and 2011 then tested using years 2012 to 2015. The models are trained with surface reflectance and temperature bands with and without the moisture bands for a more controlled and fair experiment. The results of testing the models without the moisture bands are presented in Table 4.2, which further confirm the results found in the first experiment, Section 4.1.1.

Table 4.2: Models’ performance based on the AGM and AAGM for years 2010 to 2015 without considering the moisture parameter

<i>Model</i>	<i>AGM</i>				<i>AAGM</i>
	<i>2012</i>	<i>2013</i>	<i>2014</i>	<i>2015</i>	<i>2012-2015</i>
LSTM	2.64	1.39	2.03	1.59	1.91
LSTM+GP	2.63	1.39	1.90	1.24	1.79
CNN	1.50	1.65	0.97	1.69	1.45
CNN+GP	1.37	1.73	0.91	1.54	1.39
CNN-LSTM	0.69	0.93	0.73	1.04	0.85
CNN-LSTM+GP	0.64	1.06	0.74	1.03	0.87

In addition, the same models are trained and tested with the moisture bands, with the results showcased in Table 4.3. When compared to the results without using moisture bands in Table 4.2, it is clear that the results with the moisture bands have a much lower AGM per year and AAGM.

The results are better summarized in Figure 4.1, which further highlights the improvement provided by using the moisture bands. In addition, it shows that the CNN-LSTM architecture consistently outperforms the other tested models, followed by CNN and finally the LSTM model. Additionally, the GP almost always leads to an improvement in prediction, except with for the CNN-LSTM in 2015 without moisture bands.

Finally, the percentage AGM improvement between the proposed CNN-LSTM+GP model using moisture bands and the worst performing model, which is the LSTM model

Table 4.3: Models’ performance based on the AGM and AAGM for years 2010 to 2015 after considering the moisture parameter

<i>Model</i>	<i>AGM</i>				<i>AAGM</i>
	<i>2012</i>	<i>2013</i>	<i>2014</i>	<i>2015</i>	<i>2012-2015</i>
LSTM	2.32	1.57	1.50	1.93	1.83
LSTM+GP	2.40	1.38	1.37	1.34	1.62
CNN	1.42	1.33	1.27	1.24	1.32
CNN+GP	1.28	1.40	1.09	1.31	1.27
CNN-LSTM	0.75	0.83	0.63	1.00	0.80
CNN-LSTM+GP	0.35	0.51	0.37	1.02	0.56

without using moisture bands, is calculated and presented in Table 4.4. The results show that the CNN-LSTM+GP consistently outperforms the simple LSTM model by an average of 71%.

Table 4.4: AGM and AAGM percentage improvement compared to worst performing model

<i>Model</i>	<i>AGM</i>				<i>AAGM</i>
	<i>2012</i>	<i>2013</i>	<i>2014</i>	<i>2015</i>	<i>2012-2015</i>
Simple LSTM Model	0.35	0.51	0.37	1.02	0.56
CNN-LSTM+GP Model	2.64	1.39	2.03	1.59	1.91
AGM Improvement	87%	63%	82%	36%	71%

Comparing with the results obtained in [112], Table 4.5 presents the percentage RMSE improvement from year 2012 to 2015 between the results of the proposed CNN-LSTM+GP model with moisture and the authors’ original models, using the same input data, apart from moisture, and predicting for the same years. It is evident that the proposed model outperforms the literature model in each of the predicted years, with an overall average RMSE percentage improvement of 31%.

The improvement can be attributed to the inclusion of moisture data as well as the use of CNN-LSTM compared to the CNN, which utilizes the feature extraction abilities

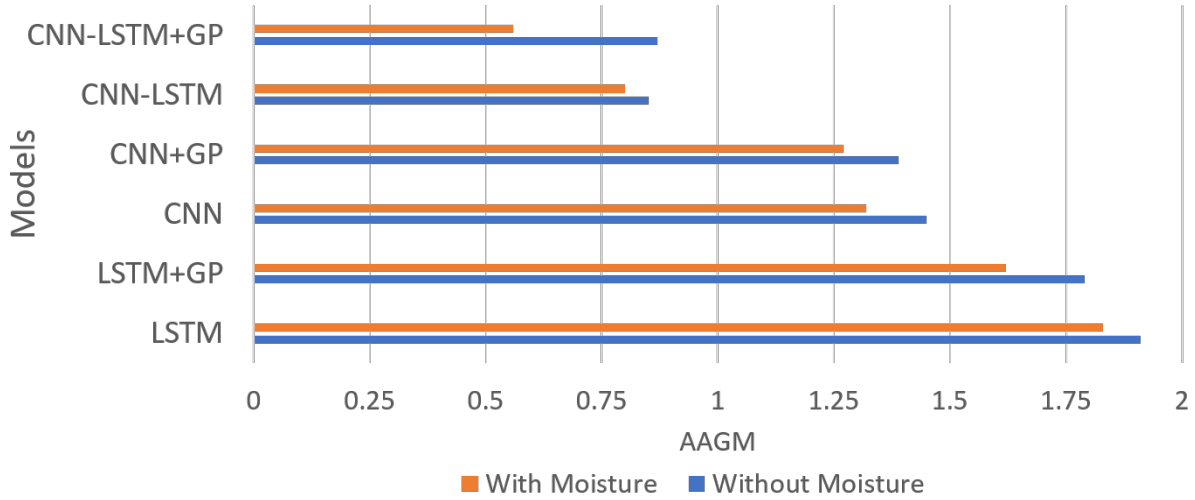


Figure 4.1: AAGM of tested models with & without moisture

of CNN along with the time-dependent information retention of LSTM. Additionally, the proposed CNN-LSTM model performs better than the CNN described in [112] with a much smaller number of nodes, layers and parameters, thus the proposed model seems to capture features in a more effective manner.

Table 4.5: RMSE improvement compared to literature model

<i>Model</i>	<i>RMSE</i>				<i>Average RMSE</i>
	<i>2012</i>	<i>2013</i>	<i>2014</i>	<i>2015</i>	<i>2012-2015</i>
Literature model in [112]	5.68	5.83	4.89	5.67	5.52
CNN-LSTM+GP Model	3.60	3.66	3.28	4.67	3.80
RMSE Improvement	37%	37%	33%	18%	31%

4.2 Daily Forecasting in a Single County

For this experiment, the surface reflectance, temperature, and moisture satellite images are used to train the models to forecast strawberry yield in Santa Barbara. The previous experiment in Section 4.1 demonstrated how the proposed CNN-LSTM model outperforms the simpler models, thus this experiment aims to test if that conclusion holds for the more relevant case study: forecasting daily FP yields in California. More specifically, the experiment focuses on investigating the effect of extending the forecasting window further into the future after which the models, LSTM, CNN-LSTM, and CNN-LSTM_Ens, are tested for a fixed number of days ahead, 35 days ahead. As previously mentioned in Section 3.3.2, the GP is not used in the remaining experiments since it relies on spatiotemporal variance in the data in terms of annual change and geographical location and the experiments in California are fixed to a single state.

4.2.1 Exploring Daily Strawberry Yield Forecasting Windows

This experiment explores how far into the future the models can forecast, which requires modification of the data structure. During the preprocessing of the satellite images described in Section 3.2, the ground truth output values corresponding to each sample are varied in the number of days ahead. For this experiment, for any given day x , the previous $x - 140$ until x days are used to forecast y days into the future, with y values of 0, 7, 14, 21, 28, and 35 days ahead. The AGM results of the tested LSTM, CNN-LSTM, and CNN-LSTM_Ens models forecasting varying days ahead are presented in Figure 4.2, which indicates that the AGM increases the further the forecasting is in the future. This trend is as expected since longer forecasting horizons require newer data that is closer to them in time. Additionally, the CNN-LSTM Ensemble is consistently the best performing when forecasting across all tested days ahead; it has the lowest AGM compared to the LSTM and CNN-LSTM models.

Taking the furthest forecasting window of 35 days ahead, Figure 4.3 visualizes the forecasted yield values by the LSTM, CNN-LSTM, and CNN-LSTM_Ens models versus the true yield values for years 2018 and 2019, where SIM denotes satellite images. It is visually clear that the CNN-LSTM Ensemble is the closest to forecasting the true yield values, even though it is not able to forecast the steep trend peaks and the sharp fluctuations. This is most likely due to the inconsistent fluctuation of the trend peaks evident in Figure 3.5, which means that the models would not be able to capture peaks unique to the tested years that do not correlate to the learned trends.

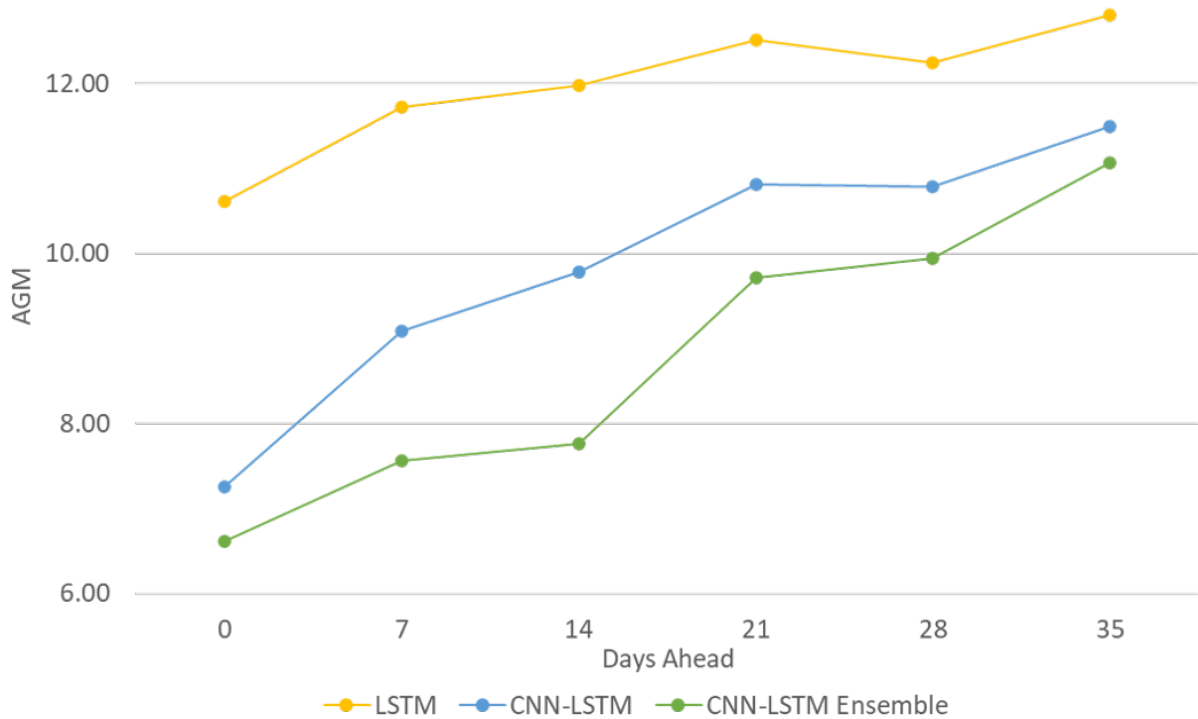


Figure 4.2: Performance based on AGM of LSTM, CNN-LSTM, and CNN-LSTM_Ens models for forecasting yield in Santa Barbara for multiple days ahead

