Versatile Deep Learning Forecasting Application with Metamorphic Quality Assurance

by

Islam Nasr

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, 2023

Author's Declaration

I hereby declare that this thesis consists of material I authored: see Statement of Contribution 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 Contribution

The following publications have resulted from the work presented in the thesis:

1. **I. Nasr**, L. Nassar and F. Karray, "Transfer Learning Framework for Forecasting Fresh Produce Yield and Price," 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 2022, pp. 1-8.

2. **I. Nasr**, L. Nassar and F. Karray, "Enhancing Fresh Produce Yield Forecasting Using Vegetation Indices from Satellite Images," 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Prague, Czech Republic, 2022, pp. 2863-2868.

3. **I. Nasr**, L.Nassar, F.Karray, "Enhanced Deep Learning Satellite-based Model for Yield Forecasting and Quality Assurance Using Metamorphic Testing", 2023 International Joint Conference on Neural Networks (IJCNN), Queensland, Australia, 2023.

4. **I. Nasr**, L. Nassar, and F. Karray. A Study of the Interactive Role of Metamorphic Testing and Machine Learning in the Quality Assurance of a Deep Learning Forecasting Application. International Journal of Information Technology (BJIT), 2023 (in press).

5. Y. Abdelkareem, **I. Nasr**, L. Nassar and F. Karray, "COVID-19 Self-Test Guidance System For Swab Collection Using Deep Learning," 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Prague, Czech Republic, 2022, pp. 2350-2357.

Abstract

Accurate estimates of fresh produce (FP) yields and prices are crucial for having fair bidding prices by retailers along with informed asking prices by farmers, leading to the best prices for customers. To have accurate estimates, the state-of-the-art deep learning (DL) models for forecasting FP yields and prices, including both station-based and satellite-based models, are improved in this thesis by providing a new deep learning model structure. The scope of this work covers forecasting a horizon of 5 weeks ahead for the fresh produce yields and prices. The proposed structure is built using an ensemble of Attention Deep Feedforward Neural Network with Gated Recurrent Units (ADGRU) and Deep Feedforward Neural Network with embedded GRU units (DFNNGRU); (DFNNGRU-ADGRU ENS). The station-based version of the ensemble is trained and tested using as input the soil moisture and temperature parameters retrieved from land stations. This station-based ensemble model is found to outperform the literature model by 24% improvement in the AGM score for yield forecasting and 37.5% for price forecasting.

For the satellite-based model, the best satellite image preprocessing technique must be found to represent the images with less data for efficiency. Therefore, a preprocessing approach based on averaging is proposed and implemented then compared with the literature approach, which is based on histograms, where the proposed approach improves performance by 20%. The proposed Deep Feed Forward Neural Network with Embedded Gated Recurrent Units (DFNNGRU) ensembled with Attention Deep GRUs (ADGRU) is then tested against well-performing models of Stacked-AutoEncoder (SAE) ensembled with Convolution Neural Networks with Long-short term memory (CNNLSTM), where the proposed model is found to outperform the literature model by 12.5%.

In addition, interpolation techniques are used to estimate the missing VIs values due to the low frequency of capturing the satellite images by Landsat. A comparative analysis is conducted to choose the most effective technique, which is found to be Cubic Spline interpolation. The effect of adding the VIs as input parameters on the forecasting performance of the deep learning model is assessed and the most effective VIs are selected. One VI, which is the Normalized Difference Vegetation Index (NDVI), proves to be the most effective index in forecasting yield with an enhancement of 12.5% in AGM score.

A novel transfer learning (TL) framework is proposed for better generalizability. After finding the best DL forecasting model, a TL framework is proposed to enhance that model generalization to other FPs by using FP similarity, clustering, and TL techniques customized to fit the problem in hand. Furthermore, the similarity algorithms found in literature are improved by considering the time series features rather than the absolute

values of their points. In addition, the FPs are clustered using a hierarchical clustering technique utilizing the complete linkage of a dendrogram to automate the process of finding the similarity thresholds and avoid setting them arbitrarily. Finally, the transfer learning is applied by freezing some layers of the proposed ensemble model and fine-tuning the rest leading to significant improvement in AGM compared to the best literature model.

Finally, a forecasting application is implemented to facilitate the use of the proposed models by the end users through a friendly interface. For testing the quality of the application deployed code and models, metamorphic testing is applied to assess the effectiveness of the machine learning models while machine learning is used to automatically detect the main metamorphic relations in the software code. The interactive role played by metamorphic testing and machine learning is investigated through the quality assurance of the forecasting application. The datasets used to train and test the deep learning forecasting models as well as the forecasting models are verified using metamorphic tests and the metamorphic relations in the generalization code are automatically detected using Support Vector Machine (SVM) models. Testing has revealed the unmatched requirements that are fixed to bring forward a valid application with sound data, effective models, and valid generalization code.

Acknowledgements

I would like to thank all the little people who made this thesis possible. I would like to extend my heartfelt gratitude to my supervisor, Prof. Fakhri Karray for his support, encouragement, and expertise have been instrumental in shaping the direction of my research.

I would especially like to thank Dr. Lobna Nassar for her help and guidance with the experiment ideas, documentation, discussions and revisions throughout my master's, her insightful feedback have significantly enhanced the quality of this work.

Lastly, I extend my deepest appreciation to Prof. Narayan, Prof. Ponnambalam, for their invaluable contributions in reviewing this document and providing thoughtful insights.

Dedication

I dedicate this thesis to the most important people in my life, whose unwavering love, support, and encouragement have been my constant motivation:

To my loving parents, your guidance, sacrifices and unconditional love have shaped me into the person I am today. This thesis is a tribute to your love, dedication, and unwavering support as my parents.

To my beloved wife, your unwavering love, patience, and understanding have been my anchor throughout this journey. Your presence by my side and your belief in me have given me the strength to overcome challenges and pursue my dreams.

To my dear brothers, you have been my companions, confidants, and pillars of strength. Your encouragement, guidance, and belief in my capabilities have motivated me to strive for excellence.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fresh produce prices are increasing drastically, where it is reported that for 2023, an increase of 10-13% is encountered in the prices of fresh produce in Canada [1], although a regular inflation is encountered all over the world, many stores are increasing the prices with rates higher than the inflation. The main factor that should affect the fresh produce price level is the yield level. Hence, having an accurate forecasting model for fresh produce yield and price leads to fair market prices for consumers. It does not only protect the farmers from the devaluation of their produce by retailers but also helps retailers in having informed bidding prices. Hence, finding a highly accurate and reliable forecasting framework with consistent quality is an essential task.

1.1 Motivation

Throughout the years, researchers have been tackling the forecasting of crop parameters as yield and price using a variety of approaches. Previous research is made in the domain of crop yield and price forecasting using soil and weather parameters retrieved from station recorded data and satellite data [2]. The use of vegetation indices is also highlighted by multiple researchers to be effective in agricultural fields for crop yield detection and prediction [3, 4]. Using the soil and weather parameters provides good insights on their own, same as using the vegetation indices alone, for yield predictions. However, combining both types of parameters has been neglected despite its potential for improving the forecasts.

Additionally, multiple authors, such as in [5, 6], have proven that applying transfer learning to deep learning models provides improved results and faster training over training

a new model especially when not enough data instances are available. There are two prerequisites for effective transfer learning: having a complete dataset to train the base model and a good similarity module to assure that the transfer learning is applied among similar datasets; the base and target datasets. However, the scarcity of complete datasets increases the need for a good imputation technique. Finally, all machine learning (ML) models are gauged by error scores that are dataset dependent. Therefore, having more general standards as guidelines to assess the soundness of the model and the datasets used to create it is essential for its quality assurance.

1.2 Challenges

The main target of this work is to provide an application that provides highly effective forecasting of fresh produce yield and price with transfer learning techniques to enable the generalization of forecasting to unknown fresh produce provided by the application users. The first encountered challenge is the lack of datasets for fresh produce, where the availability of daily datasets is very limited. The second challenge faced is designing a model that is capable of forecasting the output parameters with competing scores, in addition to being able to apply transfer learning to unknown fresh produce to the model in a way that enables the transfer learning model to perform better than a model trained solely on the second fresh produce, where the challenge encountered in the transfer learning is not only in the application of transfer learning but also in finding a similarity method that could determine similar fresh produce time series effectively. The last challenge is to ensure the quality of the developed model, this is since machine learning techniques quality is not thoroughly tested; it is evaluated using just test scores.

1.3 Proposed Solution

Different deep learning models are proposed in literature to tackle the fresh produce attributes forecasting using both station-based parameters and satellite-based parameters [2, 5, 7, 8]. Additionally, transfer learning has gained a lot of interest in this domain by many researchers who apply it to either save learning time or to improve model results [5, 9].

The work in this thesis focuses on improving deep learning forecasting models to enhance the performance of the literature models in [5, 7]. First, the data preprocessing is improved by introducing an imputation model and comparing it with literature imputation

models in [11] to find the best model for enhancing the imputation effectiveness. Second, a new forecasting model is proposed with both station and satellite-based models where satellite vegetation indices are utilized to improve the forecasting.

For the generalization, a similarity module is proposed to enable transfer learning which is essential for efficiently getting forecasting models for new crops. Finally, a quality assurance framework is proposed to assure that the quality of the proposed model is not only assessed using defined data, but also assured based on more comprehensive metamorphic tests. Those tests assure the presence of metamorphic relations that act as high-performance constraints independent of the test data.

Finally, an application combining all proposed models is implemented to enable utilization of the proposed solution by end users.

1.4 Contributions

The main contributions of this thesis are summarized as follows:

- A deep learning model architecture is proposed; DFNN-ADGRU ENS, for forecasting fresh produce yield and price with enhanced results on previous literature model [7].
- The forecasting using satellite images is enhanced by:
  1. Adding the NDVI input parameter.
  2. Applying an ensemble imputation method for missing data with ¿50% missing, consisting of an ensemble method proposed in [11] and linear interpolation technique based on the nature of the missing data for the NDVI.
  3. Reducing the overhead costs of satellite images preprocessing by using average preprocessing rather than histograms.
- Contributions for the generalization of the model:
  1. A clustering module for time series by using clustering techniques and similarity between time series to group together similar time series.
  2. Proposing a transfer learning framework [10] which consists of the clustering module with the transfer learning model structure to apply transfer learning to fresh produce time series.

- Quality assurance module is proposed with metamorphic relations and metamorphic tests that are applied to assure the quality of proposed deep learning model.
- An application with similarity, forecasting and imputation modules is designed, implemented and tested.

1.5 Organization of the Document

The rest of this thesis is organized as follows: Chapter 2 covers the background and literature review for: datasets preprocessing and imputation, deep learning models and the quality assurance. In Chapter 3, the proposed solution is presented. In Chapter 4, the conducted experiments are illustrated with the reported results and their analysis. In Chapter 5, the application is presented with sample results. Lastly, in Chapter 6 the conclusion is provided along with the planned future research directions.

Chapter 2

Background and Literature Review

In this chapter, the background on the datasets utilized in this work is provided in Section 2.1: the tabular data is covered in subsection 2.1.1 while the retrieved satellite data is illustrated in subsection 2.1.2. In Section 2.2, an overview of the literature data preprocessing techniques is presented including imputation methods in subsection and the data preprocessing methods for the data fed into the deep learning models in subsection 2.2.2. Deep Learning is discussed in Section 2.3; well-established artificial neural network layers and literature models of time series forecasting are elaborated in subsection 2.3.1, and subsection 2.3.2 describes the evaluation metrics used in the performance evaluation of the deep learning models. The transfer learning topic is discussed in Section 2.4 where the similarity techniques are covered in subsection 2.4.1 and the transfer learning etablished frameworks are discussed in subsection 2.4.2. Finally, the literature machine learning quality assurance models are explored in Section 2.5.

2.1 Datasets

This section will cover the main sources of data used throughout the thesis, the two main sources of data are tabular data for station-based forecasting discussed in subsection 2.1.1, and satellite images for satellite-based forecasting discussed in subsection 2.1.2 with an overview of different vegetation indices that can be retrieved from satellite images.

2.1.1 Tabular Station Based Data

The utilized datasets contain the data for the input parameters as weather and soil datasets along with their corresponding output datasets such as the fresh produce yields and prices. The data for California, Santa Maria soil and weather parameters is downloaded from the National Oceanic and Atmospheric Administration site [12], whereas the data for yields and prices of the three considered fresh produce, which are strawberry, blueberry, and raspberry, is obtained from California Strawberry Commission Website [13].

Due to the scarcity of real datasets and the need for a higher number of datasets to evaluate different aspects of the system, additional yield datasets are synthesized using a weighted average of the yields of each pair of the three FP real datasets. The pairs of weights for the synthesized datasets range from (0.1,0.9) to (0.9,0.1) with (+0.1, -0.1) change when synthesizing a new dataset such that both weights add up to 1. Three unordered pairs with unique nonidentical elements can result from using the 3 real FPs. Further, having 9 possible combinations of each distinct pair results in an additional 27 synthesized datasets hence a total of 30 datasets with the real data. The synthesized datasets are named by the initials of their pairs of fresh produce, FP_a and FP_b, used in generating them followed by the weight of the FP_a and a letter showing their binary similarity where S means that FP_b is similar to FP_a and D if dissimilar; no need to mention the weight of FP_b since it can be implicitly deduced by subtracting the FP_a weight from 1. For example, a synthesized dataset $SB.7S$ is generated using the weighted average of strawberry and blueberry yields with weights (0.7,0.3) and the binary similarity shows that they are similar, and $RB.6D$ is generated using the weighted average of raspberry and blueberry yields with weights (0.6, 0.4) and the binary similarity shows that they are dissimilar.

2.1.2 Satellite Based Data

Satellite Images: The use of satellite data has been emerging in many fields, this is due to having numerous satellites covering vast geographic areas and the availability of this data for free. The data from satellites is used in fields of earth observations such as predicting rains and earthquakes [14] [15], in addition to many other fields including poverty and vegetation cover predictions [16] [17]. The global coverage of satellite images makes them useful for extracting information which can be utilized in many applications such as using surface reflectance in forecasting fresh produce yield [18]. Many satellites are available for researchers who can freely access them using Google Earth Engine (GEE) [19]. The most frequently used satellites in the previously mentioned domains are Moderate Resolution

Imaging Spectroradiometer or MODIS, Landsat, Sentinel, and NASA_USDA. The criteria for selecting the most effective satellite mainly depend on the spatial resolution of the satellite, the availability of images during the required date intervals and the frequency of taking images.

MODIS satellite provides a spatial resolution of 250m, 500m and 1km [19], which makes each pixel in the image carry many details, hence it is the best for pixel-based tasks as in [18]. The main advantage of MODIS satellite is its high frequency in taking images compared to the other satellites; images are taken every three days for the same location. On the other hand, Landsat satellites provide images with spatial resolution of 30m which are taken every eight days for the same location [20]. There are two versions of Sentinel satellites, Sentinel-1 and 2, Sentinel-2 satellite is more popular for having images captured in recent dates with better spatial resolution than Sentinel-1 [19] hence, it is more suitable for applications where old data is not needed. In addition, Sentinel-2 provides images with the lowest spatial resolution of 10-20m; the images are taken every 7 days for the same location [20]. NASA_USDA provides a daily image of global soil moisture with spatial resolution of 10 km coverage. Landsat has many different versions of satellites ranging from Landsat-1 to Landsat-9, where the most relevant satellites to the dates under consideration are Landsat-7 launched in 1999 and Landsat-8 launched in 2013. Lastly, Sentinel satellite has two identical satellites which are Sentinel-2B and Sentinel-2A launched in March 2017 and June 2015 respectively [21]. Figure 2.1 shows images retrieved from Landsat-8, Sentinel-2B, and Google Earth satellites used during the retrieval of NDVI.

The remarkable contribution of using parameters retrieved from satellite data in fresh produce yield forecasting is highlighted by J. You et al. in [22] where annual soybean yields are predicted using satellite images of surface reflectance and temperature. In [18], the authors utilize the Soil Moisture and Soil Temperature, from NASA_USDA and MODIS Satellites in [23] and [24], as input parameters to deep learning yield forecasting models by obtaining the average of Surface and Subsurface Soil moisture, and the average of Day and Night Temperatures.

Vegetation Indices: One of the applications where using satellite data proved to be most effective is the agriculture field of remote sensing which provides useful insights for vegetations. The vegetation index is a spatial transformation of two or more bands to enhance the contribution of vegetation properties retrieved from satellite images [26]. The vegetation indices maximize the sensitivity to vegetation characteristics retrieved [25]. There are six main vegetation indices used in literature for crop yield prediction and detection [26, 27]:

1. Enhanced Vegetation Index (EVI) which is used in vegetation monitoring, vegetation estimation and health of crop detection. It is calculated using the Near Infrared

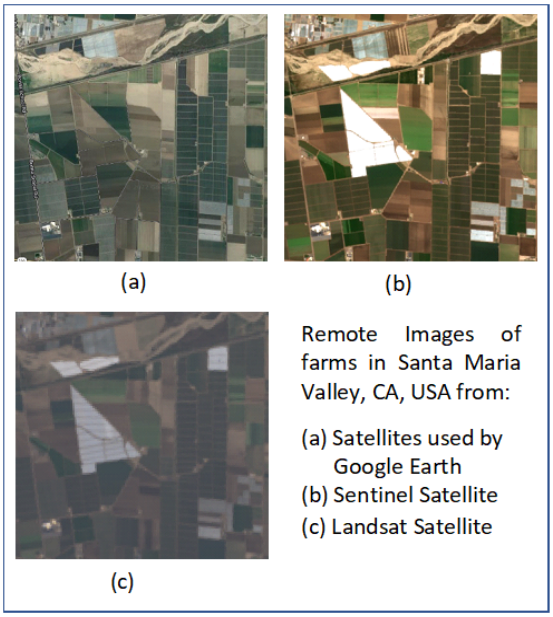

Figure 2.1: Images samples for farms in Santa Maria Valley from three Satellites

(NIR), red, and blue bands [28] as in (2.1).

$$EVI = 2.5 * \frac{NIR - RED}{NIR + 6 * RED - 7.5 * BLUE + 1} \tag{2.1}$$

2. Normalized Difference Phenology Index (NDPI) which is used for removing the impact of snow and bare soil, by replacing the red band with the addition of the Short-Wave InfraRed (SWIR) to the red band and reducing the vegetation indices model saturation to the vegetation biomass which increases precision in the high biomass area [28]. NDPI is calculated as in (2.2).

$$NDPI = \frac{NIR - (0.74 * RED + 0.26 * SWIR1)}{NIR + (0.74 * RED + 0.26 * SWIR1)} \tag{2.2}$$

3. Normalized Difference Vegetation Index (NDVI) which has the same usage of EVI in vegetation monitoring, estimation of crop health and detection of crops. NDVI differs from EVI by removing the impact of the blue bands [28]. NDVI is calculated as in (2.3).

$$NDVI = \frac{NIR - RED}{NIR + RED} \tag{2.3}$$

4. Soil Adjusted Vegetation Index (SAVI) is used in minimizing the influence of spectral vegetation indices involving red and near infrared bands [28]. It is calculated as in (2.4).

$$SAVI = 1.5 * \frac{NIR - RED}{NIR + RED + 0.5} \tag{2.4}$$

5. Normalized Difference Water Index (NDWI) is used in monitoring the change in the water content in the plants leaves [28]. NDWI is calculated as in (2.5).

$$NDWI = \frac{NIR - SWIR1}{NIR + SWIR1} \tag{2.5}$$

   There are two short wave infrared (SWIR) bands denoted as SWIR1 and SWIR2. SWIR bands can penetrate thin clouds and smoke more than visible bands, SWIR1 helps discriminate between dry and wet soils whereas SWIR2 is used for getting the geological features of the soil and determining its minerals [29].

6. Tillage Index (NDTI) is used in distinguishing the non-photosynthesis vegetation biomass from the green vegetation biomass [14]. NDTI is calculated as in (2.6).

$$NDTI = \frac{SWIR1 - SWIR2}{SWIR1 + SWIR2} \tag{2.6}$$

All vegetation indices mentioned above can be calculated using satellite bands from the three satellites mentioned previously the criteria for selecting the most appropriate satellite to use for calculating these bands mainly depends on the required spatial resolution, availability of images in the required dates and frequency of taking the images.

2.2 Preprocessing and Data Imputation

The data used in developing machine learning models are usually preprocessed as the raw data might not be suitable for direct utilization. Therefore, techniques for estimating data samples are necessary in case of having missing samples. The major techniques for dealing with missing data are discussed in subsection 2.2.1, which are categorized into two types: the statistical interpolation approaches and deep learning imputation approaches. In addition, an overview of the preprocessing techniques for station-based and satellite-based data is given in subsection 2.2.2.

2.2.1 Missing Data Imputation

This section will give an overview of the statistical data interpolation techniques, in addition to the established deep learning approaches for missing data imputation.

Statistical Interpolation Approaches: The main goal of the statistical procedures applied on datasets is making an efficient and valid inference, about the population of interest, using less samples. Recovering missing values is tackled by many statistical techniques ranging from simple filling methods, such as those estimating the missing values with the mean of the available ones or replicating the previously available value right before the missing set until the first successive value appears, to more complex and effective imputation techniques, such as those deploying deep learning techniques. Authors in many domains provide techniques for imputing the time series missing values using statistical models [30] [31], while others have proposed deep learning models for more accurate imputation especially in cases where there are missing chunks of successive data [32]. Despite the effective performance of the deep learning techniques, they are not as efficient as the simple ones. The deep learning model has higher creation and operating costs, which might not be justified especially if the missing values are randomly distributed rather than successively occurring in the form of chunks. Therefore, in this section a brief overview is provided for four simple interpolation techniques which are the Linear, Bessel, Cubic Spline, and One-way Spline Interpolation techniques [33].

1. Linear Interpolation: Finding values using the linear interpolation technique is based on using linear polynomials to build new points in an interval containing missing values among known data points [34]. As in [35], linear interpolation is calculated by finding f(x) which is the value of the dependent variable, as in (2.7).

$$f(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0} * (x - x_0) \tag{2.7}$$

   Where x0 and x1 are known values of the independent variable x. Linear interpolation is preferred for its simplicity and speed. However, despite its acceptable accuracy when used with closely spaced datasets [36], it is the least accurate among the discussed interpolation methods [36].

2. Bessel Interpolation: Interpolation of missing values using the Bessel method is based on the Fourier Transform, specifically focusing on retrieving time-series periodogram, spectral density estimate [37], using the moving window of Kaiser-Bessel [24]. It is calculated as in (2.8) and (2.9).

$$p = \frac{x - x_0}{h} \tag{2.8}$$

$$\frac{y_0+y_1}{2} + (p-0.5) * \delta(y_0) + \frac{p(p-1)}{2!} * \frac{\delta(y-1)^2+\delta(y_0)^2}{2} + ... \quad (2.9)$$

The main advantage of using Bessel interpolation is that it usually offers better estimates tending to be closer to the points of the time series available and dampening the effect periodic gap affecting the interpolation methods [38]. Despite its suitability for non-linear trends, it requires a lot of computational power [39].

3. Cubic Spline Interpolation: The Cubic Spline interpolation technique is a piecewise polynomial which reduces to a cubic polynomial in each subinterval. Cubic Spline consists of two continuous derivatives, where at every interior point the function values, first and second derivatives all fit together [40]. The Cubic Spline (S(x)) is calculated as in (2.10) and by satisfying the following conditions:

   - S(x) should be a subset of C2[a,b] (C is a cubic function, and a,b are two different datapoints.)
   - On each sub-interval [x_{i-1},x_i], S(x) is polynomial of degree 3, where i=1,2,..,n and C is a cubic function.
   - y_i is the interpolated feature over interval x_i

   $$S(x_i) = \ y_i i \ \forall i \quad (2.10)$$

   Although the conditions seem complex, Cubic Spline is preferred for its calculation simplicity, numerical stability, and smoothness of the interpolated curve, however it has a drawback of being sensitive to the accuracy of the given data [41].

4. One-way Spline Interpolation: One-way Spline is a more constrained version of the Bessel interpolation, if the source data is monotonic, only increasing or decreasing, it will always produce monotonic results. However, if the source data is non-monotonic the results won't necessarily be monotonic, yet it always produces a well-constraint curve with the fewest possible overshoots and oscillations [42]. Its main advantage is overcoming many oscillations between datapoints hence successfully capturing the overall signal shape.

Deep Learning Imputation Approaches: Conventional nondeep machine learning approaches are ineffective in capturing the temporal variations in multivariate time series [43]. Hence, utilizing deep learning (DL) models for missing data imputation has gained vast research attention in the past few years. Over time the DL imputation models went from simple DL models such as RNN in [44, 45] to ones with customized compound DL models such as those in [46, 47].

Missing data falls into one of three categories [48]:

1. Missing Completely at Random (MCAR) which shows that a missing value in a feature is unaffected by any other features in the dataset, where the reason for missing has nothing to do with the data values.

2. Missing at Random (MAR) which explains the event when a value is missing due to other values of other features.

3. Missing Not at Random (MNAR) explains the event where there is a relationship between the propensity of a value to be missing and its values.

DL models proposed in [11] were developed to handle two types of missing, MCAR and MAR. Where MAR is usually handled due to the needed task of daily-available datasets, and the dataset sources are not as frequent as the needed data.

Authors in [11] proposed two techniques to recover missing data. The first technique is more suited for recovering data that is randomly missing, where a Voting Regressor Ensemble is applied to ensemble the top two performing DL imputation models found: LSTM-Deep GRU and Residual GRU, by averaging the regression results of those component models. A recommendation framework of the top two performing DL imputation models to be ensembled for various time series types is found in [11].