4.2.2 Exploring the Parameters’ Significance for Price Forecasting

The work in Section 4.2.1 is further built upon to forecast strawberry prices. Since the market price depends on several factors that are not captured by remote sensing data, the farmer price is forecasted as it mainly relies on the yield; with higher yields typically leading to lower prices. From this, it can be deduced that the yield and price share similar influential factors that can be used for forecasting. It should be noted that while the price time series itself could be used to forecast prices in the future, the work focuses on the use of remote sensing data as they are more globally available for forecasting. The same preprocessed satellite images used in Section 4.2.1 are used in this experiment, while the models tested are LSTM and CNN-LSTM to forecast 35 days ahead. The MAE, RMSE, R^2 , and AGM scores of the tested models for the test set of 2019 and the second half of

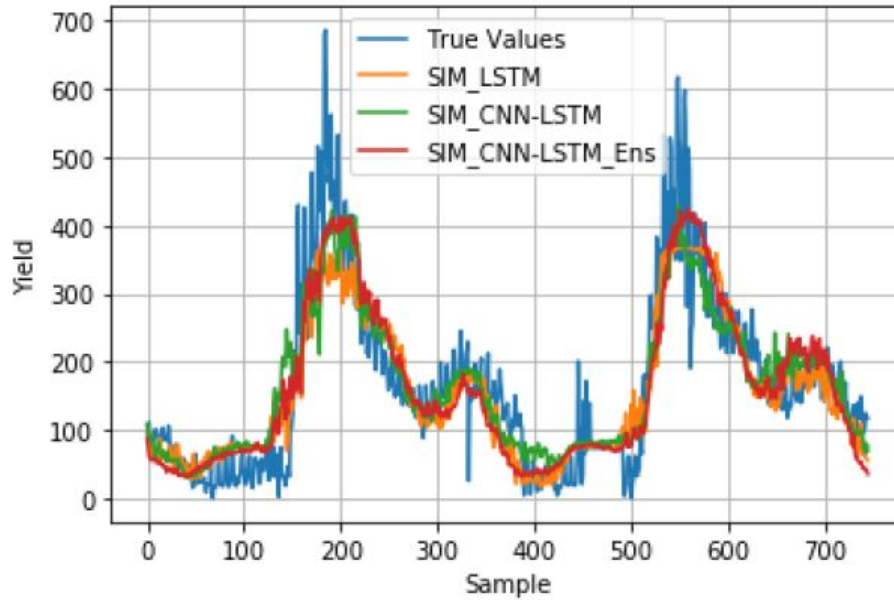


Figure 4.3: Forecasted LSTM, CNN-LSTM, and CNN-LSTM_Ens vs. true yield values

2018 are all presented in Table 4.6. It is worth noting that the CNN-LSTM Ensemble is omitted for price forecasting as it does not produce promising results. From the results, it is evident that the CNN-LSTM outperforms the LSTM in price forecasting as well by an AGM improvement of 50%.

Table 4.6: Results for forecasting prices using LSTM and CNN-LSTM

<i>Score</i>	<i>LSTM</i>	<i>CNN-LSTM</i>
MAE	0.26	0.21
RMSE	0.34	0.26
R^2	0.52	0.71
AGM	0.14	0.07

The forecasted time series of the results above are visualized in Figure 4.4, with the common window of forecasting spanning 596 days from the end of 2019 counting backwards, which roughly translates to the second half of 2018 and all 2019. The figure shows that the CNN-LSTM model is better at capturing the price trends compared to the LSTM, even

when missing some fluctuations. This limitation can be due to those fluctuations relying on factors not represented by the used soil parameters, which hinders the models ability to learn them.

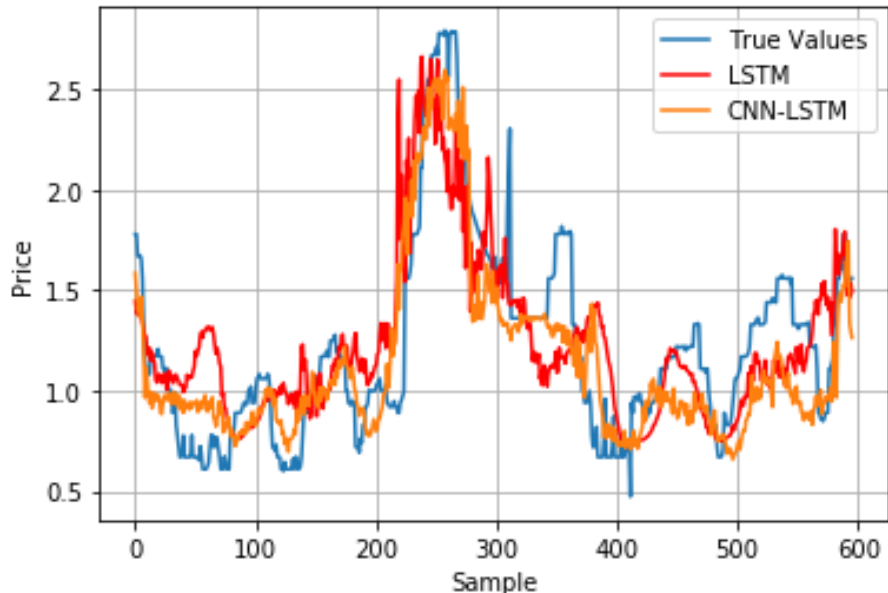


Figure 4.4: Forecasted values of price using LSTM and CNN-LSTM models versus true price values

4.3 Daily Yield Forecasting in Multiple Counties

To investigate the effect of location on forecasting daily yield, this set of experiments further explores applying CNN-LSTM, SAE, VAE, and voting regressor models in two additional counties which are compared to the same benchmark model in [112]. For this experiment, the arsenal of tested models is expanded by introducing the SAE and VAE classes of autoencoders for forecasting strawberry yield values. In addition, the case study is extended to cover 2 more counties in California that are prominent in strawberry production, namely Venture and Monterey, for more generalized experimentation. It is worth noting that the surface reflectance bands were not used for training in this experiment and onward as it was discovered they do not improve the performance significantly while being the largest of the bands in size; which is most likely due to redundancy with the temperature bands. The tested models in this experiment are the CNN model proposed in [112] which is used

as a benchmark as well as the proposed CNN-LSTM, SAE, VAE, and CNN-LSTM-SAE Ensemble models for forecasting 35 days ahead. CNN-LSTM and SAE were chosen to be ensembled by the voting regressor based on their similar performance. The MAE, RMSE, R^2 , and AGM scores of the deployed models are presented in Table 4.7.

Table 4.7: Yield forecasting performance of CNN, VAE, CNN-LSTM, SAE, & CNN-LSTM-SAE Ensemble models

Santa Barbara					
<i>Score</i>	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>	<i>CNN-LSTM-SAE Ensemble</i>
MAE	52.28	38.30	40.56	43.09	36.83
RMSE	72.55	56.69	55.07	62.53	52.90
R^2	0.67	0.80	0.81	0.76	0.83
AGM	20.40	9.48	9.01	12.82	7.80
Ventura					
<i>Score</i>	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>	<i>CNN-LSTM-SAE Ensemble</i>
MAE	62.21	41.91	41.61	44.45	38.61
RMSE	81.42	59.79	59.78	62.76	55.00
R^2	0.53	0.75	0.75	0.72	0.79
AGM	33.65	12.85	12.81	14.92	10.01
Monterey					
<i>Score</i>	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>	<i>CNN-LSTM-SAE Ensemble</i>
MAE	58.79	54.00	54.47	53.91	50.98
RMSE	82.63	79.30	78.12	84.84	75.25
R^2	0.86	0.87	0.88	0.86	0.89
AGM	9.67	8.39	8.10	10.00	7.16

From Table 4.7, it is evident that the CNN-LSTM-SAE Ensemble is better in forecasting yield in all three counties compared to its individual component models as well as the

proposed VAE model and the CNN benchmark from literature. This outcome matches expectations as the combination of the efforts of CNN-LSTM and SAE is expected to produce enhanced performance with reduced variance and better forecasting. CNN-LSTMs are praised for their ability to extract spatiotemporal dependencies, while SAEs are known for their ability to optimize their feature extraction through constrained feature space; combining their results would minimize the effect of their limitations. In addition, the CNN-LSTM and SAE perform on par with each other across all three counties, which could be an indication of the constraints of feature extraction limited by the information carried in the data. The proposed VAE model was expected to perform better, but since it does not utilize LSTM cells then temporal dependencies are most likely not captured. Further, the results are among across all three counties, which indicates that there are regional similarities among them.

The improvement percentage achieved by the tested models is better visualized in Table 4.8. It is clear from the table that the CNN-LSTM-SAE Ensemble has the greatest performance improvement compared to the CNN model proposed in literature especially in Ventura with the highest AGM improvement percentage compared to other tested counties.

Table 4.8: CNN-LSTM-SAE Ensemble yield forecasting AGM percentage improvement to the state-of-the-art models across three counties

<i>Counties</i>	Yield forecasting improvement % based on AGM achieved by CNN-LSTM-SAE Ensemble			
	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>
Santa Barbara	62%	18%	13%	39%
Ventura	70%	22%	22%	33%
Monterey	26%	15%	12%	28%

The results showcased in Table 4.8 are better summarized in Figure 4.5, the largest improvement being compared to the model proposed in [112], which is constrained by lacking the ability to extract temporal dependencies that are crucial in forecasting time series.

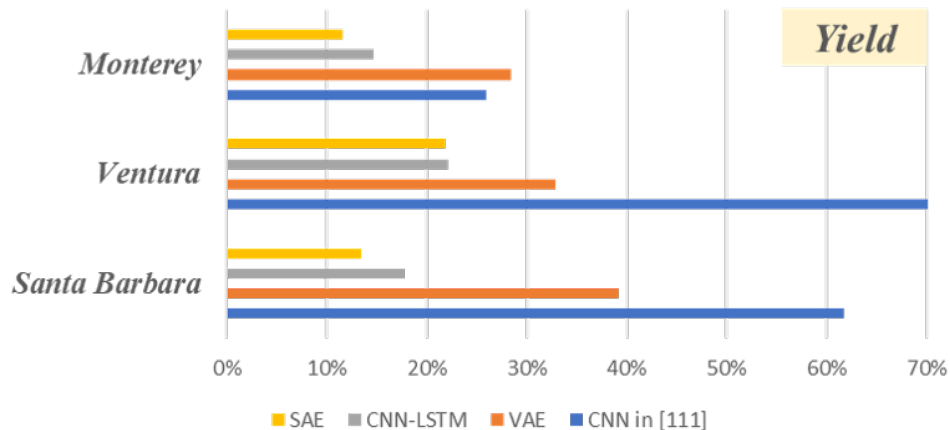


Figure 4.5: AGM percentage improvement in yield forecasting of the CNN-LSTM-SAE Ensemble to the CNN benchmark model, VAE, and its two component models: SAE and CNN-LSTM

In terms of computational performance, both the CNN-LSTM and SAE models took around 1 second per epoch and trained for 20 epochs while the VAE model took around 4 seconds per epoch and trained for 20 epochs including both the autoencoder and regression training. In addition, the CNN model proposed in [112] took around 10 seconds per epoch and trained for 20 epochs. The fact that the proposed models have lower computational costs compared to the literature CNN benchmark and yet outperform it indicates that increasing computational complexity does not always produce better results.

4.4 Daily Price Forecasting in Multiple Counties

A similar approach to the previous experiment is followed to investigate the capabilities of the proposed models in forecasting daily strawberry prices in the three counties. The models are compared to the literature model proposed in [112]. Section 4.3 experiment approach is utilized to explore the expanded number of tested models in forecasting strawberry prices in the three aforementioned counties; Santa Barbara, Ventura, and Monterey. It should be noted that the farmers' price is used in this experiment since it has a higher correlation to soil parameters compared to the market price which is influenced by external factors such as supply, demand, and socioeconomic factors that are not captured by remote sensing data. The tested models are the literature CNN model proposed in [112] as well as

the proposed CNN-LSTM, SAE, VAE, and CNN-LSTM-SAE Ensemble forecasting models for 35 days ahead. The MAE, RMSE, R^2 , as well as AGM scores of the deployed models are displayed in Table 4.9.

Table 4.9: Price forecasting performance of CNN, VAE, CNN-LSTM, SAE and CNN-LSTM-SAE Ensemble

Santa Barbara					
<i>Score</i>	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>	<i>CNN-LSTM-SAE Ensemble</i>
MAE	0.32	0.26	0.25	0.29	0.24
RMSE	0.43	0.32	0.32	0.37	0.30
R^2	0.23	0.58	0.58	0.42	0.64
AGM	0.29	0.12	0.12	0.19	0.10
Ventura					
<i>Score</i>	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>	<i>CNN-LSTM-SAE Ensemble</i>
MAE	0.27	0.24	0.24	0.27	0.23
RMSE	0.39	0.35	0.33	0.37	0.33
R^2	0.60	0.68	0.72	0.66	0.72
AGM	0.13	0.09	0.08	0.11	0.08
Monterey					
<i>Score</i>	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>	<i>CNN-LSTM-SAE Ensemble</i>
MAE	0.20	0.17	0.14	0.17	0.15
RMSE	0.27	0.24	0.23	0.27	0.23
R^2	0.32	0.49	0.52	0.35	0.54
AGM	0.16	0.10	0.09	0.14	0.09

Similar to the yield forecasting, it is clear from Table 4.9 that the voting ensemble outperforms both its individual component models, the CNN benchmark model, as well as the VAE when forecasting price in all three counties. Moreover, it should be noted that the

SAE performs on par with the CNN-LSTM, if not better. This could be credited to the feature extraction obtained through dimensionality reduction using the encoder-decoder LSTM setup.

The improvement percentage achieved by the tested models can be better visualized in Table 4.10. From the table, it is evident that the CNN-LSTM-SAE Ensemble is, on average, the best in price forecasting, with the SAE performing equally as good for forecasting price in both Ventura and Monterey. This illustrates that the performance of the models is dependent on the location’s historical soil parameters and price trends.

Table 4.10: CNN-LSTM-SAE Ensemble price forecasting AGM percentage improvement to the state-of-the-art models across three counties

<i>Counties</i>	Price forecasting improvement % based on AGM achieved by CNN-LSTM-SAE Ensemble			
	<i>CNN in [112]</i>	<i>CNN-LSTM</i>	<i>SAE</i>	<i>VAE</i>
Santa Barbara	66%	17%	17%	47%
Ventura	38%	11%	0%	27%
Monterey	44%	10%	0%	36%

The results presented in Table 4.10 are visualized for clarity in Figure 4.6, with the largest improvement being compared to the model proposed in [112] that relies mostly on convolutional techniques for spatial feature extraction.

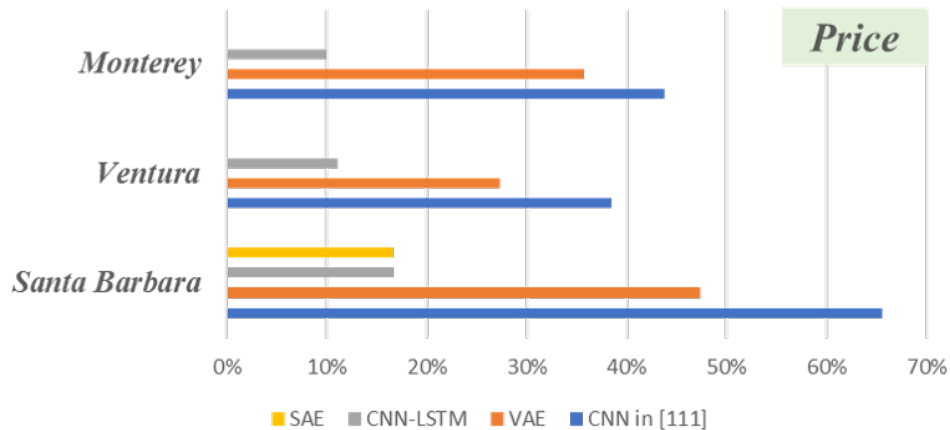


Figure 4.6: AGM percentage improvement in forecasting price of the CNN-LSTM-SAE Ensemble compared to the CNN benchmark model, VAE, and its two component models: SAE and CNN-LSTM

4.5 Incremental Learning for Yield Forecasting

To ensure the forecasting models remain up to date as new data is collected, incremental learning is investigated in this experiment. The experiment explores the effect of incrementally feeding the models with annual data on forecasting performance in forecasting daily strawberry yields. The experiment process followed is as described in Figure 2.5, where the model is incrementally trained with new batches of data. This approach, yet similar to batch learning in application, differs in the assumption that each batch is time dependent on the previous one, hence input sequence is important as the batches are not available at once. Moreover, it differs in its purpose which is the offline update of trained models using new information. The CNN-LSTM Ensemble is utilized in this experiment to forecast daily strawberry yield values, and its performance is evaluated as the model trains on newer years. For the experiment setup, the dataset used ranges from 2011 to 2019 and only the temperature and moisture bands are used, with last year of the dataset used for testing while the preceding 7 years are incrementally added to evaluate the incremental learning performance. Figure 4.7 shows the AGM scores of the CNN-LSTM Ensemble model as it trains incrementally on years' 2011 to 2018 data while always being tested on 2019 data. It is evident that the AGM remarkably improves over time, indicating that the model successfully updates its learning based on new years' data while retaining the vital

information it learns from previous years.

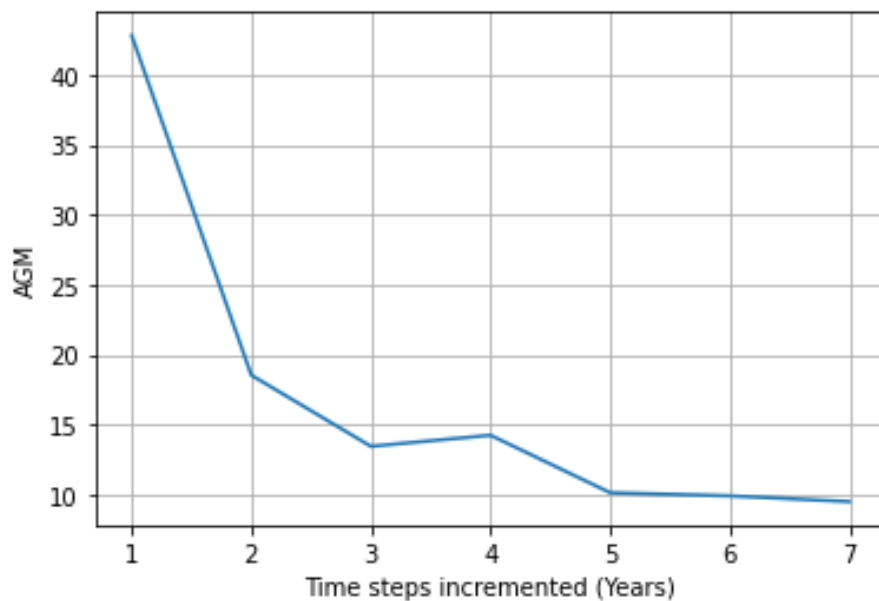


Figure 4.7: AGM forecasting scores of CNN-LSTM_Ens model in 2019 when trained using the incremented yearly data of 7 years from mid 2011 to 2018

The results of the forecasted yield for 2019 of the CNN-LSTM Ensemble model as well as its two individual CNN-LSTM components are presented in Figure 4.8. The results reiterate that the CNN-LSTM Ensemble model outperforms its individual components in forecasting daily yields with incremental learning as well, which supports the effectiveness of voting regressors.

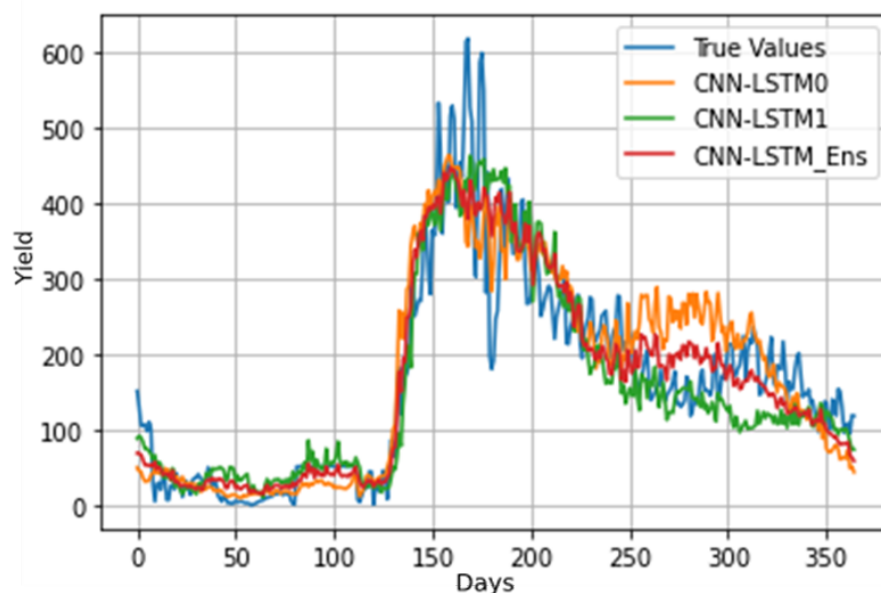


Figure 4.8: Forecasted yield values by CNN-LSTM_Ens model and its two CNN-LSTM components compared to true yield values in 2019

4.6 Transfer Learning for Yield Forecasting

To generalize the capabilities of the models, and to reduce their training costs, TL is applied to the models. TL can help transfer feature extraction of the models from forecasting one FP to forecasting other similar FPs. This experiment explores the effect of using TL in forecasting daily raspberry yields using knowledge extracted from strawberry forecasting models. The data used for the experiment has the temperature and moisture bands and the historical raspberry yield in Santa Barbara County. To test the effectiveness of TL, the models tested have the CNN-LSTM architecture described in Section 3.3.3 and include the model pretrained on strawberry yield, the pretrained model with TL applied, and a new model trained from scratch on raspberry yield without TL. For this experiment to be effective, the crops must be similar in nature for sufficient knowledge from one to be effective on the other. Thus, a similarity investigation is presented in Figure 4.9, which indicates that both crops share a common annual seasonality with the yield fluctuating at similar cycles. The key difference is the magnitude of the yield; with strawberry yield being higher than raspberry yield, however this difference in scale would not hinder the TL. In addition, since the experimentation county for both tested FPs is the same, Santa Barbara,

then the temperature and moisture data is identical, adding to the potential effectiveness of using TL.