The second technique is more suited to recover chunks of missing data. Chunk missing data is usually addressed as MNAR, as the missing of a huge chunk of the data would have a factor for missingness which is not directly captured by the data. Hence, a transfer learning model proposed in [11]; built of 4-layers of LSTM, is used where the knowledge from the base model is transferred to impute the missing values in the target dataset using the target model [52].

A hybrid model combining the chunks and random method as proposed in [11] is utilized by combining the Ensemble and Transfer Learning models where it classifies the missing values into either missing random or chunks; the Ensemble model is used for the random missing imputation and the Transfer Learning for the chunks.

2.2.2 Dataset Preprocessing

In this section, the preprocessing applied to the datasets in Section 2.1 is discussed for both station-based and satellite-based data.

Station-Based Data Preprocessing: For the output parameters, due to the missing values in the FP yield and price datasets, imputation is applied using the deep learning

technique proposed in [11]. For the input parameters, it is noticed that the utilized dataset in subsection 2.1.1. contains a high number of parameters hence it is essential to extract the most effective ones for efficiency. Yet, deciding the most effective parameters is a challenging task therefore a Random Forest regressor is trained and feature importance is extracted to automate the feature selection [53]. The soil moisture and temperature are found to have the highest influence on the yield. On the other hand, the importance of considering the date as an input parameter in seasonal time series forecasting is discussed in [54]. Hence, the Day of Year (DoY), in the range [1-365], is extracted from the dates, then two cyclic features are produced by calculating the sine and cosine of the DoY. The calculated features are added to the dataset and the feature selection method is reapplied. It is found that the calculated cyclic features highly influence the yield. Therefore, it is decided to consider the calculated cyclic features along with the soil moisture and temperature as input parameters to the DL models. A lag of 20 weeks of soil moisture and temperature is used to forecast 5 weeks ahead yield and price; this lag is recommended in [7]. This increases the total number of the soil moisture and temperature parameters to 280, therefore a Principal Component Analysis (PCA) is applied to obtain the minimum parameter set with the maximum proportion of variance, which is 92%, that resulted in a set 34 parameters. After adding the two cyclic parameters, the sine and cosine of the DoY, to that set a total of 36 parameters are considered as input. The dataset is then divided into train and test sets, the first 80% of the time series are used for training and the last 20% for testing.

Satellite Based Data Preprocessing: The input and output parameters are extracted from two data sources from 2011 till 2019 covering Santa Maria, California: The first data source is the satellite-based input data of soil parameters, with soil surface and subsurface moisture parameters retrieved from MODIS Soil Moisture satellite [19] with frequency of one reading every 3 days, and soil day and night temperature retrieved from MODIS Land Surface Temperature satellite [18] with daily frequency. A mask of the relevant area for satellite data is attained using the coordinates of the required county and applied. Two main approaches for satellite data preprocessing are investigated for handling the input datasets and their effectiveness is compared. A new parameter, retrieved from Landsat satellites is preprocessed, and added to the other preprocessed parameters. This parameter is called Normalized Difference Vegetation Index (NDVI) which is retrieved weekly or biweekly; hence it is imputed to have daily values. The second data source is the tabular yield output values, containing the daily values of strawberry yield, detailed in subsection 2.1.1. An imputation technique is applied to handle the randomly missing data in all retrieved datasets.

1. Preprocessing using Histograms: Due to the huge size of images, an approach is

utilized based on the histogram approach introduced in [5] and applied for the models. The frequency counts of pixel values are considered to represent the images assuming that the images hold the property of permutation invariance, where the location of the pixel does not contain relevant information. Calculated histograms are then normalized and combined in a 3D matrix, with dimensions of (time window, bands, bin) which are set to (140, 4, 32); 140 is the best forecasting window as found in [2], 4 is the number of features and 32 is the count of bins.

2. Preprocessing using Averaging: Another preprocessing approach is proposed where each satellite image is represented with a single daily average of the parameter image pixels. This value is found by: First, averaging all pixels in each daily satellite image of the parameter; the image covers the region of interest. Second, rescaling and using the average value as a representative of that parameter on one single day [55]. This approach is considered due to the huge processing and memory power needed by the first approach of histograms.

2.3 Deep Learning

In this section, the major types of neural networks used in time series forecasting are discussed and plotted in subsection 2.3.1. In addition to that, the major evaluation metrics used in forecasting the models for time series forecasting is recorded in subsection 2.3.2.

2.3.1 Artificial Neural Networks

Deep learning is deployed in the field of time series forecasting as it provides prominent results compared to those achieved by the traditional machine learning algorithms [6]. The deep learning models try to resemble the behavior of neurons in the human brain by using deep neural networks [56]. The main type of artificial neural networks used for handling data sequences to recognize patterns related to spatial information is the recurrent neural networks. The three main recurrent neural network subtypes are Long-Short-Term Memory (LSTMs), Recurrent Neural Network (RNNs), and Gated Recurrent Units (GRUs). RNNs are used to remember information from previous inputs, however they usually struggle with long-term dependencies [6]. The LSTM units are useful for capturing long-term dependencies with a forget gate to decide when to stop remembering past inputs. The GRU units are proposed to improve on the LSTM cells by having simpler

structure with equal immunity to the vanishing gradient [6]. The structures of the recurrent neural network subtypes are visible in Figure 2.2 .

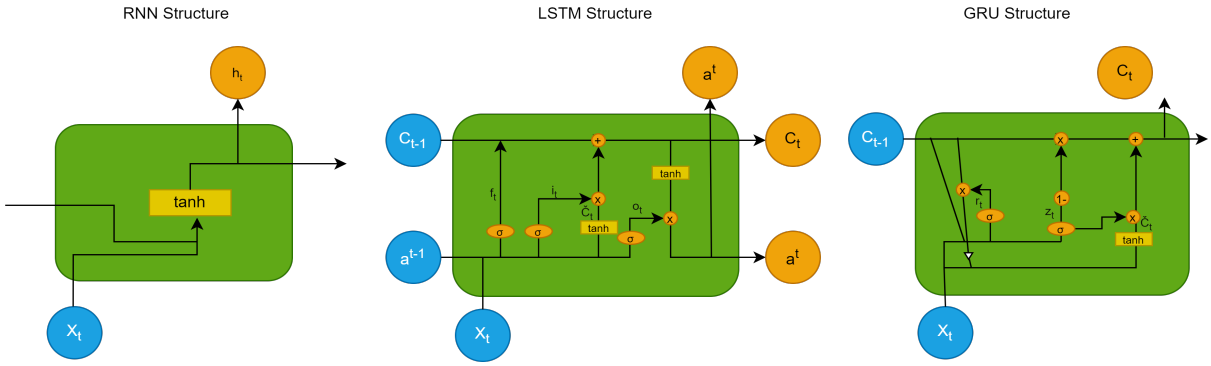

Figure 2.2: Recurrent Neural Network Subtypes Structure

Using simple models of recurrent neural networks is not proven to be very effective in time series forecasting. Hence, more complex models are deployed to achieve higher effectiveness. In [2], the authors used a voting ensemble; which is simply averaging results of multiple models, to ensemble a CNN-LSTM model, a combination of convolutional neural networks and long short-term memory units, with a SeriesNet-GRU model, which is a combination of SeriesNet and Gated Recurrent Units. The resulting ensemble is used to forecast fresh produce yield and price. In [7], deep learning models are deployed to forecast the houses electricity load in a residential area; they proposed a novel DNN-CAE 2 model to enhance the work in [57]. A review of the proposed deep learning models for electricity forecasting is presented in [58] where the authors described simple DL models, such as RNNs, DFFNNs, LSTMs, GRUs, along with compound ones, such as LSTM-dense, LSTM-FFNN, and GRU-ANN. The Deep Feedforward Neural Networks (DFFNN) are normal feed forward networks which are composed of dense layers; the reason for naming it deep is that they contain more than 3 dense layers. In addition, the attention layers have gained the interest of researchers because they make the model focus on important features in the input signals as illustrated in [59].

2.3.2 Evaluation Metrics

Four evaluation metrics are used in analyzing the performance of the deployed DL models: First, the Mean Absolute Error (MAE) which is the mean of individual prediction errors

over all instances of the test set [60]. Second, the Root Mean Squared Error (RMSE); which is the mean of squared prediction errors over all instances in the test set [60]. Third, the R-Squared Score (R^2) which represents the percentage of variation in the dependent variable explained by variation in the independent variables [61]. Lastly, the Aggregated Measure (AGM) combines all previous methods by averaging the two error measures, RMSE and MAE, then scaling them by (1-R^2) [62].

2.4 Transfer Learning

This section covers the importance of transfer learning techniques, where the application of transfer learning requires similarity techniques to categorize whether a dataset is suitable for applying transfer learning or not. Therefore, similarity techniques are discussed in subsection 2.4.1 followed by the established transfer learning techniques in subsection 2.4.2.

2.4.1 Similarity and Clustering Techniques

Finding the similarity among time series is an essential task for applying transfer learning. Many researchers tackled the similarity task in different ways. In [63], the authors divided the utilized distance measures into four categories: shape-based, edit-based, feature-based, and structure-based. For the shape-based measure of the derivative-dynamic time warping (D-DTW) the distances between first derivative of the time series are calculated. The authors proved that the D-DTW outperforms the other three measures. However, calculating the first derivative of daily changing time-series fails to give good intuition of the signals especially with fluctuating time series. In [64], the dynamic multi-perspective similarity measure is proposed based on giving weights, such that more recent dates have higher weights, then measuring DTW with embedded Canberra distance. The proposed measure is computationally expensive, and weighting based on closer dates ignores long term dependencies. Moreover, the similarity measure in [65] is based on local extrema by computing DTW distances between extremums of different time-series which gives huge enhancement in terms of computing power. Yet, this measure would be more applicable to smooth time series which are not fluctuating daily. The authors in [66] proposed an adaptive feature-based dynamic time warping (AFDBTW) which is based on aligning two sequences based on their points' local and global features rather than values or derivatives. This method is found to outperform both classical and derivative methods. The local features at point an are based on its predecessor and successor points; a_{n-1} and a_{n+1}. Conversely, the global

features are based on all preceding and succeeding values to a_n; $a_{1\,to\,n-1}$ and $a_{n+1\,to\,m}$ where m is the time series length.

In [67], the authors proposed a framework for determining the similarities between pairs of fresh produce. The authors proposed a method for finding the similarity based on output parameters and another based on input parameters. By utilizing time series decomposition, the proposed framework achieved good results with the tested datasets. However, a major drawback of the proposed framework is setting the thresholds arbitrarily based on trial and error and giving lower similarity percentage based on the difference in lag time; for each one-month lag, the similarity percentage of one feature drops by 5%. This would affect the results while the main target is to have time series with similar behavior rather than having a penalty for the lag difference. In [68], transfer learning is applied to find electrical consumption forecasting models for houses with similar consumption time series. The authors used a combination of Euclidean, Cosine and DTW distances to decide the similarity among those electrical consumption time series. Their main concern is how to find the house with electrical consumption that is most similar to all surrounding houses, which is the centermost house, to apply transfer learning from its forecasting Base model to get Target models for all surrounding houses. That centermost house is selected based on the smallest cumulative distance to other houses.

The discussed similarity methods are usually utilized in grouping data into clusters, which are groups of similar elements. In [68], only one centermost house is found but a similar approach could be applied if multiple centers should be identified through clustering techniques. Clustering techniques are divided into hierarchical, partitional, grid, density based and model-based techniques [69]. In the hierarchical techniques, the clusters are formed by iteratively dividing or joining clusters together until reaching the one single cluster or element. The hierarchical techniques are further divided into 3 groups. The single linkage clustering, where the distance between clusters A and B is decided by the minimum distance between any two elements a and b where $a \in A$ and $b \in B$. Conversely, the complete linkage clustering considers the maximum distance between any two elements a and b where $a \in A$ and $b \in B$. Finally, the average linkage, where the average distance is considered instead. Partitioning clustering is assigning data into clusters, without a hierarchical structure, using a distance measure; the most common example is the k-means clustering using the Euclidean distance. The data is classified into k-random non-deterministic clusters with means that change every iteration based on the initial centers and the chosen k. A more deterministic clustering algorithm is proposed in [70] and improved in [71] which is the K-Medoids. Medoid means that the chosen center is a representative object from the cluster rather than a calculated mean of its points as in the K-means algorithm; for the same data with same K, the same representative objects can

be found. The K-medoids can be preferred over the K-means if the user needs to deploy a distance measure other than the Euclidean distance. However, the number of clusters (K), still can't be automatically decided; hence K needs to be arbitrarily decided before clustering.