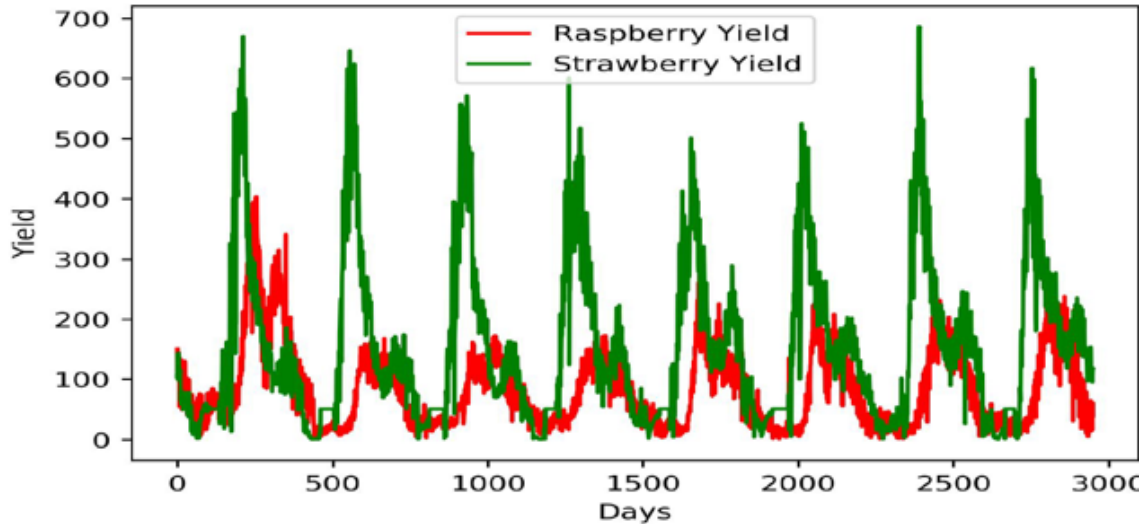


Figure 4.9: Comparison between strawberry and raspberry yields from 2011 to 2019

Table 4.11 presents the computational time, MAE, RMSE, R^2 , and AGM of the three tested models. The first model is pretrained on strawberry yield data without any modification, the second model takes the weights of the convolution layers of the pretrained model and fine-tunes the remaining LSTM and fully connected layers using raspberry data, and the third model is trained on raspberry yield with randomly initialized weights. The found results show that the unmodified pretrained CNN-LSTM is incapable of forecasting raspberry yield using only knowledge of strawberry yield. Moreover, the TL CNN-LSTM is the best performing model compared to the other two, with an AGM score lower by almost 28% and a shorter computational time by 49% compared to the raspberry CNN-LSTM. The reduction in computational time is due to the reduced number of parameters that need to be learned compared to the full trained raspberry CNN-LSTM. Moreover, the results indicate that the spatial feature extraction learned from strawberry yield provides improved forecasting performance even when compared to learning purely using raspberry data.

Table 4.11: Evaluation metrics scores of CNN-LSTM using TL compared to non-TL approaches

<i>Score</i>	<i>Pretrained CNN-LSTM</i>	<i>TL CNN-LSTM</i>	<i>Raspberry CNN-LSTM</i>
Time (sec)	0	36	71
MAE	98.22	21.26	23.50
RMSE	146.20	28.74	32.06
R^2	-3.88	0.812	0.766
AGM	595.99	4.71	6.51

The actual forecasted yield values for all three tested models are depicted in Figure 4.10 which indicate that the pretrained model forecasted values in subfigure (a) follow a trend much different than raspberry yields, a trend closer to strawberry yields. This is expected since it is trained solely using strawberry yields. Subfigure (b) presents the TL model's forecasted yields while subfigure (c) shows the model trained on raspberry yields without TL. Both models are able to follow the raspberry data trend with slight variation.

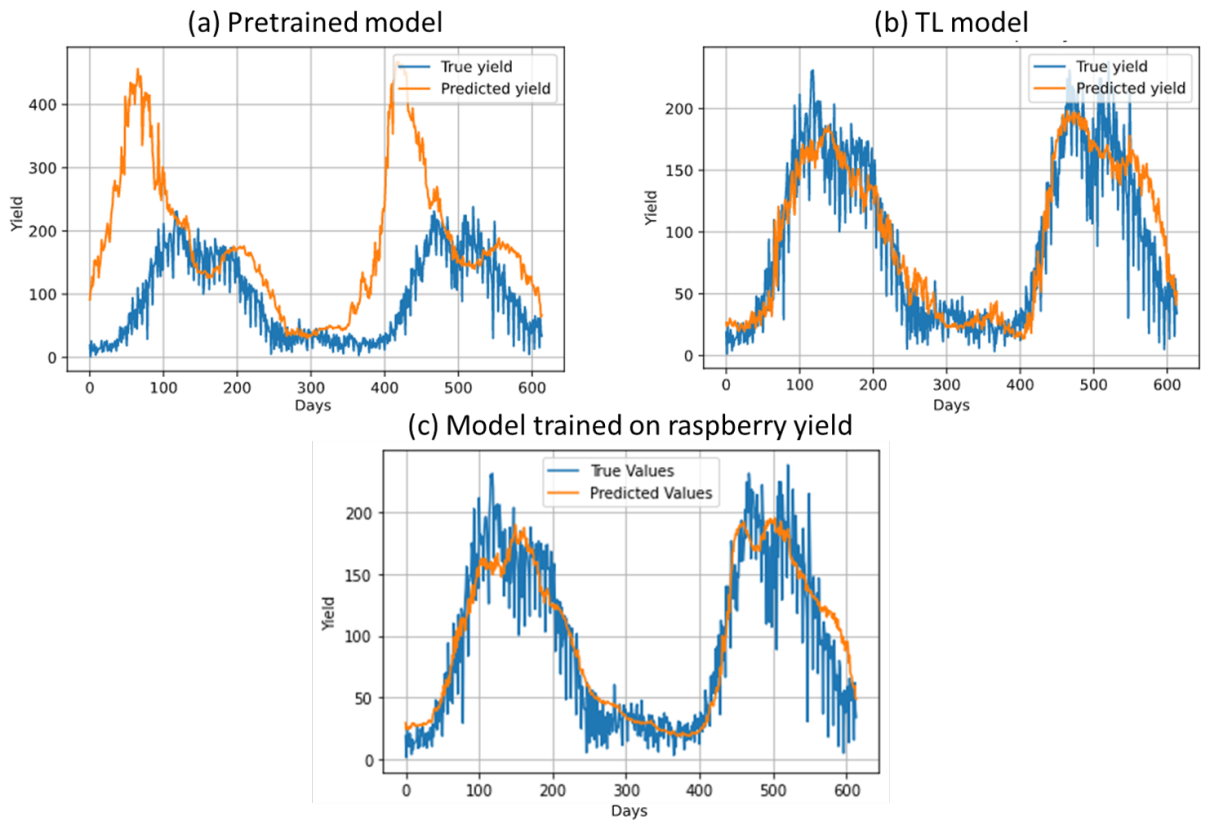


Figure 4.10: The true raspberry yield values versus the yields forecasted by (a) model pretrained on strawberry yield, (b) TL model, and (c) model trained on raspberry yield without TL

4.7 Chapter Summary

This chapter provides details of the conducted experiments, along with their results and results entail.

The first set of experiments described in this chapter focuses on the USA Midwest, it further explores the use of deep learning models in predicting annual soybean yields and compares the performance of the proposed models with literature. Moreover, the effect of adding moisture bands is investigated with the deployed models. It is found that the proposed CNN-LSTM outperforms the literature model by an average RMSE percentage

improvement of 31% and the addition of the satellite images of surface and subsurface moisture levels improves the prediction performance.

The second experiment investigates the generalizability of the approach by applying it to Santa Barbara, California to forecast daily strawberry yield and price. The experiment explored the effect of varying the forecasting window on the models' performance and it was found that the deployed models consistently lose forecasting effectiveness the further they forecast in the future. In addition, the CNN-LSTM Ensemble is found to outperform each of its components as well as the LSTM. The experiments also investigate the performance of the models in price forecasting, and concluded that the CNN-LSTM outperforms the LSTM.

The third set of experiments investigates the approach with new proposed models and across different counties. The added models were SAE and VAE along with a voting regressor, while the new counties explored were Ventura and Monterey. The results indicate that the CNN-LSTM-SAE Ensemble outperforms the other deployed models, including the literature model by up to 70% AGM improvement for yield forecasting and 66% for price forecasting.

The fourth experiment is exploring the use of incremental learning with CNN-LSTM Ensemble for yield forecasting to update it upon arrival of new data. The results show that over time, the forecasting AGM consistently drops as new data is introduced, indicating that the CNN-LSTM Ensemble does not lose forecasting performance when incrementally training on new data. The last experiment explores the application of TL to generalize the approach. Raspberry yields are forecasted using a model pretrained on strawberry yield, a pretrained model with TL applied, and a new model trained on raspberry yield without TL. The results show that the use of TL provides the best performance with a reduction in computation time.

Chapter 5

Web Application

The previous chapter presented the conducted experiments including their results and analysis. This chapter showcases the design and implementation of the web application that combines the experiments to create an interface and application for forecasting yield and price in a given county for a given FP type requested by the user.

5.1 Design and Implementation

The design of the application is divided into three distinct operation cases as presented in Figure 5.1. The first case utilizes pretrained models directly for forecasting, while the second and third cases modify those models through TL to be applicable for similar FPs. For all cases, the common required inputs are the forecast date range, output type whether it is yield or price, forecast horizon which can be 1 day, 1 week, 2 weeks, 3 weeks, 4 weeks, or 5 weeks ahead, county, and FP type. It should be noted that each horizon corresponds to a pretrained model trained on that horizon. The options exist to give the user flexibility in how far they choose to forecast at the cost of reduced accuracy. In addition, the county variable determines the satellite images input data, while the FP type variable determines the output data. Moreover, the output would be presented as a single value if the requested forecast is for a single day and a graph is provided if the requested forecast is for a range of days.

The first case is when the user requires forecasting for a FP type in a county for which pretrained models are stored in the system, so only the aforementioned common inputs are required. The second case is when the user requests forecasting for a new FP type,

that does not have a stored pretrained model, in one of the previously studied counties with satellite data predownloaded and stored. In this case, the user is requested to input an additional file containing the output data for yield or price which will be used for TL. The third case is used when the user requests forecasts for a FP type in a new county where pretrained models are unavailable but the FP yield or price, output, as well as the county soil conditions, input, are similar to other FPs and counties for which pretrained models are available. In this case, TL can be applied as well and the user is requested to provide the FP yield and price output data for TL as well as the FIPS code of the specified county which is needed to download the satellite images. It is key to note that this approach only applies forecasting for input and output data that are deemed similar, which is determined by applying the similarity check proposed in [38]. If the similarity check proves the two FPs to be dissimilar, based on input and output, the application does not proceed with forecasting; building a separate model trained entirely on the newly provided data is suggested.

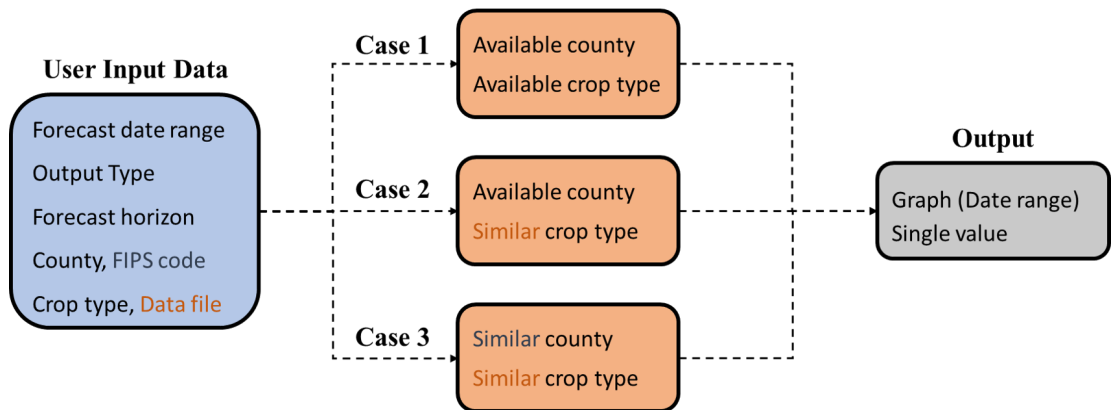


Figure 5.1: Web application design flowchart

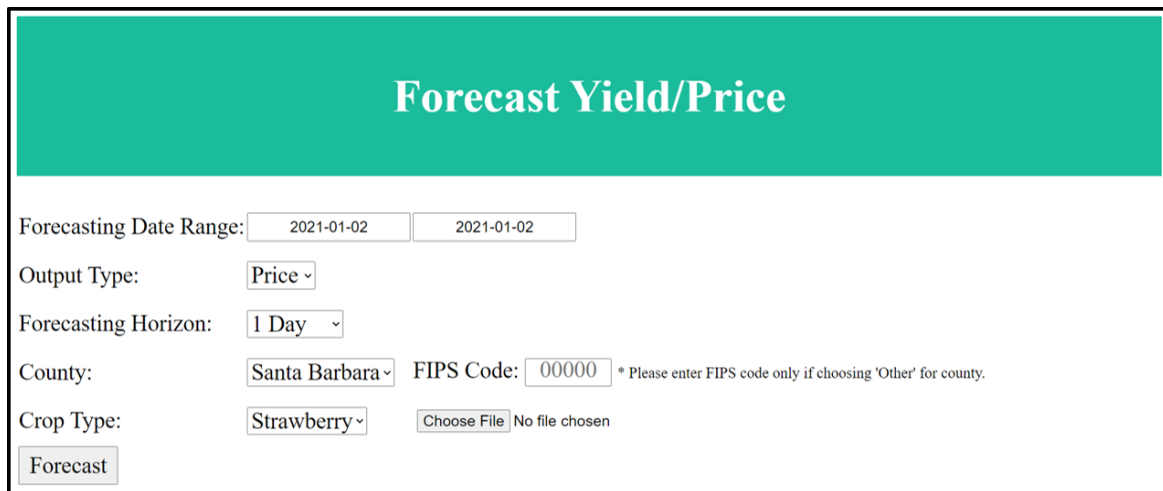
The web application is implemented using Python language and Flask library [28] for back-end programming and HyperText Markup Language (HTML) for front-end programming of the interface.

5.2 Testing

After the application is designed and implemented, several test cases are applied to cover and present most of the application’s functionality. For all cases, the current day is assumed to be 1st January 2021 and forecasting ahead of that.

Test Case 1: Forecasting price with available county & FP

For the first test case, the scenario tested for price forecasting as shown in Figure 5.2. The county chosen is Santa Barbara and the requested forecast is for strawberry yields. Moreover, the requested output forecast is set to a single day with the horizon set to use a 1 day ahead forecasting model.



The screenshot shows a web application interface titled "Forecast Yield/Price". The interface includes several input fields and a "Forecast" button. The "Forecasting Date Range" is set to "2021-01-02" to "2021-01-02". The "Output Type" is set to "Price". The "Forecasting Horizon" is set to "1 Day". The "County" is set to "Santa Barbara" and the "FIPS Code" is "00000". A note next to the FIPS code says "* Please enter FIPS code only if choosing 'Other' for county." The "Crop Type" is set to "Strawberry" and there is a "Choose File" button with the text "No file chosen". A "Forecast" button is located at the bottom left of the form.

Figure 5.2: Application screenshot showing the user input for test case 1

The resulting forecasted price is given to the user as depicted in the output screenshot in Figure 5.2 is shown in Figure 5.3 as a single forecasted value output.

Forecasted Price: \$ 1.3983158

Figure 5.3: Screenshot showing the application output for test case 1

Test Case 2: Forecasting yield with available county & FP

A test case scenario for yield forecasting is also implemented with the input parameters shown in Figure 5.4. In this case, the user requests a forecast for 1 week ahead of the assumed current day using the 1 week ahead horizon model.

Forecast Yield/Price

Forecasting Date Range:

Output Type:

Forecasting Horizon:

County: FIPS Code: * Please enter FIPS code only if choosing 'Other' for county.

Crop Type: No file chosen

Figure 5.4: Application screenshot showing the user input for test case 2

The output of the application is presented in Figure 5.5 in graph format.

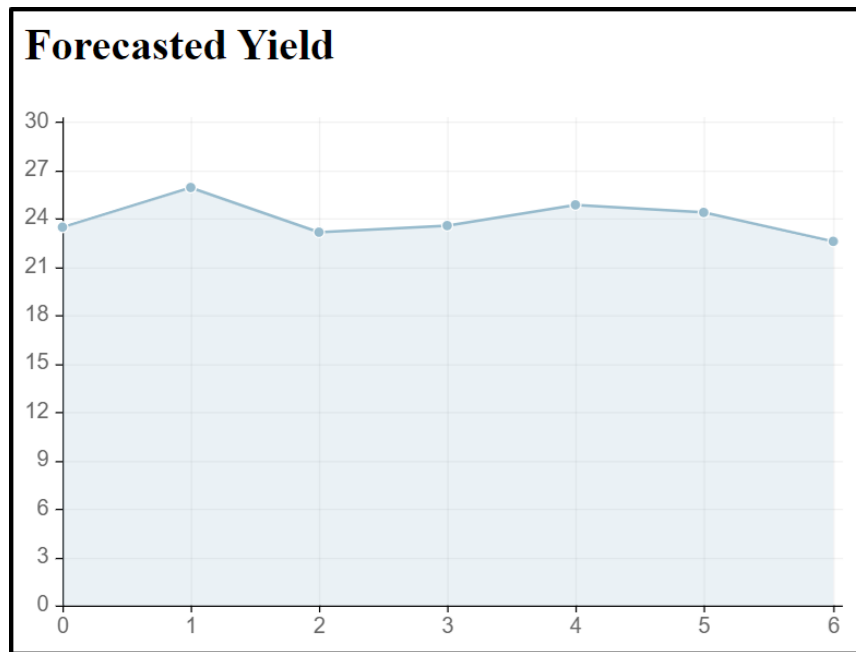
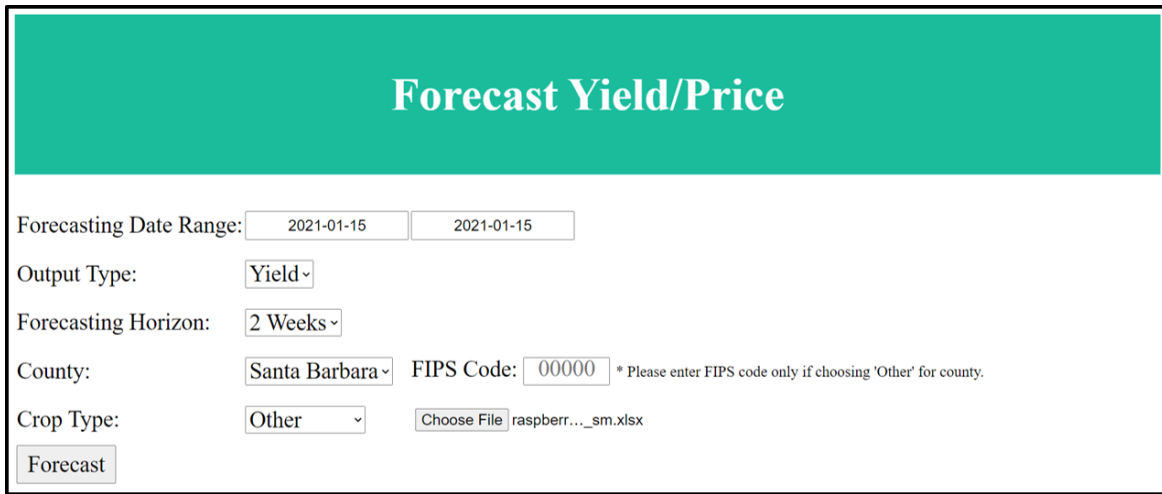


Figure 5.5: Screenshot showing the application output for test case 2

Test Case 3: Forecasting one day yield with available county & similar FP

For testing the second operation case, Case 2 in Figure 5.1, the requested forecast is for Santa Barbara County which has available data stored, but the FP type is of raspberry which does not have a stored pretrained model. Since the data of raspberry and strawberry FPs is verified to be similar, the application applies TL using the raspberry data uploaded by the user and the pretrained strawberry model. The parameters set for the scenario as well as the uploaded raspberry yield file are presented in Figure 5.6.



The screenshot shows a web application interface titled "Forecast Yield/Price". The interface includes the following input fields and controls:

- Forecasting Date Range: Two date input fields, both containing "2021-01-15".
- Output Type: A dropdown menu set to "Yield".
- Forecasting Horizon: A dropdown menu set to "2 Weeks".
- County: A dropdown menu set to "Santa Barbara".
- FIPS Code: An input field containing "00000" with a note: "* Please enter FIPS code only if choosing 'Other' for county."
- Crop Type: A dropdown menu set to "Other".
- File Upload: A "Choose File" button followed by the filename "raspberr..._sm.xlsx".
- Forecast: A button at the bottom left of the form.

Figure 5.6: Application screenshot showing the user input for test case 3

Figure 5.7 illustrates the output forecast based on the input parameters shown in Figure 5.6.

Forecasted Yield: 28.238825 Pounds/Acre

Figure 5.7: Screenshot showing the application output for test case 3

Test Case 4: Forecasting range of yields with available county & similar FP

A similar test scenario is applied for forecasting a range of days specified in Figure 5.8 along with other input parameters.

Forecast Yield/Price

Forecasting Date Range:

Output Type:

Forecasting Horizon:

County: FIPS Code: * Please enter FIPS code only if choosing 'Other' for county.