2.4.2 Established Frameworks

The deep learning techniques require a huge set of training and testing datasets which require expensive resources and incur high computational costs [9]. Hence, transfer learning became a popular topic in time-series forecasting due to the experiments conducted on its effect in [72]. It improved the results in many computers vision tasks as in [73], natural language processing (NLP) tasks with the most common example of Bidirectional Encoder Representations from Transformers (BERT) [74], forecasting time series of power, electricity and crop yields and prices [2], [68], [9], as well as imputation tasks [75]. Generally, in transfer learning a Base model is trained to learn the features of a large dataset, then the same model is retrained or finetuned using a Target dataset to perform a similar or new task [76]. Based on [76], transfer learning can be divided into two main categories: Data and Model based. In the Model based category, the most used technique is a Parameter Control Strategy which is Parameter sharing. Parameter sharing is implemented either by sharing layers through freezing some while finetuning others, or finetuning all the layers [76]. Transfer learning is mainly improving the results due to having less dependency on data and labels, but it usually suffers from negative transfer performance deterioration of new model or overfitting.

Frameworks are proposed with most of the transfer learning applications, one example in image processing and object detection is GAIA [73]. GAIA is used in object detection where new images are taken as input with powerful pretraining weights based on huge datasets fitting many fields, and the model is then retrained on the new images. Another framework is proposed in [74] for BERT, where BERT model is trained on unlabeled data, then the model is fine-tuned using labeled data for an NLP task. An imputation framework using transfer learning is proposed in [75] where a Base model is trained with the most similar sequence to the missing data, then transfer learning is applied using fine-tuning to calculate the missing values. In time series forecasting, transfer learning is applied in a variety of ways: In [9], fine-tuning of the Base models, of multiple time series, is applied to get the Target models. In [68], the centermost home is found, and a Base model is created to forecast its power consumption then the model is fine-tuned to get the Target models for forecasting the surrounding houses' consumption. In [2], a fresh produce is chosen to train the Base model. The Target model is then found using transfer learning from that

Base model by freezing some of its layers and retraining the rest with less epochs using similar fresh produce time series.

2.5 Quality Assurance

The verification of any piece of code requires comparing the code's expected and actual outputs [77]. This type of verification is challenging, incomprehensive and sometimes impossible, hence metamorphic testing (MT) is introduced [78]. Metamorphic testing uses sets of relations between the input and output that must be satisfied, these relations are referred to as Metamorphic Relations (MRs) as in [79] and [80]. Quality assurance using MRs is tested in several domains including testing machine Learning models [81, 82, 83], and in testing regular software code in [84]. The testing of ML models is usually done by a left alone dataset called the test set, which tests the model's performance level using performance measures [85]. A main key challenge facing ML models is the absence of test oracles as pointed out in [86]. Hence, this necessitates the use of MRs for assuring the quality of the designed ML models.

MRs are the generalization of any relation type; equalities, inequalities, periodicity, convergence, and many other relations, in general metamorphic relations represent the concepts of program invariants. However, the main difference between invariants and metamorphic relations is that invariants must hold for every execution, but the MR is the relation holding between different executions [77].

The literature work tackling quality assurance of machine learning algorithms using metamorphic testing is covered in subsection 2.4.1. In subsection 2.4.2, the role machine learning algorithms play in predicting metamorphic relations in software code is discussed along with its importance for quality assurance.

2.5.1 Metamorphic Testing of Machine Learning

In [86], authors pointed out three key challenges faced in testing machine learning systems: The absence of test oracles that provide correctness degree of the model's output. The large input space of machine learning models, where a high volume of diverse application field data is fed into the model. Finally, the high cost of white-box testing. For the absence of test oracles and cost of white-box testing: the metamorphic testing eliminates the need for such test oracles in [81, 82, 83] by finding suitable metamorphic relations that must

hold in the tested machine learning models and assuring that those relations are fulfilled by those models.

The authors in [78] utilized metamorphic relations and tested it using metamorphic testing. One of their metamorphic relations states that transforming the model's input should not affect the output, which is verified using a metamorphic test that starts by transforming the data input D using a transformation function T then comparing the results of the trained models' output, where one model is trained on D and the other on T(D). It is found that both models give the same output for input X and T(X). On the other hand, an efficient white-box technique with test constraints is introduced by the authors to achieve code coverage, where they developed test cases to verify the model by using prior domain knowledge and analyzing the problem. Finally, the authors compared the results of those test cases with the results of running the developed metamorphic tests where they found that metamorphic testing reveals more flaws in the tested model than the white-box test cases. Hence, it is found that the metamorphic testing is an effective verification technique in the machine learning field.

2.5.2 Finding Metamorphic Relations using Machine Learning

Despite the high effectiveness of using metamorphic testing for machine learning models, automating the software testing process is important to eliminate the need for domain experts to evaluate each piece of software code. In addition, deciding the metamorphic relations that need to be tested can be an expensive process. Therefore, researchers have developed multiple techniques that utilize the control flow graphs (CFG), which are first introduced in [87], and applied methods for extracting features from those graphs. Those features are labelled with the main metamorphic relations and used to train machine learning models which are then used to automatically identify whether those relations hold or not in any unseen code. In [88], the authors utilized a random walk algorithm to extract the features from node features. Authors in [89] proposed a similar approach for extracting features from the CFG nodes by explicitly extracting node features and adding path features which maintain the code sequence. In [80], the authors automated the extraction of the features by generating a directed CFG and applied a graph kernel algorithm to find the similarity between graphs. The output of the algorithm is then normalized and passed as features to a support vector machine (SVM) classifiers with proper labels for the metamorphic relations.

Chapter 3

The Proposed Solution

In this section, the details of the main aspects of the proposed solution, for handling the task of fresh produce forecasting, are presented. First, the deployed data imputation model is discussed in Section 3.1. The proposed deep learning forecasting model, which is deployed to enhance the fresh produce yield forecasting, is explained in Section 3.2. The proposed transfer learning framework, for generalizing the forecasting DL model to other fresh produce using transfer learning, is detailed in Section 3.3. Finally, the quality assurance techniques used to assure the quality of the deployed model and similarity code are discussed in Section 3.4.

3.1 Imputation Model

It is essential to have an accurate reliable imputation technique due to the scarcity of having datasets with daily samples along with the high possibility that even available daily datasets have many missing values. The problem of missing data imputation is handled previously in [11], where the authors proposed an ensemble imputation model for random missing data and a transfer learning model for cases of missing chunks of data, as explained in subsection 2.2.1. However, a major drawback of the ensemble model is that it works only if the missing values are <50% of the data, and it is found that the random missing is exceeding 50% while the chunks of missing values are less than 50% of the data. Hence, the transfer learning is applied for the missing chunks and the ensemble is replaced by an interpolation technique discussed in subsection 2.2.1 which is found to be effective. It is found that NDVI attribute enhances the performance of the deep learning forecasting model. The deployed imputation technique is based on using the cubic spline

interpolation for interpolating random missing values in the NDVI dataset retrieved from Landsat Satellite [19]; where the readings are taken every 3 days with many missing values in the middle resulting in multiple chunks. This suggests combining the interpolation method with the proposed chunk imputation transfer learning technique proposed in [11] where the base dataset in this case is the NDVI dataset retrieved from MODIS Satellite [19] which is available daily. Despite the high frequency of MODIS, Landsat is still preferable due to its high spatial resolution compared to MODIS.

3.2 Forecasting Models

Different forecasting models are proposed in this work and tested, a simple GRU in addition to several compound models. The first compound model is the ACNNGRU which consists of multiple convolutional layers for capturing the essential information from the input features, an attention layer, and two GRU layers to capture the time-dependent information with 20 units each. The second compound model is the ADGRU which contains multiple time-distributed dense layers, 3 stacked GRU layers with 20 units each and an attention layer. Having multiple dense layers is inspired by the DFFNN structure in [2] to capture the relationship between the independent variables. The last compound model is the DFFNNGRU which consists of 3 consecutive blocks; each block has a time-distributed dense layer followed by a GRU layer with 20 units. The 3 blocks are followed by a flatten and a dense layer. The DFFNNGRU model is used to take advantage of the DFFNN networks in capturing time relevant information from the GRU layer. The models proposed are then combined by a voting ensemble which averages the forecasted output obtained from its component models. This results into 4 ensembles: GRU-DFFNNGRU, GRU-ACNNGRU, ACNNGRU-DFFNNGRU, and DFFNNGRU-ADGRU. Further, the effect of adding the best performing model in the literature; ATT-CNN-LSTM for station-based data and SAE-CNNLSTM for satellite-based data in [2], to the simple and compound models is tested. The best-found model as highlighted in multiple papers [10, 55, 90] is the DFFNNGRU-ADGRU ensemble which found to outperform literature models and all other proposed model ensembles, the model structure is visible in Figure 3.1.

3.3 Transfer Learning

This section provides an overview of the similarity techniques used to identify which time series are similar to each other, which is then utilized in clustering datasets into different

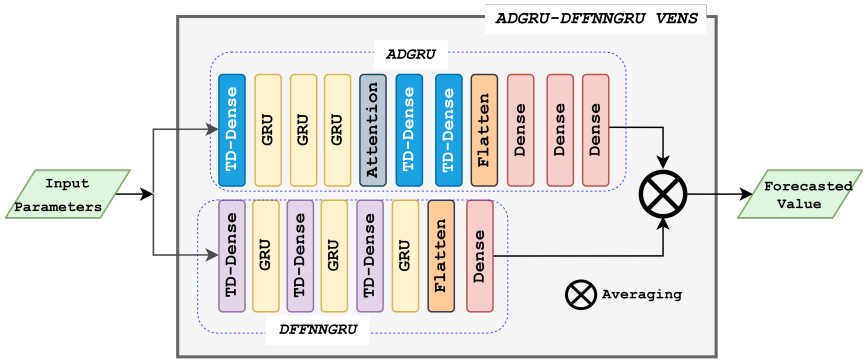

Figure 3.1: The ADGRU-DFFNNGRU model structure resulting from the ensemble of ADGRU and DFFNNGRU using averaging

clusters using hierarchical clustering approach. In addition to covering the transfer learning technique applied to the said model in Section 3.2. to be able to apply the model on similar datasets.

3.3.1 Clustering and Similarity Solution

To enable grouping the datasets into clusters, which is useful in the transfer learning framework, the similarity among the time series must be determined. The similarity method introduced in [2] is utilized to measure the yield similarity among FPs. However, rather than finding the similarity between the time series, the Z-score is calculated instead based on the method in [3]. The Z-score shows the distribution of the values which enables the similarity measure to capture the similarity in the behavior of the time series rather than the similarity based on the exact values of their points. Therefore, the calculated Z-score is used in the utilized similarity measure to get the local features (L) and global (G) features of each point in the time series. The local and global features of the calculated Z-scores (z) are calculated as in (3.1) and (3.2) where each feature is represented by a vector of shape $(2, n)$, where n is the size of the time series. Using these features, a distance measure is calculated as proposed in [2] using (3). Then DTW is used, where DTW mainly overcomes the problem of alignment between two time series $X_{1..N}$, $Y_{1..M}$ creating a distance matrix D which is NxM [29] resulting in the optimal warping pass DTW calculation using the D matrix as in (4).

$$L_{yi} = \{[z_i - z_{i-1}], [z_i - z_{i+1}]\} \tag{3.1}$$

$$G_{yi} = \{[(\sum_{x=0}^{i-1} z_x)/i], [(\sum_{x=i}^{n} z_x)/(n-i)]\} \tag{3.2}$$

Where Z is the z-score, n is the length of the time series, L_{yi} is the local features for a point i in time series y and G_{yi} represents global features for a point i in the time series y.The distance between two points (d), point i in time series t_a and point j in time series t_b is calculated in (3.3).

$$d(t_{ai}, t_{bj}) = |L_{ai} - L_{bj}| + |G_{ai} - G_{bj}| \tag{3.3}$$

The cumulative DTW distance between two time series, Distance(t_a,t_b) is calculated by building up the D matrix using (3.4) where i falls in the range $[0, N]$ and j falls in the range $[0, M]$, then returning the distance result as the value at the last cell of the matrix which can be referenced as $D(N, M)$.

$$D(i, j) = d(t_{ai}, t_{bj}) + min[D(i-1, j-1), D(i-1, j), D(i, j-1)] \tag{3.4}$$

The measured distances among the time-series are used to group the FPs into clusters. Hierarchical clustering is used for that purpose where a dendrogram is built based on the complete linkage explained in subsection 2.4.1. Beside finding the possible clusters, the dendrogram helps in deciding the similarity threshold based on the hierarchical clustering method which gives allowable range for the clustering threshold to avoid setting it arbitrarily based on trial and error as in [3]. Due to using the Z-score, the distance values range from 0-1. Hence, the percentage similarity (PSim) between two FP time series t_a and t_b is calculated using (3.5), and based on the chosen threshold, the binary similarity is deduced using (3.6).

$$PSim(t_a, t_b) = (1 - Distance(t_a, t_b)) * 100] \tag{3.5}$$

$$BSim(t_a, t_b) = \begin{cases} 0, & \text{if } PSim < ST \\ 1, & \text{if } PSim \geq ST \end{cases} \tag{3.6}$$
$$where ST = (1 - CT) * 100$$

Where t_a and t_b are the two FP time series, $PSim$ and $BSim$ are the percentage and binary similarity between them, ST and CT are the similarity and clustering thresholds and the clustering threshold is based on the distance measure.