Crop Type: raspberr..._sm.xlsx

Figure 5.8: Application screenshot showing the user input for test case 4

Figure 5.9 presents the requested forecasts in a graph format based on the input parameters shown in Figure 5.8.

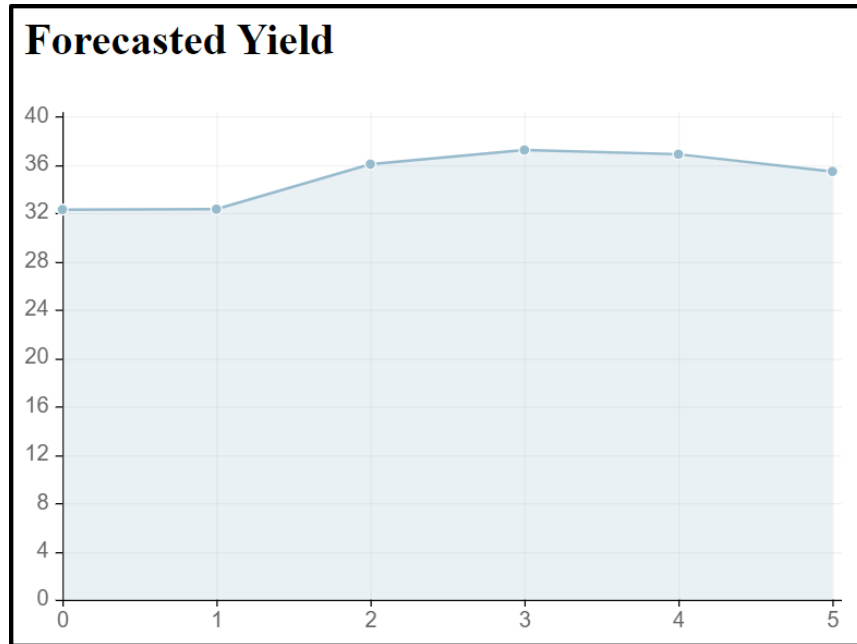
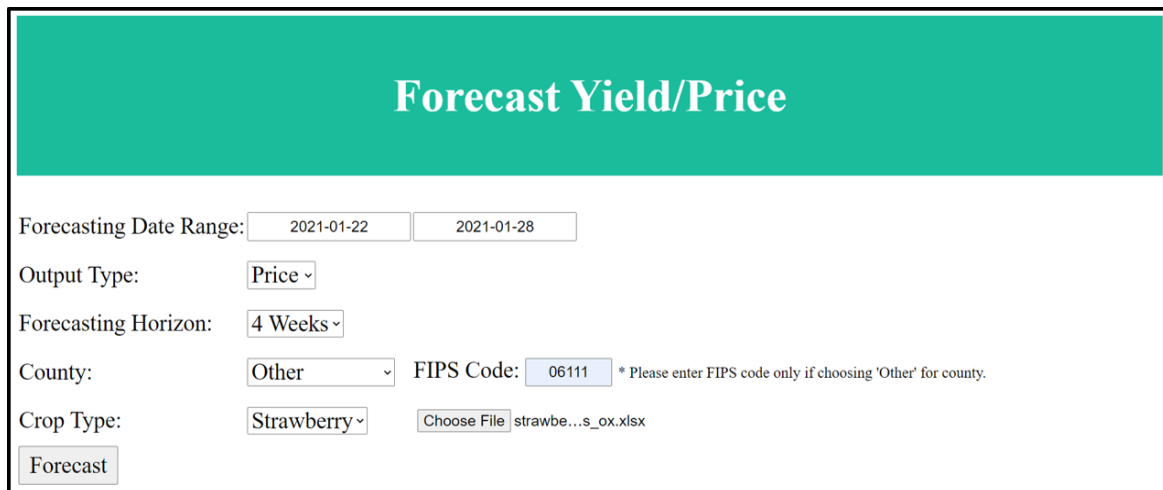


Figure 5.9: Screenshot showing the application output for test case 4

Test Case 5: Forecasting price for similar FP in similar county

The last operation case is tested by requesting forecasts for strawberries in Ventura county that has neither its satellite images nor pretrained models available in storage. Thus, the user is required to upload the FP output data file as well as the FIPS code of the requested county which is '06111' for Ventura. The parameters chosen for this scenario and the uploaded Ventura data file are shown in Figure 5.10.



The screenshot shows a web application interface titled "Forecast Yield/Price". The interface includes the following input fields and controls:

- Forecasting Date Range:** Two date input fields with values "2021-01-22" and "2021-01-28".
- Output Type:** A dropdown menu with "Price" selected.
- Forecasting Horizon:** A dropdown menu with "4 Weeks" selected.
- County:** A dropdown menu with "Other" selected.
- FIPS Code:** A text input field with "06111" entered. A note next to it reads: "* Please enter FIPS code only if choosing 'Other' for county."
- Crop Type:** A dropdown menu with "Strawberry" selected.
- File Upload:** A "Choose File" button followed by the filename "strawbe...s_ox.xlsx".
- Action:** A "Forecast" button.

Figure 5.10: Application screenshot showing the user input for test case 5

The output of the input parameters in Figure 5.10 is depicted in Figure 5.11 graph.

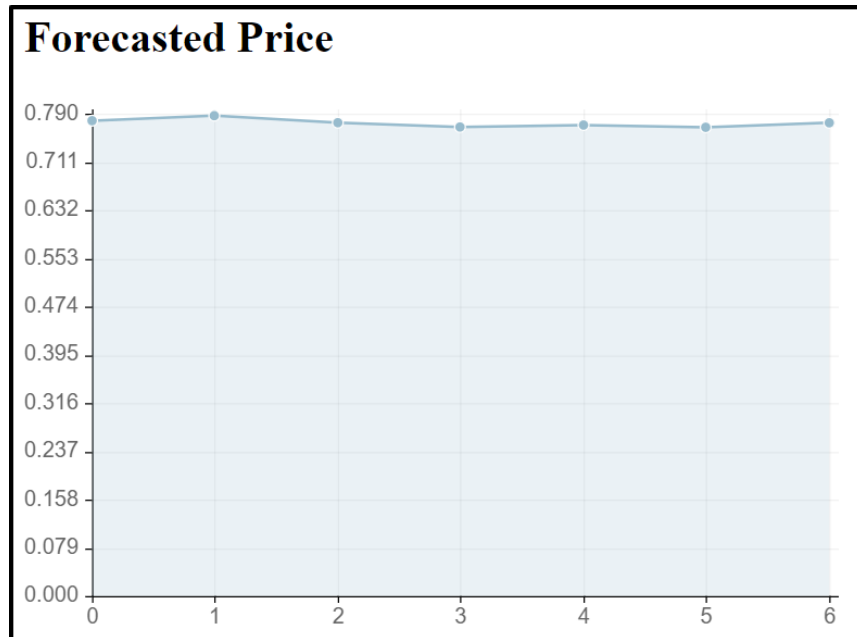


Figure 5.11: Screenshot showing the application output for test case 5

Test Case 6: Forecasting yield for similar FP in similar county

Another test scenario is also implemented for yield forecasting with the input parameters displayed in Figure 5.12. The user requests forecasting for 5 weeks ahead of the assumed current day using the 5 weeks ahead horizon model.

Forecast Yield/Price

Forecasting Date Range:

Output Type:

Forecasting Horizon:

County: FIPS Code: * Please enter FIPS code only if choosing 'Other' for county.

Crop Type: strawbe...s_ox.xlsx

Figure 5.12: Application screenshot showing the user input for test case 6

Figure 5.13 illustrates the output forecasts in graph format based on the input parameters chosen in Figure 5.12.

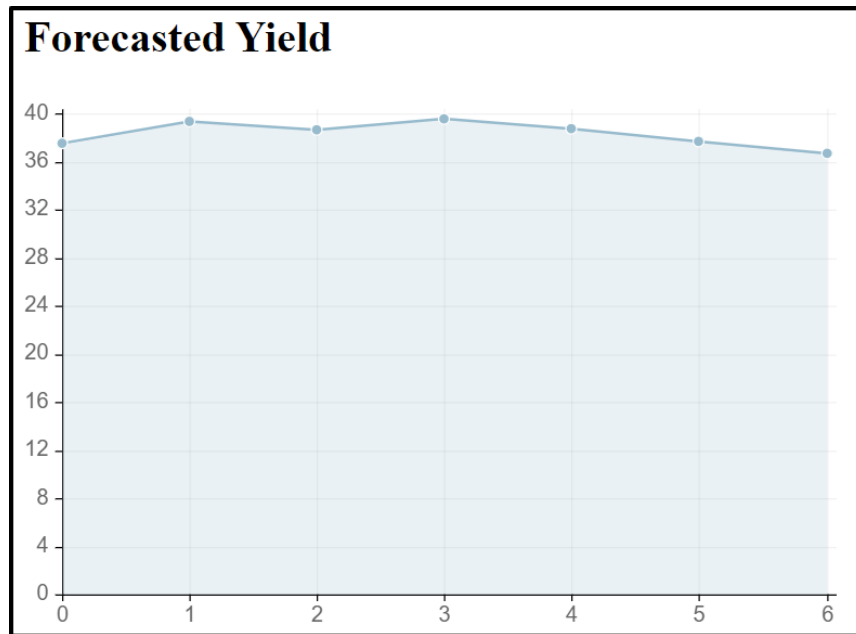


Figure 5.13: Screenshot showing the application output for test case 6

5.3 Chapter Summary

This chapter presents and describes the web application designed and implemented for forecasting FP yield and price. Test cases are applied to each of the three proposed functionalities and the results indicate the web application operates as expected.

Chapter 6

Conclusions and Future Work

The main objective of this thesis is to explore and address the challenges faced in FP yield and price forecasting using satellite images. The work investigates the limitations in literature such as limited generalizability and more frequent forecasting, and proposes solutions. The proposed solution includes several novel forecasting model architectures and ensemble techniques for FP yields and prices. The efficacy of the proposed models is explored in various experimental setups using simple and comprehensive evaluation metrics. Moreover, dimensionality reduction techniques are adapted by literature when processing satellite images.

The work presents several conducted experiments and their results. The first experiment compares the proposed models to literature to gauge their effectiveness. It applies the models to the same Midwestern counties applied in literature to predict annual soybean yield and investigates the effect of moisture data on prediction performance. It is observed that the proposed CNN-LSTM outperforms the model in literature by an average RMSE percentage improvement of 31%. In addition, adding satellite images of surface and subsurface moisture levels improves the prediction. The second experiment takes the framework of the approach and applies it to forecasting strawberry yield and price in Santa Barbara, California. Specifically, this experiment investigates the effect of varying the forecasting window on the forecasting performance and concludes a consistent decrease in accuracy as the window shifts further into the future. In terms of models' comparison, the CNN-LSTM Ensemble is found to outperform the other deployed models in yield forecasting, while the CNN-LSTM outperforms the LSTM in forecasting price. Afterwards, the third experiment expands the scope of the work by including autoencoders and additional counties in California. The results show that the CNN-LSTM-SAE Ensemble is the best at forecasting yield and price compared to other deployed models, including the model described in literature

by up to 70% for yield and 66% for price. Furthermore, the work explores an incremental learning approach to examine the adaptability of the deployed model to updates upon the arrival of new data. The results indicate that the deployed CNN-LSTM Ensemble forecasting performance does not degrade due to vanishing gradient as it trains incrementally on new data. Moreover, the thesis investigates the application of TL to help generalize the pretrained models to forecast similar FPs yields and prices. The experimental results conclude that using TL provides the best performance with nearly 28% improvement in AGM along with 49% reduction in computation time compared to a non-TL approach.