3.3.2 Proposed Framework

The transfer learning is applied from an existing Base model (FP_{BM}) that is trained using a base fresh produce (FP_B) to get a new target model (FP_{TM}) for a new target fresh produce (FP_T); the FP_B represents the cluster of FPs that is most similar to FP_T. The Transfer learning is applied by freezing some layers of the FP_{BM} then fine-tuning the rest of the layers using the FP_T data. The framework for the transfer learning is illustrated in Figure 3.2.

To find the most similar FP_B to each new fresh produce FP_T, the similarity check is applied as explained in subsection 3.3.1 between that FP_T and the representative FP_B in each of the n clusters, (FP_{B_a}), for a from 1 to n, using the hierarchical clustering algorithm explained in subsection 2.4.1. The FP_{Ba} with the least distance to FP_T is chosen. The Base DL model of the chosen Base FP (FP_{BM_a}) is loaded from the FP Base Models database and the transfer learning is applied to create a new model for the target FP (FP_{TMa}). The FP_{TM_a} is saved in the FP Target Models database to be used for forecasting the FP_T yield. If a similar FP_B to FP_T can't be found, FP_T is assigned a new cluster and a new Base model $FP_{BM(n+1)}$ is created from scratch using FP_T data and saved in the base models database and used for FP_T yield forecasting. Periodic re-clustering is needed which should take place when the total number of single element clusters is exceeding the average size of all other clusters.

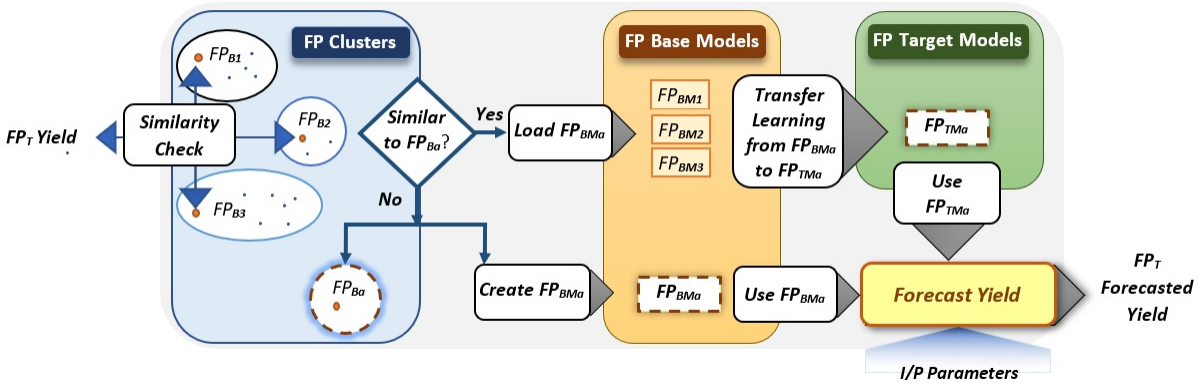

Figure 3.2: The proposed transfer learning framework application [FP_T : new Target Fresh Produce, FP_B: Base Fresh Produce, BM: Base Model, TM: Target Model]

3.4 Quality Assurance

In this section, the proposed quality assurance framework for the application is discussed. The section is divided into two subsections: The first covers the quality assurance of the deep learning model, which is further divided into two sections, one for the quality of the deployed data and the other for the quality of the created model. The second section covers the automatic detection of MRs in software code using machine learning SVM model.

3.4.1 DL Model Quality Assurance Using Metamorphic Tests

The quality assurance of the deep learning models and the utilized data is not a straightforward task. Fewer mechanisms are available for testing the effectiveness of a deep learning model compared to those deployed in software testing. To verify the quality of a DL model and the data used in training, domain experts can define a set of constraints or properties for the data and model then verify the presence of such properties. Hence, metamorphic relations are proposed to define the data properties and model behavior expected by domain experts which can be verified by metamorphic tests (MTs). To ensure the quality of the proposed DL model and its data, in this section the metamorphic relations are designed based on the relations established by domain experts as those in [81, 82, 83]. Due to the similarity between the domain of the proposed work and the work in [81, 82, 83], the metamorphic relations in [81, 82, 83] are utilized after modifying them to fit the tested application beside introducing other new MRs. For consistency, all the relations designed for data verification and model quality assurance are studies and classified under the main categories of metamorphic relations identified in [91] as: Permutative (P), Additive (A), Multiplicative (M), Invertive (IV), Inclusive (I) and Exclusive (E). In addition to two introduced relations: Bounds (B) and Sustainable (S). Finally, proper metamorphic tests are designed to verify the presence of each data and model metamorphic relation.

Data Metamorphic Relations: To assess the quality of the utilized data, a correlation analysis is conducted. Correlation calculations must share known behaviors to facilitate finding metamorphic relations that must be held for the data to be valid. These relations are based on the main categories of metamorphic relations listed above. In this section, nine data metamorphic relations (DMRs) are formally defined for the data deployed by the tested DL model. Those DMRs are classified under four of the main categories of MRs: Permutative, Invertive, Additive, and Multiplicative, in addition to the two categories of Bounds and Sustainable relations. Nine data metamorphic tests (DMTs) are designed to verify each of the defined relations.

1. Bounds metamorphic relation: If the applied function is bounded, the result must not fall out of the identified boundary. The bounds DMR and DMT are as follows:

   - DMR1-B: When calculating the correlation coefficient matrix C for a dataset with n features as in (3.7).

   $$\forall i, j \leq n \implies -1 \leq c_{i,j} \leq +1 \tag{3.7}$$

   DMT1-B: Calculate the correlation coefficient matrix for the dataset then loop over all elements of the matrix to ensure that each value is bounded between -1 and +1.

2. Permutative metamorphic relations: show that a change or permutation in the order of the data does not affect the output. The permutative DMRs and DMTs are as follows:

   - DMR2-P: Changing the location of a feature with respect to the other features should not affect its correlation coefficient with the rest of the features; as defined in (3.8).

   $$\begin{gathered}\text{For a dataset with } n \text{ input features set } \{x_1, x_2, \ldots, x_n\} \text{ and} \\ \text{correlation coefficient matrix } C, \forall i, j \in [1, n], C[x_i, x_j] \text{ should} \\ \text{not change across the } n! \text{ permutations of the feature set.}\end{gathered} \tag{3.8}$$

   DMT2-P: Calculate the correlation coefficient matrix of a dataset with n features. Change the order of the features then recalculate the correlation coefficient matric. The correlation coefficient between any two features should remain the same regardless of their new order.

   - DMR3-P: Changing the order of the data-points should not affect the correlation coefficient between the attributes; as defined in 3.9.

   $$\begin{gathered}\text{For a dataset with } n \text{ rows } \{y_1, y_2, \ldots, y_n\} \text{ and a correlation} \\ \text{coefficient matrix C, C should remain the same for all the} \\ n! \text{ row permutations of the dataset.}\end{gathered} \tag{3.9}$$

   DMT3-P: For a dataset with n rows, calculate the correlation coefficient of the dataset. Shuffle the rows randomly using a random permutation technique then recalculate the correlation coefficient matrix. Compare the two matrices' elements, they should be identical.

- DMR4-P: Two identical features must have a correlation coefficient of 1; as defined in (3.10).

  For a set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}, \forall i \in [1, n]$, duplicating any feature x_i results in a new set of input features with correlation coefficient matrix C such that:
  $$C[x_i, x_{i-duplicate}] = 1 \tag{3.10}$$

  DMT4-P: select any feature from a dataset with n features, duplicate that feature then calculate the correlation coefficient between the selected feature and its duplicate. The calculated correlation should always be equal to the maximum correlation coefficient of 1.

3. Invertive metamorphic relations: explain the effect of inverting some or all of the inputs. The invertive DMR and DMT are as follows:

   - DMR5-IV: Any two opposite features should lead to a correlation coefficient of -1; as defined in (3.11).

     For a set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}, \forall i \in [1, n]$, duplicating the feature x_i and multiplying the duplicate by -1, to get $-x_i'$, should result in a new correlation coefficient matrix of the new set of input features with $C[x_i, -x_i'] = -1$ (3.11)

     DMT5-IV: Having a dataset with n features, duplicate a random feature and add the feature's inverse to the dataset by multiplying its values by -1. Calculate the correlation coefficient which should result in the maximum negative correlation coefficient of -1 between the chosen feature and its inverted duplicate.

4. Additive metamorphic relations: show that adding or subtracting a constant should not influence the results. The additive DMR and DMT are as follows:

   - DMR6-A: Linear Scaling of feature values by addition doesn't affect its correlation; as defined in (3.12).

     For a set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}, \forall i, j \in [1, n]$, scaling feature x_i by adding a constant value b to all its values to get $-x_i + b$, should result in an identical coefficient matrix C to the one without scaling.i.e. $C[x_i, x_j] = C[x_i + b, x_j]$ (3.12)

DMT6-A: Select any of the dataset features. Introduce a new feature through scaling the selected one by adding a constant value to all its values.

Compare the correlation coefficient of the selected feature before scaling and any other feature in the dataset with the correlation after scaling. Both correlations should be equal.

5. Multiplicative metamorphic relations: show that multiplying or dividing a constant should not influence the results. The multiplicative DMR and DMT are as follows:

   - DMR7-M: Linear Scaling of feature values by multiplication leads to the same correlation; as defined in (3.13).

     For a set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}, \forall i, j \in [1, n]$, scaling feature x_i by multiplying all its values by a constant value b to get $-x_i * b$, should result in an identical coefficient matrix C to the one without scaling.i.e. $C[x_i, x_j] = C[x_i * b, x_j]$ (3.13)

     DMT7-M: Select any of the dataset features. Introduce a new feature through scaling the selected one by multiplying a constant value to all its values. Compare the correlation coefficient of the selected feature before scaling and any other feature in the dataset with the correlation after scaling. Both correlations should be equal.

6. Sustainable metamorphic relations: show that if an input causes an error, this error must be handled correctly with a proper technique. The sustainable DMRs and DMTs are as follows:

   - DMR8-S: Introducing a new feature with variance of 0 should result in an error message.
     The correlation coefficient uses standard deviation in its calculation, hence a data with variance 0 causes a zero division, which needs to be properly handled by the application.
     DMT8-S: The correlation coefficient function can be used to calculate a correlation between a zero-variance attribute and another attribute, this should be handled without crashing and an appropriate message should be shown such as "Math Error: Division by zero".
   - DMR9-S: Outliers could significantly alter the results therefore they should first be removed before calculating those results.

DMT9-S: Introduce a new data-point which is a clear outlier and compare the correlation coefficient before and after introducing the outlier. Both correlations should be equal.

Model Metamorphic Relations: To verify the effectiveness and quality of a DL model, metamorphic relations are designed by domain experts to verify the model quality as in [92]. In this section, the learning model metamorphic relations are formally defined along with their metamorphic tests which are designed to verify each of the defined relations. The test is then evaluated based on the Score Percentage Change (SPC) which is the percentage change in the evaluation scores of the original model, i.e. oldScore, and the model after performing the test, i.e. newScore, as defined in (3.14).

$$SPC = \frac{\|oldScore - newScore\|}{oldScore} * 100 \tag{3.14}$$

1. Permutative LMRs and LMTs:

   - LMR1-P: Changing the order of features in models' input should not affect the forecasting scores; as defined in (3.15).

     For a set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}$, the forecasting score should not change across the $n!$ permutations of the feature set. (3.15)

     LMT1-P: Train the model with the dataset then records the model performance scores. Create a new dataset by reordering the features of the original dataset then construct a new model trained on the new data. Test the new model and record the forecasting scores. Compare the scores of the two models. The differences should be insignificant as that caused by variance in initialization.

   - LMR2-P: Changing the location of datapoints in models' input should not affect the forecasting scores; as defined in (3.16).

     For a dataset with n rows $\{y_1, y_2, \ldots, y_n\}$, the forecasting scores should remain the same for all the $n!$ row permutations of the dataset. (3.16)

     LMT2-P: Train a model on a dataset and record the performance scores. Create a new dataset by shuffling the entries of the first dataset. Train and test a new model using the new dataset and record forecasting scores. The performance scores of both models should be approximately equal.

- LMR3-P: Time steps should be considered in order.

  If time-steps are not considered in order, the order in which the data has been input to the model could affect the performance.

  LMT3-P: Train a model and record its forecasting scores. Shuffle the training and validation data then retrain the model and record the new scores. The performance scores should remain the same before and after shuffling.

2. Additive LMRs and LMTs:

   - LMR4-A: Scaling a feature in training and validation by adding or subtracting a constant must not affect the scores; as defined in (3.17).