Finally, a forecasting web application for FP yields and prices is designed and implemented with a friendly user interface. The application utilizes models and approaches obtained from the experiments to increase its scope.

Along with the noted achievements of this work, some limitations should be acknowledged and tackled as future work. For the experiments that involve comparison with literature, a wider variety of existing models and approaches could be compared to the proposed models to authenticate their performance and novelty. On top of that, additional evaluation metrics could be used for cross-dataset comparison to better study model performance applied to different datasets; namely the Mean Absolute Scaled Error (MASE) as suggested by [37, 23]. Moreover, the availability of FP data is a major limiting factor in determining the scope of the experiments, thus expanding the scope of the datasets as well as the models would help verify and improve the obtained results. Finally, investigating the performance on new FPs and counties is vital to further gauge and enhance the capabilities of the models and application.

References

- [1] National Aeronautics and Space Administration. NASA-USDA Global Soil Moisture Data. <https://earth.gsfc.nasa.gov/hydro/data/nasa-usda-global-soil-moisture-data>, accessed 16 November 2020.
- [2] European Space Agency. About SMOS. <https://earth.esa.int/eogateway/missions/smos>, accessed 12 February 2021.
- [3] Nesreen Ahmed, Amir Atiya, Neamat Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29:594–621, 08 2010.
- [4] North American Raspberry & Blackberry Association. Overview of the caneberry industry: Facts & figures. <https://www.raspberrylblackberry.com/consumers/overview-of-the-caneberry-industry-facts-figures/>, accessed 20 March 2021.
- [5] Yun Bai, Yong Li, Bo Zeng, Chuan Li, and Jin Zhang. Hourly pm2.5 concentration forecast using stacked autoencoder model with emphasis on seasonality. *Journal of Cleaner Production*, 224:739–750, 2019.
- [6] Dana H Ballard. Modular learning in neural networks. In *AAAI*, pages 279–284, 1987.
- [7] Domenico Benvenuto, Marta Giovanetti, Lazzaro Vassallo, Silvia Angeletti, and Massimo Ciccozzi. Application of the arima model on the covid-2019 epidemic dataset. *Data in Brief*, 29:105340, 2020.
- [8] Simone Bianco, Gianluigi Ciocca, Paolo Napoletano, and Raimondo Schettini. An interactive tool for manual, semi-automatic and automatic video annotation. *Computer Vision and Image Understanding*, 131:88–99, 02 2015.

- [9] Alexei Botchkarev. A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14:045–076, 2019.
- [10] Gavin Brown. Ensemble learning. *Encyclopedia of machine learning*, 312:15–19, 2010.
- [11] Lorenzo Bruzzone and D Fernandez Prieto. An incremental-learning neural network for the classification of remote-sensing images. *Pattern Recognition Letters*, 20(11-13):1241–1248, 1999.
- [12] Wilhelm Burger. *Digital Image Processing An Algorithmic Introduction Using Java*. Texts in Computer Science. Springer London, London, 1st ed. edition, 2008.
- [13] California Strawberry Commission. Custom District Reports. <https://www.calstrawberry.com/en-us/market-data/district-report>, accessed 16 November 2020.
- [14] Hugo C.C. Carneiro, Felipe M.G. França, and Priscila M.V. Lima. Multilingual part-of-speech tagging with weightless neural networks. *Neural Networks*, 66:11–21, 2015.
- [15] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [16] Tianfeng Chai and R.R. Draxler. Root mean square error (RMSE) or mean absolute error (MAE)?– Arguments against avoiding RMSE in the literature. *Geoscientific Model Development*, 7:1247–1250, 06 2014.
- [17] François Chollet et al. Keras. <https://keras.io>, 2015.
- [18] NASA EOSDIS Land Processes Distributed Active Archive Center (LP DAAC). LP DAAC MODIS land products. <http://lpdaac.usgs.gov/>, accessed 16 November 2020.
- [19] Damien Sulla-Menashe and Mark A Friedl. User Guide to Collection 6 MODIS Land Cover (MCD12Q1 and MCD12C1) Product, 2018. https://lpdaac.usgs.gov/documents/101/MCD12_User_Guide_V6.pdf, accessed 16 November 2020.

- [20] Erick Meira de Oliveira and Fernando Luiz Cyrino Oliveira. Forecasting mid-long term electric energy consumption through bagging arima and exponential smoothing methods. *Energy*, 144:776–788, 2018.
- [21] Zhengxiao Du, Jie Tang, and Yuhui Ding. POLAR: Attention-Based CNN for One-Shot Personalized Article Recommendation. In Michele Berlingerio, Francesco Bonchi, Thomas Gärtner, Neil Hurley, and Georgiana Ifrim, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 675–690, Cham, 2019. Springer International Publishing.
- [22] Clay Ford. *Is R-squared Useless?*, October 2015. <https://data.library.virginia.edu/is-r-squared-useless/>, accessed 10 March 2021.
- [23] Philip Hans Franses. A note on the mean absolute scaled error. *International Journal of Forecasting*, 32(1):20–22, 2016.
- [24] Kuniyiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, April 1980.
- [25] Alexander Gepperth and Barbara Hammer. Incremental learning algorithms and applications. In *European Symposium on Artificial Neural Networks (ESANN)*, Bruges, Belgium, 2016.
- [26] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [27] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.
- [28] Miguel Grinberg. *Flask web development: developing web applications with python*. O’Reilly Media, Inc., 2018.
- [29] James Douglas Hamilton. 11.6. *Vector Autoregressions and Structural Econometric Models*, pages 324–336. Princeton University Press, December 1994.
- [30] Keith W Hipel and A Ian McLeod. *Time series modelling of water resources and environmental systems*. Elsevier, 1994.
- [31] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

- [32] Zixin Hu, Qiyang Ge, Shudi Li, Li Jin, and Momiao Xiong. Artificial intelligence forecasting of Covid-19 in China. *arXiv preprint arXiv:2002.07112*, 2020.
- [33] Chiou-Jye Huang and Ping-Huan Kuo. A deep CNN-LSTM model for particulate matter (PM_{2.5}) forecasting in smart cities. *Sensors*, 18(7):2220, 2018.
- [34] Lei Huang, Xianglong Liu, Binqiang Ma, and Bo Lang. Online semi-supervised annotation via proxy-based local consistency propagation. *Neurocomputing*, 149:1573–1586, 2015.
- [35] A.R. Huete. 11 - Remote Sensing For Environmental Monitoring. In Janick F. Artiola, Ian L. Pepper, and Mark L. Brusseau, editors, *Environmental Monitoring and Characterization*, pages 183–206. Academic Press, Burlington, 2004.
- [36] Rob J Hyndman and George Athanasopoulos. Forecasting: principles and practice, 2nd edition, 2018. <http://OTexts.com/fpp2>, accessed 25 May 2021.
- [37] Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.
- [38] Fatemeh Jafari, Lobna Nassar, and Fakhri Karray. Time series similarity analysis framework in fresh produce yield forecast domain. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2021. Submission no. #711.
- [39] K. U. Jaseena and Binsu C. Koor. A hybrid wind speed forecasting model using stacked autoencoder and lstm. *Journal of Renewable and Sustainable Energy*, 12(2):023302, 2020.
- [40] Zehui Jiang, Chao Liu, Nathan P. Hendricks, Baskar Ganapathysubramanian, Dermot J. Hayes, and Soumik Sarkar. Predicting county level corn yields using deep long short term memory models. *ArXiv*, abs/1805.12044, 2018.
- [41] David 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*, 141:116–128, 2014.
- [42] Devinder Kaur, Shama Naz Islam, Md Mahmud, et al. A VAE-Based Bayesian Bidirectional LSTM for Renewable Energy Forecasting. *arXiv preprint arXiv:2103.12969*, 2021.

- [43] Nobuaki Kimura, Ikuo Yoshinaga, Kenji Sekijima, Issaku Azechi, and Daichi Baba. Convolutional neural network coupled with a transfer-learning approach for time-series flood predictions. *Water*, 12(1), 2020.
- [44] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [45] F. Kratzert, D. Klotz, C. Brenner, K. Schulz, and M. Herrnegger. Rainfall-runoff modelling using Long Short-Term Memory (LSTM) networks. *Hydrology and Earth System Sciences*, 22(11):6005–6022, 2018.
- [46] T Senthil Kumar. Video based traffic forecasting using convolution neural network model and transfer learning techniques. *Journal of Innovative Image Processing (JIIP)*, 2(03):128–134, 2020.
- [47] Tarald O. Kvalseth. Cautionary Note about R2. *The American Statistician*, 39(4):279–285, 1985.
- [48] Tuong Le, Minh Thanh Vo, Tung Kieu, Eenjun Hwang, Seungmin Rho, and Sung Wook Baik. Multiple electric energy consumption forecasting using a cluster-based strategy for transfer learning in smart building. *Sensors*, 20(9), 2020.
- [49] Yann Lecun and Francoise Soulie Fogelman. Modeles connexionnistes de l'apprentissage. *Intellectica, special issue apprentissage et machine*, 2, January 1987.
- [50] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.
- [51] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.
- [52] Ioannis E Livieris, Emmanuel Pintelas, and Panagiotis Pintelas. A CNN-LSTM model for gold price time-series forecasting. *Neural computing and applications*, 32(23):17351–17360, 2020.
- [53] Vincenzo Lomonaco and Davide Maltoni. Comparing incremental learning strategies for convolutional neural networks. In Friedhelm Schwenker, Hazem M. Abbas, Neamat El Gayar, and Edmondo Trentin, editors, *Artificial Neural Networks in Pattern Recognition*, pages 175–184, Cham, 2016. Springer International Publishing.

- [54] Jie Lu, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, and Guangquan Zhang. Transfer learning using computational intelligence: A survey. *Knowledge-Based Systems*, 80:14 – 23, 2015. 25th anniversary of Knowledge-Based Systems.
- [55] Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. Towards understanding regularization in batch normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [56] Fei Lv, Min Han, and Tie Qiu. Remote sensing image classification based on ensemble extreme learning machine with stacked autoencoder. *IEEE Access*, 5:9021–9031, 2017.
- [57] M. Shahbandeh. Strawberry production in the United States in 2019, by state (in 1,000 cwt), May 2020. <https://www.statista.com/statistics/194235/top-10-strawberry-producing-us-states/>, accessed 20 March 2021.
- [58] Shukla Manish and Jharkharia Sanjay. Agri-fresh produce supply chain management: a state-of-the-art literature review. *International Journal of Operations & Production Management*, 33(2):114–158, January 2013.
- [59] Francesco Marra, Cristiano Saltori, Giulia Boato, and Luisa Verdoliva. Incremental learning for the detection and classification of gan-generated images. In *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, 2019.
- [60] Ayaka Masaki, Kent Nagumo, Bikash Lamsal, Kosuke Oiwa, and Akio Nozawa. Anomaly detection in facial skin temperature using variational autoencoder. *Artificial Life and Robotics*, 26(1):122–128, February 2021.
- [61] Michele Meroni, François Waldner, Lorenzo Seguini, Hervé Kerdiles, and Felix Rembold. Yield forecasting with machine learning and small data: what gains for grains? *arXiv preprint arXiv:2104.13246*, 2021.
- [62] Ronaldo Messina and Jérôme Louradour. Segmentation-free handwritten chinese text recognition with lstm-rnn. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 171–175, 2015.
- [63] Lobna Nassar, Ifeanyi Emmanuel Okwuchi, Muhammad Saad, Fakhri Karray, and Kumaraswamy Ponnambalam. Deep learning based approach for fresh produce market price prediction. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2020.