     For a training or validation set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}$, $\forall i \in [1, n]$, scaling feature x_i by adding or subtracting a constant value b to all its values to get $x_i b$ should not affect the forecasting performance scores. (3.17)

     LMT4-A: Get a random feature from the dataset, then add a constant to all its values. Train the model and measure the performance score. The performance scores before and after adding the constant values to that attribute should remain the same.

   - LMR5-A: Scaling a feature in validation by adding or subtracting a constant must have a huge impact on the performance score; as defined in (3.18).

     For a validation set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}, \forall i \in [1, n]$, scaling feature x_i by adding or subtracting a constant value b to all its values to get $x_i b$ should have a high impact on the forecasting performance scores. (3.18)

     LMT5-A: Select a random feature from the validation dataset then add a constant to all inputs of this feature. Compare the forecasting results before and after scaling the validation dataset, the scores after scaling should have a high percentage change SPC >10% compared to the ones before scaling.

3. Multiplicative LMRs and LMTs:

- LMR6-M: Scaling a feature in training and validation by multiplying/dividing a constant must not affect the scores; as defined in (3.19).

  For a training or validation set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}$, $\forall i \in [1, n]$, scaling feature x_i by multiplying or dividing a constant value b to all its values to get $x_i * /b$ should not affect the forecasting performance scores. (3.19)

  LMT6-M: Get a random feature from the dataset, then multiply a constant by all inputs of this feature. Train the model and measure the performance score, the score before scaling should not significantly differ from the score after scaling.

- LMR7-M: Scaling a feature in the validation dataset by multiplying or dividing a constant must have a huge impact on the model performance score as defined in (3.20).

  For a validation set of n input features $\{x_1, x_2, x_3, \ldots, x_n\}, \forall i \in [1, n]$, scaling feature x_i by multiplying or dividing a constant value b to all its values to get $x_i * /b$ should have a high impact on the forecasting performance scores. (3.20)

  LMT7-M: Select a feature from the validation dataset then multiply all inputs of this feature by a constant. Compare the forecasting results before and after scaling the validation dataset by calculating the SPC; the scores after scaling should highly vary from those before scaling with SPC >10%.

4. Inclusive LMRs and LMTs: shows the effect of adding new elements and assuring the input size fed to the model.

   - LMR8-I: All training data should be included in the generation of a sequence otherwise the effectiveness of the trained model is not guaranteed as defined in (3.21).

     For a dataset with n rows $\{x_1, x_2, x_3, \ldots, x_n\}$, time-steps value t and future days to forecast d, the final size of the dataset should be $n - t - d$ entries. (3.21)

LMT8a-I: truncate the dataset to have exactly the length of time-steps + future days to forecast; a single input record with correct dimensions. The model should train on this single input.

LMT8b-I: truncate the dataset to have length less than time-steps + future days to forecast; the data now has incorrect dimensions. Try to train the model. A suitable error message should be displayed such as "Incompatible dimensions".

- LMR9-I: Assuring that right amount of data is used for validation.

  Having a dataset with n entries, with time-steps value t and future days to forecast d, the model can be able to forecast if $n \geq t + d$.

  LMT9a-I: Validation set is truncated to have exactly the length of time-steps + future days to forecast, the model should be able to forecast the result.

  LMT9b-I: Validation set is truncated to have length less than time-steps + future days to forecast, the model must give suitable error message such as "Incompatible dimensions".

- LMR10-I: Inclusion of informative attributes should enhance the model performance.

  A model should perform better when adding more informative attributes with low correlation to the rest of the model attributes.

  LMT10-I: Train a model with a subset of attributes, add a low correlated attribute with the model attributes, and highly correlated with output. The model performance scores should improve.

- LMR11-I: The model's performance should not enhance if a highly correlated attribute with the model's attributes is added.

  LMT11-I: Train a model with a set of attributes. Add a highly correlated attribute with one or more of the attributes used in training the first model. Compare the new model's performance score to the old model, no significant enhancement should be recognized.

5. Exclusive LMRs and LMTs: shows the effect of removing elements and informative attribute to the model.

   - LMR12-E: Introducing training data with range 0 should be handled by the application.

     LMT12-E: Set a data attribute to a constant value to have a range of 0, despite the fact that the model training should not be affected, no useful information is gained from such attributes.

     Hence, this should result in having low model performance scores.

- LMR13-E: Introducing validation data with range 0 should result in an error.
  LMT13-E: Set the validation attribute to 0, the model should be able to train correctly but validation scores should result in an error therefore an error message should be issued such as "Validation data cannot be constant".

3.4.2 Automatic Detection of MRs in Software Code using ML

Automatic detection of metamorphic relations is investigated by researchers, an effective machine learning Support Vector Machine (SVM) model is found and used in [80, 93] to automatically detect metamorphic relations. The code is first translated into Control Flow Graphs (CFGs), which is a graph that can be used to represent the software code using nodes and edges as connectors. The CFG is analyzed using graph kernels, which are functions used to measure the similarity between pairs of graphs that are used in classification [94]. The similarity between the two graphs is calculated by a Random Walk Kernel [95], which corresponds to the atomic operation execution in the program. The similarity score is then normalized, and the kernel then maps the CFGs into a feature space.

After calculating the similarity score, the feature-based content of the CFGs is manually labelled by experts using three metamorphic relation labels: permutative, additive and inclusive. To enable the model to identify whether a metamorphic relation is valid or not, for each metamorphic relation a model is trained. To improve the models' performance the dataset must be enlarged by applying and testing the four types of mutants suggested in [80]: logically equivalent mutants of atomic operations, non-logically equivalent mutants of atomic operations, equivalent mutant of varying loop logic, and logically equivalent dead code mutants.

The data is then fed into an SVM classifier which is tested before and after mutations. The classification accuracy is found to depend on the type of mutant used. The Non-logically equivalent mutants (M2) degrade the accuracy; hence they are not considered in the experiment.

The detection of metamorphic relations is usually done by domain specialists, metamorphic relation is detected based on the method in [80] using machine learning. This automatic detection of MRs is achieved by having a huge dataset of software codes which are translated into Control Flow Graphs (CFGs) [87]. The CFGs are then translated into features by using graph kernels. The features retrieved from the CFGs are then fed into a support vector machine (SVM) model, which identifies whether a certain metamorphic relation exists or not. To assure the accuracy of the data fed to the SVM model, the codes

are labelled manually, with a yes/no flag reflecting whether the input metamorphic relation exists or not in the code. To increase the size of the training dataset, mutation operators are applied on the software codes, and the models are retrained.

After the models are ready, they are used to detect whether certain MRs are available in the generalization code or not, applying the same procedure explained above to the generalization code. After extracting the features, the CFGs of different codes are fed into three different types of metamorphic relation detection models: Additive, Permutative and Inclusive relations. To ensure best practice, the highest performing model out of each type is used and before running the models, the software code is labelled according to the expected metamorphic relations which should exist in that code; as illustrated in Figure3.3.

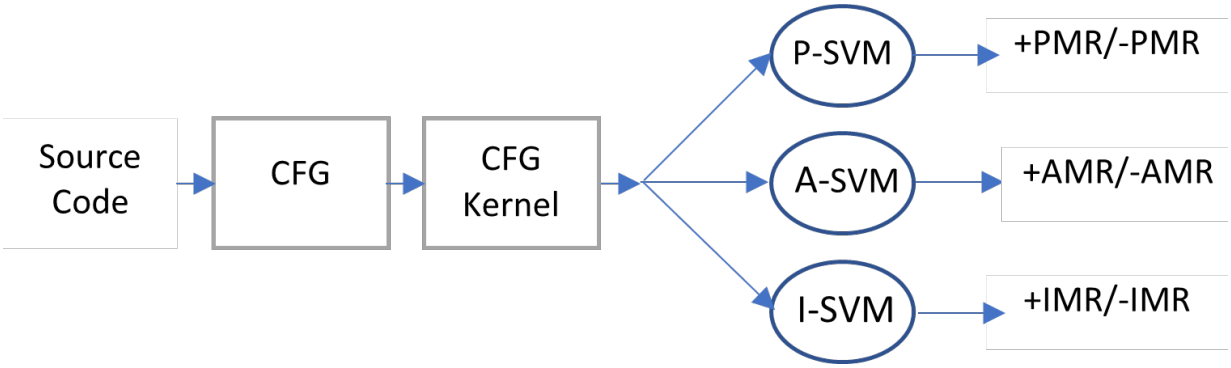

Figure 3.3: Automatic detection of metamorphic relations in software codes using SVMs

Chapter 4

Experiments and Found Results

The conducted experiments along with their found results are summarized and discussed in this section. In Section 4.1, the experiments carried out for finding the best imputation method for estimating the missing values are illustrated and their found results are discussed. In Section 4.2, the experiments focus on improving the forecasting model: subsection 4.2.1 concentrates on finding the best yield and price forecasting models using station-based data while subsection 4.4.2 considers finding the best forecasting models using station-based data, assesses the data preprocessing methods and the satellite data parameters used in training the model by measuring the effect on the model's forecasting performance. In Section 4.3, the transfer learning techniques are tested and evaluated by applying them to the best-found model to enable the generalization of the model to similar datasets. In Section 4.3, the proposed quality assurance techniques, for assuring the quality of the data and model, are tested and the found results are analyzed.

4.1 Finding the Best Statistical Imputation Model

In this experiment, the effect of imputing the vegetation indices (VIs) parameters using the statistical imputation approaches on the forecasting performance is explored; the VI parameters are added to the soil moisture and temperature input parameters. Based on [28], the three major vegetation indices that have the most impact on forecasting yield are NDVI, NDTI, and EVI. Those three vegetation indices are appended to the soil moisture and temperature parameters dataset in [2], as in subsection 2.1.1. Hence, the resulting dataset includes five input features: soil moisture, soil temperature, NDVI, NDTI and EVI, which are mapped to the yield output parameter.

Since the dataset contains a huge number of missing values, interpolation is essential for estimating those missing values to improve the training of the forecasting models. Four statistical interpolation techniques are tested and evaluated by assessing their effect on enhancing the forecasting performance: the lower the resulting aggregated error measure (AGM) the higher the performance and the better the interpolation technique.

The interpolation techniques are assessed by using the vegetation indices interpolated by each technique along with the soil moisture and temperature parameters in training the DL forecasting model as input parameters. The best interpolation model is the one that leads to the best forecasting performance or the lowest AGM.

The Exact Replication is tried first as a base method to compare its performance to the considered four interpolation methods by calculating the percentage change in performance caused by each method. The Vegetation Indices dataset is then fed to an interpolation method, outputting all the features interpolated depending on the type of interpolation used from the 4 interpolation techniques explained in subsection 2.1.2- Statistical Approaches. Four interpolation methods are tested and their impact on the forecasting performance is evaluated and compared to the simple Exact Replication method. The Exact Replication technique is applied to the VIs before using them as input parameters to the ensemble of the deep learning models. The resulting forecasting AGM score is found to be 9.81 which is compared to the other four interpolation methods as follows:

1. The Linear interpolation technique is applied to the vegetation indices before using them to forecast the yield. This achieves an AGM score of 9.94 showing a close performance to the replication method with 1.4% degraded performance.

2. The Bessel interpolation technique is then applied which reduces the AGM score to 8.04 showing an improvement of 18% over the replication method.

3. The One-Way Spline method achieves an AGM score of 5.77 showing a 42% performance enhancement compared to the replication method.

4. The Cubic Spline interpolation technique further reduces the AGM score to 5.53, showing an improvement of 44% to the replication method.

To manually verify and confirm the found results, the performance of applying the interpolation techniques on the NDVI parameter is visually assessed as well. As evident from Figure 4.1: The yellow line representing the NDVI values after applying the Exact Replication technique is not smooth and contains either sudden vertical shifts in values or straight horizontal segments due to having many repetitive values, which reflects very bad

performance. The red line represents the NDVI values after linear interpolation. The only difference between the linear and replication lines is that the linear technique connects the two consecutive points by straight lines, which could result in a smoother signal, however the results are peaking only at known points, which also explains having the worst performance. The lines representing the three remaining interpolation techniques, Bessel, Cubic Spline and One-Way Spline, are very close to each other. However, the Cubic Spline blue line has more curved edges in undetermined points like on the point between 23rd and 30th of November, having more concise parabolas is behind getting the best performance. The Bessel (Green) and One-Way (Gray) lines are overlapping, however, a closer look at Figure 4.1 reveals that the values resulting from deploying the One-Way Spline method are smoother near the known points which justifies its higher performance.

The results of the conducted experiments across the 3 main vegetation indices prove that the top two forecasting performances are achieved when using the Cubic Spline and One-Way Spline interpolation techniques.

Figure 4.1: NDVI values across three months estimated using replication and four other interpolations methods

4.2 Finding the Best Forecasting Model

This section is divided into two subsections: one covers the experiments conducted to find the best forecasting model for fresh produce yield and price using station-based data, while the other explores the use of satellite-based data in forecasting.

4.2.1 Forecasting Using Station Based Data

The main target of this group of experiments is to find the best model for forecasting the yield and price in Santa Maria. The inputs to the models are the soil, yield and price datasets discussed in subsection 2.1.1 after deploying the preprocessing elaborated in subsection 2.2.2. The output is 5 weeks ahead of the yield or price. To evaluate the performance of the forecasting models proposed in subsection 2.3, the evaluation metrics stated in subsection 2.3.2 are used where the target is reducing the AGM score without degrading the R^2 score.

1. Yield Forecasting: Different combinations of models are tested and compared to the literature model in [2]. First, the model in [2] is modified by replacing the LSTM layers with GRU layers resulting in the ACNNGRU model. It is found that this model improves the AGM score by 0.47 while degrading the R^2 score by 1%. This shows the higher effectiveness of the GRU in this task compared to the LSTM. Then, novel ADGRU and DFFNNGRU models are proposed, both showing same R^2 scores as the ACNNGRU, but the ADGRU is found to reduce the performance leading to an AGM score higher by 0.02 compared to the ACNNGRU. However, the DFFNNGRU slightly improves the AGM by 0.02. Finally, a simple GRU model is tested, which degrades the R^2, and AGM scores compared to the model in [2], these results are reported in Table 4.1.

   The proposed models are further improved by introducing the ensemble which combines different models together using the average prediction method explained in subsection 2.2.2, Satellite Based Data Preprocessing where a representative average is used to represent each satellite image available. Due to the simplicity of the GRU model, the effect of ensembling it with other models is explored. In addition, the ATT-CNN-LSTM model ensembling is investigated as well because of its high performance as reported in many literature works. The results of the ensembles are visible in Table 4.1. Clearly the ensemble of the proposed DFFNNGRU with ADGRU, or DFFNNGRU-ADGRU, gives the best results improving the R^2 score from

Table 4.1: Results of simple, compound and ensembled FP yield forecasting models.

Evaluation Metric	Simple	Compound			Ensemble								Best in Literature
	GRU	*ACNN -GRU*	*ADGRU*	*DFFNN -GRU*	*ATT-CNN-LSTM ACNN -GRU*	*ATT-CNN-LSTM ADGRU*	*ATT-CNN-LSTM GRU*	*ATT-CNN-LSTM DFFNN -GRU*	*GRU DFFNN -GRU*	*GRU ACNN -GRU*	*ACNNGRU DFFNN -GRU*	*DFFNNGRU ADGRU*	*ATT-CNN-LSTM SeriesNet -GRU*
MAE	36.76	36.25	35.14	36.52	35.81	33.96	33.83	35.23	34.62	33.98	34.46	33.54	40.70
RMSE	53.31	51.57	51.88	51.47	52.02	50.14	49.87	50.74	50.32	49.54	51.47	48.09	58.8
R^2	0.83	0.84	0.84	0.84	0.84	0.85	0.85	0.85	0.85	0.85	0.85	0.86	0.85
AGM	7.70	7.03	7.05	7.01	7.15	6.36	6.26	6.66	6.47	6.17	6.13	**5.68**	7.5

Table 4.2: Results of simple, compound and ensembled FP price forecasting models.

Evaluation Metric	Simple	Compound			Ensemble								Best in Literature
	GRU	*ACNN -GRU*	*ADGRU*	*DFFNN -GRU*	*ATT-CNN-LSTM ACNN -GRU*	*ATT-CNN-LSTM ADGRU*	*ATT-CNN-LSTM GRU*	*ATT-CNN-LSTM DFFNN -GRU*	*GRU DFFNN -GRU*	*GRU ACNN -GRU*	*ACNNGRU DFFNN -GRU*	*DFFNNGRU ADGRU*	*ATT-CNN-LSTM SeriesNet -GRU*
MAE	0.2	0.21	0.19	0.22	0.2	0.2	0.19	0.2	0.2	0.19	0.2	0.2	0.23
RMSE	0.25	0.25	0.24	0.28	0.25	0.24	0.25	0.24	0.25	0.24	0.25	0.24	0.29
R^2	0.75	0.75	0.78	0.7	0.76	0.77	0.76	0.77	0.75	0.77	0.75	0.77	0.68
AGM	0.06	0.06	0.05	0.07	0.05	0.05	0.05	0.05	0.06	0.05	0.06	**0.05**	0.08

previous literature by (1%) and reducing the AGM score by 1.82 which is about 24% improvement compared to the ensembled models in [2].

2. Price Forecasting: For forecasting the price of strawberry in Santa Maria, the combinations of models mentioned above are utilized for price forecasting. The results of the simple and compound models all prove enhancement of price forecasting results as shown in Table 4.2. It is found that the ensemble of DFFNNGRU ADGRU improves the AGM by 37.5% and the R^2 score by 9% compared to the literature model. However, forecasting the price can be affected by other economic factors, not only the environmental ones which affect the reliability of the forecasting model. Hence, it is important to consider the performance of the chosen model in both yield and price forecasting applications. Therefore, despite having more than one price forecasting model with the same AGM, the DFFNNGRU ADGRU ensemble is preferred for its high-performance persistence across both yield and price forecasting applications.

4.2.2 Forecasting Using Satellite Based Data

Three experiments are conducted to improve the satellite-based forecasting: The first experiment determines which data preprocessing technique is more effective whether the prelagged histograms method proposed in [18] or the averaging method utilized in [90]. The second experiment provides a comparison between the proposed model and the literature model of SAE-CNNLSTM ENS, and determines the best model based on the AGM scores. The third experiment determines whether the addition of the satellite NDVI input attribute improves the best-found model's performance as claimed by [55] or not. The Percentage

Table 4.3: Comparison of the two preprocessing techniques by training the best literature satellite model once with satellite parameters' histograms and another with their averages then evaluating the forecasting performance using assessment measures.

Assessment Measures	**Training with Satellite Parameters' Histograms**			**Training with Satellite Parameters' Averages**		
	CNN-LSTM	***SAE***	***CNNLSTM-SAE_ENS***	***CNN-LSTM***	***SAE***	***CNNLSTM-SAE_ENS***
R^2	0.8	0.81	0.83	0.86	0.8	0.85
MAE	38.3	40.6	36.83	35.31	42.14	36.18
RMSE	56.69	55.1	52.9	49.34	59.57	51.84
AGM	9.48	9.01	7.8	5.94	10.4	**6.82**

Improvement (PI) in models' performance based on AGM is calculated as in (4.1).

$$PI = \frac{|Score_a - Score_b|}{\max(Score_a, Score_b)} \times 100 \tag{4.1}$$

1. Finding the Best Data Preprocessing Method: For this experiment two data preprocessing techniques are utilized; one is using histograms to represent the satellite images and the other uses pixels averaging. The comparison is done using the SAE-CNNLSTM ENS, which is trained once using the histograms of the input parameters, and another using the averages where each parameter is represented by the average of the satellite image pixels of the area under investigation. The forecasting performance assessment measures are used to compare the two models as in [18]; the found results are presented in Table 4.3.

   The results show that even though histograms should give better results, as they preserve more spatial information, the model trained using the satellite parameters' average is found to perform better than the model trained using their histograms by an AGM percentage improvement of 12.5%. Moreover, calculating the satellite parameters' averages is more efficient than finding the histograms as it reduces the processing time.

2. Finding the Best Forecasting Model: In this experiment the two DL models in of CNN-LSTM-SAE Ens and DFFNNGRU-ADGRU Ens are compared using the same training and test sets. The models' performance is evaluated based on the AGM scores. Based on the found results in Table 4.4, the proposed satellite model DFNN-ADGRU ENS is found to outperform the literature CNN-LSTM-SAE Ens by a PI of 20%. This shows that the proposed model would be more effective in forecasting fresh produce yield.

3. Enhancing the Input Parameters Set: In [55], the authors eliminated PCA dimensionality reduction; where informative features is extracted to better represent the data

Table 4.4: Performance Comparison of CNN-LSTM-SAE_Ens vs DFNNGRU-ADGRU_Ens when trained on satellite parameters averages.

Assessment Measures	Best Satellite Literature Model			Proposed Satellite Ensemble Model		
	CNN-LSTM	***SAE***	***CNNLSTM-SAE_ENS***	***DFNNGRU***	***ADGRU***	***DFNNGRU-ADGRU_ENS***
R^2	0.86	0.80	0.85	0.86	0.84	0.86
MAE	35.31	42.14	36.18	33.31	36.23	32.05
RMSE	49.34	59.57	51.84	49.75	52.06	48.42
AGM	5.94	10.40	6.82	5.92	6.89	**5.45**

Table 4.5: Performance Comparison of DFNNGRU-ADGRU Ens. with and without the addition of the NDVI parameter

Assessment Measures	With NDVI			Without NDVI		
	DFNNGRU	***ADGRU***	***ENS***	***DFNNGRU***	***ADGRU***	***ENS***
R^2	0.85	0.86	0.87	0.86	0.84	0.86
MAE	34.92	33.42	32.05	33.31	36.23	32.05
RMSE	51.39	49.37	48.12	49.75	52.06	48.42
AGM	6.57	5.81	**5.35**	5.92	6.89	5.45

[96], as it is found to have negligible effect, authors also found that adding the NDVI satellite parameter as input helps in improving the forecasting model's performance, hence the effectiveness of adding the NDVI parameter as input to train the best found model, DFNNGRU-ADGRU ENS, is tested in this experiment. The ensemble is trained by adding NDVI to the input features, leading to feature dimensions of 140 days by 3 features rather than 2 features, and the model is evaluated on the same test set with the same date range as the previous experiment. Table 4.5 shows the forecasting performance after NDVI addition where a slight performance improvement of 2% in the AGM PI is noticed. This suggests adding the NDVI especially that this addition doesn't affect the training time.

4.3 Finding the Best Transfer Learning Method

The transfer learning is applied using the best performing ensemble model of ADGRU and DFFNNGRU, or ADGRU_DFFNNGRU found in subsection 4.2.1, as the Base model. The Base model blue layers in Figure 3.1 are frozen while the rest of the layers are finetuned using FP_T data which takes 1 to 2 epochs to converge leading to fast efficient training. The models are applied by creating a Base model for strawberry (FP_{B1}) and another for blueberry (FP_{B2}) then using the framework proposed in subsection 3.3.2, the input to the framework is the yield of a new fresh produce which is raspberry as FP_T. The similarity

check is applied to find the similarity between raspberry and strawberry then between raspberry and blueberry yield time series.

The similarity check can result in the most straightforward case of finding one similar Base FP_B to the FP_T which is happening in this case where it is found that the $PSim$ of raspberry with strawberry is 76% and 62% with blueberry. Hence, based on the $BSim$ with 70% ST, it can be deduced that raspberry is similar to strawberry and dissimilar to blueberry. Therefore, the strawberry is chosen as the FP_B and the strawberry Base model FP_{BM} is loaded and deployed in the transfer learning for creating a new Target model for raspberry. The Target model is saved in FP Target Models database to be used in forecasting raspberry yield. The other two scenarios that can occur after applying the similarity check are: First, finding more than one similar Base model, in this case the closest one with the highest percentage similarity to FP_T is chosen as FP_B. Second, not finding any similar FP_B, in this case a new cluster is created with the FP_T as its FP_B. A new FP_{BM} for that FP_B is found by training using the new fresh produce yield data. The resulting new Base model FP_{BM} is added to the Base models database. In Table 4.6, the performance of the proposed Target model is compared to the literature model trained using strawberry data and finetuned using raspberry data as well as a model trained using only strawberry data and another trained using only raspberry data without any TL. The assessment of the obtained transfer learning results is essential for verifying the proposed transfer learning method, and models.

First, the target model is compared to using the strawberry base model for forecasting raspberry yield. As expected, the strawberry model could not predict the raspberry yield with high accuracy. Secondly, the TL model is compared to the original raspberry model trained using the raspberry data without TL. A noticeable improvement of 4% in R^2 and 14% in AGM is evident in Table 4.6. When the TL model results are compared to the literature transfer learning model in [2], a noticeable improvement of 1.5% in R^2 and 2.2% in AGM is found.

The results in Table 4.6 prove the effectiveness of the proposed transfer learning technique in transferring the learned features of the Base model to the Target model.

4.4 Quality Assurance

In this section, the designed metamorphic tests in the proposed solution described in Section 3.4 are implemented and assigned specific test cases for testing and verifying the effectiveness of the tested deep learning model proposed in Section 3.2 along with the

Table 4.6: Raspberry yield forecasting using proposed TL model compared to the model without TL, the model trained only on strawberry and the literature model.