- [64] Lobna Nassar, Ifeanyi Emmanuel Okwuchi, Muhammad Saad, Fakhri Karray, Kumaraswamy Ponnambalam, and Prarabdhya Agrawal. Prediction of strawberry yield and farm price utilizing deep learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2020.
- [65] Ramesh Neelapu, G. L. Devi, and K. S. Rao. Deep learning based conventional neural network architecture for medical image classification. *Traitement du Signal*, 35:169–182, 2018.
- [66] NIST/SEMATECH. *e-Handbook of Statistical Methods*, June 2003. <https://www.itl.nist.gov/div898/handbook/index.htm>, accessed 10 February 2021.
- [67] Jakub Nowak, Ahmet Taspinar, and Rafał Scherer. Lstm recurrent neural networks for short text and sentiment classification. In Leszek Rutkowski, Marcin Korytkowski, Rafał Scherer, Ryszard Tadeusiewicz, Lotfi A. Zadeh, and Jacek M. Zurada, editors, *Artificial Intelligence and Soft Computing*, pages 553–562, Cham, 2017. Springer International Publishing.
- [68] Thabani Nyoni. Modeling and forecasting inflation in Kenya: Recent insights from ARIMA and GARCH analysis. *Dimorian Review*, 5(6):16–40, 2018.
- [69] United States Department of Agriculture. USDA national agricultural statistics service. https://www.nass.usda.gov/Data_and_Statistics/index.php, accessed 16 November 2020.
- [70] California Institute of Technology. SMAP Specifications. <https://smap.jpl.nasa.gov/observatory/specifications/>, accessed 12 February 2021.
- [71] Christopher Olah. Understanding LSTM Networks, 2015. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>, accessed 25 February 2021.
- [72] Tassia Owen. NASA: About Terra. <https://terra.nasa.gov/about>, accessed 16 March 2021.
- [73] Tapan B. Pathak, Surendra K. Dara, and Andre Biscaro. Evaluating correlations and development of meteorology based yield forecasting model for strawberry. *Advances in Meteorology*, 2016:9525204, Oct 2016.
- [74] Amy Peerlinck, John Sheppard, and Bruce Maxwell. Using deep learning in yield and protein prediction of winter wheat based on fertilization prescriptions in precision agriculture. In *14th International Conference on Precision Agriculture*, 2018.

- [75] Yung-Hsing Peng, Chin-Shun Hsu, and Po-Chuang Huang. Developing crop price forecasting service using open data from taiwan markets. In *2015 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, pages 172–175, 2015.
- [76] Igor Gadelha Pereira, Joris Michel Guerin, Andouglas Gonçalves Silva Júnior, Gabriel Santos Garcia, Prisco Piscitelli, Alessandro Miani, Cosimo Distante, and Luiz Marcos Garcia Gonçalves. Forecasting covid-19 dynamics in brazil: A data driven approach. *International Journal of Environmental Research and Public Health*, 17(14), 2020.
- [77] Joe Phongpreecha. Early corn yields prediction using satellite images, 2018. <https://towardsdatascience.com/early-corn-yields-prediction-using-satellite-images-dcf49b24efab>, accessed 16 November 2020.
- [78] Robi Polikar, Lalita Upda, Satish S Upda, and Vasant Honavar. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)*, 31(4):497–508, 2001.
- [79] Prabhu. Understanding of Convolutional Neural Network (CNN) - Deep Learning, 2018. <https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>, accessed 12 February 2021.
- [80] Xueheng Qiu, Ponnuthurai Nagaratnam Suganthan, and Gehan A. J. Amaratunga. Fusion of multiple indicators with ensemble incremental learning techniques for stock price forecasting. *Journal of Banking and Financial Technology*, 3(1):33–42, April 2019.
- [81] Xueheng Qiu, Ponnuthurai Nagaratnam Suganthan, and Gehan A.J. Amaratunga. Ensemble incremental learning random vector functional link network for short-term electric load forecasting. *Knowledge-Based Systems*, 145:182–196, 2018.
- [82] P. S. Rachana, G. Rashmi, D. Shravani, N. Shruthi, and R. Seema Kousar. Crop price forecasting system using supervised machine learning algorithms. *International Research Journal of Engineering and Technology (IRJET)*, 6:4805–4807, 2019.
- [83] Carl Edward Rasmussen. *Gaussian Processes in Machine Learning*, pages 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.
- [84] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 29:1–98, June 2017.

- [85] Retail Profit Solutions. Where’s My Shrink: Executive Summary. <http://wheresmyshrink.com/executivesummary.html#exectop>, accessed 16 November 2020.
- [86] Mauro Ribeiro, Katarina Grolinger, Hany F. ElYamany, Wilson A. Higashino, and Miriam A.M. Capretz. Transfer learning with seasonal and trend adjustment for cross-building energy forecasting. *Energy and Buildings*, 165:352–363, 2018.
- [87] Ross Fonticella. The Usefulness of the R^2 Statistic. In *Associate of the Casualty Actuarial Society (ACAS) Winter 1998 Forum*, 1998.
- [88] Mark Sabini, Gili Rusak, and Brad Ross. Understanding satellite-imagery based crop yield predictions, 2017. Technical Report. Stanford University. <http://cs231n.stanford.edu/reports/2017/pdfs/555.pdf>, accessed 16 November 2020.
- [89] Lamyaa Sadouk. CNN approaches for time series classification. In *Time Series Analysis-Data, Methods, and Applications*, pages 1–23. IntechOpen, 2019.
- [90] Raí A. Schwalbert, Telmo Amado, Geomar Corassa, Luan Pierre Pott, P.V.Vara Prasad, and Ignacio A. Ciampitti. Satellite-based soybean yield forecast: Integrating machine learning and weather data for improving crop yield prediction in southern Brazil. *Agricultural and Forest Meteorology*, 284:107886, 2020.
- [91] Alireza Sharifi. Yield prediction with machine learning algorithms and satellite images. *Journal of the Science of Food and Agriculture*, 101(3):891–896, 2021.
- [92] Xiangqing Shen, Bing Liu, Yong Zhou, Jiaqi Zhao, and Mingming Liu. Remote sensing image captioning via variational autoencoder and reinforcement learning. *Knowledge-Based Systems*, 203:105920, 2020.
- [93] H. C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. *IEEE Trans Med Imaging*, 35(5):1285–1298, 05 2016.
- [94] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. A Comparison of ARIMA and LSTM in Forecasting Time Series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1394–1401, 2018.
- [95] Statistics Canada. Field Crops in Canada, 2017. <https://www150.statcan.gc.ca/n1/pub/11-627-m/11-627-m2017012-eng.htm>, accessed 14 November 2020.

- [96] Statistics Canada. The Daily: Farm income, 2019, 2020. <https://www150.statcan.gc.ca/n1/daily-quotidien/200526/dq200526a-eng.htm>, accessed 16 November 2020.
- [97] Matthew Stewart. Comprehensive introduction to autoencoders, 2019. <https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368>, accessed 12 March 2021.
- [98] Nima Tajbakhsh, Jae Y. Shin, Suryakanth R. Gurudu, R. Todd Hurst, Christopher B. Kendall, Michael B. Gotway, and Jianming Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312, May 2016.
- [99] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. *CoRR*, abs/1808.01974, 2018.
- [100] William Thong, Hubert Labelle, Jesse Shen, Stefan Parent, and Samuel Kadoury. Stacked auto-encoders for classification of 3d spine models in adolescent idiopathic scoliosis. In *Lecture Notes in Computational Vision and Biomechanics*, volume 20, September 2014.
- [101] UN World Food Programme. 2019 - Hunger Map, 2019. <https://www.wfp.org/publications/2019-hunger-map>, accessed 14 November 2020.
- [102] United Nations General Assembly. Transforming our World: The 2030 Agenda for Sustainable Development, 2015. <https://sustainabledevelopment.un.org/post2015/transformingourworld/publication>, accessed 14 November 2020.
- [103] Thomas van Klompenburg, Ayalew Kassahun, and Cagatay Catal. Crop yield prediction using machine learning: A systematic literature review. *Computers and Electronics in Agriculture*, 177:105709, 2020.
- [104] Anna X. Wang, Caelin Tran, Nikhil Desai, David Lobell, and Stefano Ermon. Deep transfer learning for crop yield prediction with remote sensing data. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies, COMPASS '18*, New York, NY, USA, 2018. Association for Computing Machinery.
- [105] Jin Wang, Liang-Chih Yu, K Robert Lai, and Xuejie Zhang. Dimensional sentiment analysis using a regional CNN-LSTM model. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 225–230, 2016.

- [106] Frank Warmerdam et al. GDAL, n.d. <https://gdal.org/>, accessed 27 January 2021.
- [107] Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for regression. *Advances in Neural Information Processing Systems 8*, 1996.
- [108] C. Willmott and K Matsuura. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate Research*, 30:79, 12 2005.
- [109] Yanzhao Wu, Ling Liu, Juhyun Bae, Ka-Ho Chow, Arun Iyengar, Calton Pu, Wenqi Wei, Lei Yu, and Qi Zhang. Demystifying learning rate policies for high accuracy training of deep neural networks. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1971–1980, 2019.
- [110] Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, Yuxin Peng, and Zheng Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 177–186, 2014.
- [111] Samir S. Yadav and ShivaJirao M. Jadhav. Deep convolutional neural network based medical image classification for disease diagnosis. *Journal of Big Data*, 6(1):113, December 2019.
- [112] Jiaxuan You, Xiaocheng Li, Melvin Low, David Lobell, and Stefano Ermon. Deep Gaussian Process for Crop Yield Prediction Based on Remote Sensing Data. In *Thirty-First AAAI Conference on Artificial Intelligence*, page 4559–4565, 2017.
- [113] Abdelhafid Zeroual, Fouzi Harrou, Abdelkader Dairi, and Ying Sun. Deep learning methods for forecasting covid-19 time-series data: A comparative study. *Chaos, solitons, and fractals*, 140:110121–110121, Nov 2020. 32834633[pmid].
- [114] Jianfeng Zhao, Xia Mao, and Lijiang Chen. Speech emotion recognition using deep 1D & 2D CNN LSTM networks. *Biomedical Signal Processing and Control*, 47:312–323, 2019.
- [115] Peicheng Zhou, Junwei Han, Gong Cheng, and Baochang Zhang. Learning compact and discriminative stacked autoencoder for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(7):4823–4833, 2019.