Evaluation Metric	Strawberry Model	Raspberry Model (No TL)	Raspberry Model (TL)	Literature Model
MAE	91.56	26.16	24.73	24.14
RMSE	141.66	34.96	33.28	31.77
R^2	-3.73	0.71	**0.74**	0.724
AGM	551.87	8.81	**7.58**	7.71

dataset utilized in its training, discussed in subsection 2.1.1. The test cases are executed to assure the quality of the DL model along with its data, then the found results are reported and discussed. Finally, the quality of the software code is tested using the SVM-ML technique, which uncovers the underlying MRs (Permutative, Additive, and Inclusive) in the code as discussed in subsection 3.4.2.

4.4.1 Data and Model Assessment with Metamorphic Tests

Two types of assessment are conducted for the quality assurance using metamorphic tests.

Data Assessment: The 9 DMTs, designed in subsection 3.4.1 to assess the correlation based DMRs, are implemented and specific test cases are articulated then executed for each. The 9 DMTs, the tested datasets and details of the conducted experiments are summarized in Table 4.7 with the found results and the taken corrective actions.

Model Assessment: To assess and verify the deep machine learning model, the metamorphic relations (LMRs) designed in subsection 3.4.2 are tested using the designed metamorphic tests (LMTs). Those metamorphic tests and the data used in them are listed in Table 4.8 along with their outcome. The found results verify that all the metamorphic relations are fulfilled, which proves the validity of the proposed model.

4.4.2 Quality Assurance of Software Code

To test the generalization code in subsection 3.3.1 and validate it, the nondeep-ML SVM model is used as illustrated in Figure 3.3, where 3 different models are trained for the detection of additive, permutative and inclusive metamorphic relations. Then these models are tested on the four methods of the generalization code explained in subsection 3.3.2: GlobalFeaturesMethod, LocalFeaturesMethod, DistanceBetweenPointsMethod, and SimilarityDTWMethod. Each method is then translated into its equivalent CFG; a sample

Table 4.7: Experiments and results of Metamorphic Tests on the data

MT	Utilized Dataset (DS)	Experiments & Expected Results (ER)	Outcome & Corrective Actions	Results
DMT1 -B	All DS Parameters	1- The correlation coefficient matrix is calculated 2- Matrix elements are checked to find whether or not the values lie within the bounds ER: Correlation values bounded in [-1,1]	Correlation values are bounded in [-1,+1]	DMR1-B Fulfilled
DMT2 -P	All DS Parameters	1- The correlation coefficient matrix is calculated 2- The correlation is recalculated after shifting the soil moisture parameter with the surface temperature ER: Correlation remains the same	Correlation values remain the same after shifting columns	DMR2-P Fulfilled
DMT3 -P	All DS Rows	1- The correlation coefficient matrix is calculated 2- The data rows are shuffled randomly by getting a permutation of indices of the rows and arranging them accordingly 3- A new correlation matrix is calculated and compared to the first one ER: Correlation remains the same	Correlation values remain the same after shuffling rows	DMR3-P Fulfilled
DMT4 -P	Soil Moisture	1- The soil moisture attribute is duplicated, and the copy is named as soil moisture 2 2- The correlation coefficient between the copy and the original attribute is computed ER: Correlation should be 1	Correlation between Soil Moisture and its duplicate is 1	DMR4-P Fulfilled
DMT5 -IV	Soil Temperature	1- The soil temperature attribute is duplicated The duplicate values are multiplied by -1 and named the soil temperature inverse 2- The correlation coefficient between the copy and the original attribute is computed ER: Correlation should be -1	Correlation between Soil Temperature and its inverse is -1	DMR5-IV Fulfilled
DMT6 -A	Soil Moisture	1- The correlation coefficient matrix is calculated 2- The soil moisture parameter is modified by adding 10 to all its entries 3- The correlation coefficient matrix is recalculated 4- All correlations between the dataset attributes and the original soil moisture parameter are compared to the correlations between the same set of attributes and the scaled new soil moisture parameter ER: Correlation remains the same	Correlation between all attributes and scaled Soil Moisture remain the same	DMR6-A Fulfilled
DMT7 -M	Soil Moisture	1- The correlation coefficient matrix is calculated 2- The soil moisture parameter is modified by multiplying its entries by 10 3- The correlation coefficient matrix is recalculated 4- All correlations between the dataset attributes and the original soil moisture parameter are compared to the correlations between the same set of attributes and the new soil moisture parameter ER: Correlation remains the same	Correlation between all attributes and scaled Soil Moisture remain the same	DMR7-M Fulfilled
DMT8 -S	All DS Parameters & Temp_Att of constant value 100	1. A new attribute named Temp_Att is introduced with a value of 100 for all entries 2. The correlation coefficient matrix is calculated ER: Correlation is 0	Correlation between all attributes and Temp_Att is 0	DMR8-S Fulfilled
DMT9 -S	All DS Parameters with row 100 scaled to MAX*100	1- The values in entry with index 100 in the dataset values to maximum_value of each attribute *100 2- The correlation coefficient is calculated and compared where it should not affect our model ER: Correlation remains the same	Correlation changed, *DMR9-S Not-Fulfilled*, Corrective Action: outlier detection method is added, and outliers removed, then DMR9-S is fulfilled	DMR9-S Fulfilled

Table 4.8: Experiments and results of the Metamorphic Tests on the forecasting model

MT	DS Params	Expected Output After Applied Test	Found Outcome	Result
LMT1-P	SM - ST	SPC < 10	SPC < 10: (MAE =2.4, RMSE= 0.1, R^2 =0.0)	LMR1-P Fulfilled
LMT2-P	SM - ST	SPC < 10	SPC < 10: (MAE 0.0, RMSE= 0.5, R^2 =0.0)	LMR2-P Fulfilled
LMT3-P	SM - ST	SPC < 10	SPC < 10: (MAE 0.0, RMSE= 0.5, R^2 =0.0)	LMR3-P Fulfilled
LMT4-A	SM - ST	SPC < 10	SPC < 10: (MAE 1.3, RMSE= 1.1, R^2 =0.0)	LMR4-A Fulfilled
LMT5-A	SM - ST	SPC > 10	SPC > 10: (MAE 163.0, RMSE= 196.1, R^2 =5121.6)	LMR5-A Fulfilled
LMT6-M	SM - ST	SPC < 10	SPC < 10: (MAE 1.1, RMSE= 1.1, R^2 =0.1)	LMR6-M Fulfilled
LMT7-M	SM - ST	SPC > 10	SPC > 10: (MAE 128.1, RMSE= 182.1, R^2 =591.9)	LMR7-M Fulfilled
LMT8a-I	SM - ST	Model is able to train	The model is trained	LMR8-I Semi-Fulfilled
LMT8b-I	SM - ST	Error	Message: “Incompatible dimensions”	LMR8-I Fulfilled
LMT9a-I	SM - ST	Model is able to forecast	The model can forecast	LMR9-I Semi-Fulfilled
LMT9b-I	SM - ST	Error	Message: “Incompatible dimensions”	LMR9-I Fulfilled
LMT10-I	SM, SM - ST	SPC > 10	SPC > 10: (MAE= 7.1, RMSE= 54.0, R^2 =40.3)	LMR10-I Fulfilled
LMT11-I	SM - ST - SurfaceT	SPC < 10	SPC < 10: (MAE= 2.0, RMSE= 1.7, R^2 =1.0)	LMR11-I Fulfilled
LMT12-E	SM - Const =10	SPC > 10	SPC > 10: (MAE= 34.2, RMSE= 35.0, R^2 =40.3)	LMR12-E Fulfilled
LMT13-E	SM - ST	Error	Message: “Validation data cannot be constant”	LMR13-E Fulfilled

transition for the GlobalFeatureMethod to its equivalent CFG is illustrated in Algorithm 1 and Figure 4.2. Each of the four translated methods is labelled manually with three labels +/-PMR, +/-AMR and +/- IMR where positive means that the code fulfils the MR while negative means the MR is not fulfilled by the code. Finally, each equivalent CFG is fed into the three SVM models for detecting the metamorphic relations as illustrated in Figure3.3. The results are recorded in Table 4.9.

Algorithm 1 Global Features Pseudo-code

```
1: procedure GlobalFeaturesMethod(r0,r1)
   ▷ r0 ← Array of feature inputs
   ▷ r1 ← length of r0
2:     f0 ← new array[r1,2]
3:     for i < r1 − 1 do
4:         f1 ← sum(r0,0,i-1)
5:         f2 ← sum(r0,i,r1)
6:         f0[i, 0] ← f1/(i-1)
7:         f0[i, 1] ← f2/(r1-i)
8:     end for
       return f0
9: end procedure
```

Table 4.9: Results of automatic detection of the metamorphic relations in the generalization code.

Tested Code	Permutative Model M1_M3_M4		Additive Model Original DS		Inclusive Model M1_M3	
	P-SVM O/P	Expected O/P	A-SVM O/P	Expected O/P	I-SVM O/P	Expected O/P
Global Features	-PMR	-PMR	+AMR	+AMR	+IMR	+IMR
Local Features	-PMR	-PMR	+AMR	+AMR	+IMR	+IMR
DistanceBetweenPoints	+PMR	+PMR	-AMR	-AMR	+IMR	+IMR
SimilarityDTW	-PMR	-PMR	-AMR	-AMR	+IMR	+IMR

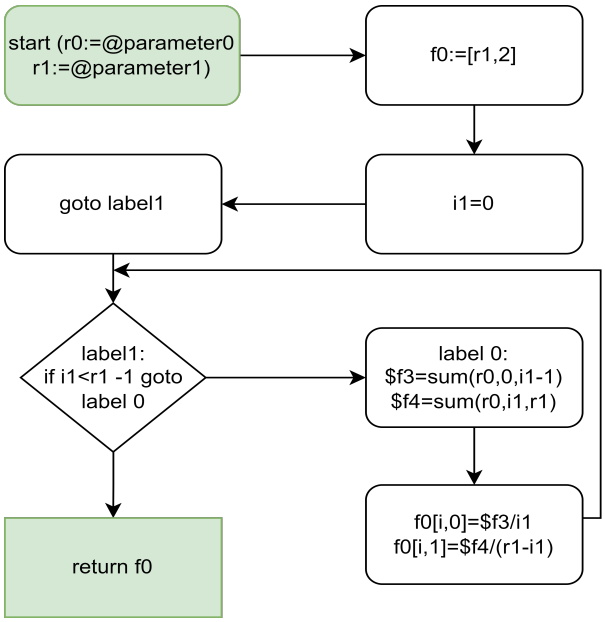

Figure 4.2: CFG created for the global features Algorithm 1

Chapter 5

Web Application

This thesis focuses on building a quality assured framework that enables the yield and price of fresh produce forecasting. The work in Chapters 3 and 4 covers the fresh produce deep learning forecasting model, the required preprocessing before the model training, the data sources and the quality assurance of the model and data. Finally, a transfer learning module is proposed to enable the generalization of the deep learning model to other fresh produce in different geographical locations. To facilitate the ease of deployment of the proposed framework by end users, a Fresh Produce Forecasting web application is developed for forecasting the yield and price using the input parameters of soil moisture and temperature, in addition to the NDVI for the satellite-based forecasting. The framework is developed using ReactJS for the front-end client-side development, and Flask for the backend server-side tasks.

The main objective of this application is to enable users to use the best-found deep learning models to forecast the yield and price of any fresh produce given the fresh produce yield. In addition, the application has an imputation module with an imputation method for estimating high rates of missing data that can be greater than 50%. Finally, the application embeds a similarity module, which enables the user to find binary and percentage similarity between any two time series.

In this chapter, subsection 5.1 covers the similarity module with sample use cases. The imputation module features are highlighted in subsection 5.2 with use cases, and subsection 5.3 the forecasting module with sample use cases from scratch training and transfer learning with a visualization of how the transfer learning framework introduced in 3.3.2 is executing.

5.1 Similarity Module

The similarity module is used to find the similarity between two time series, the similarity technique embedded in this module is explained thoroughly in subsection 3.3.1. In the similarity module, three similarity options are available; measuring the similarity among model inputs, measuring the similarity between model outputs, and measuring the similarity between models based on both, their inputs and outputs, with the relation between inputs and outputs considered.

The similarity module asks the user to input the required type of similarity, upload two CSV files for the time series, and select the parameters used in testing the similarity of both files. Once the parameters are selected, a button becomes visible for the user to calculate the similarity. The similarity technique expects as an input two time series and returns the binary and percentage similarity.

Two use cases are presented for the similarity module: the first is for two presumably similar time series, and the second is for two presumably non-similar time series. The use cases and their results are reported in subsection 5.1.1 and 5.1.2 respectively.

5.1.1 Case 1: Similar Time Series

In this use case, the similarity type being tested is input-output. Two files are uploaded which are presumably similar, one file is for Strawberry fresh produce which is retrieved from Santa Maria County, and the other file is for Raspberry fresh produce which is retrieved from Santa Maria County as well. The chosen columns to find the similarity are: Temperature (K), Moisture (mm) for input parameters, and the output parameters are called Strawberry and Raspberry for each file respectively representing the fresh produce yield. The similarity is computed, and the results of this use case are visible in Figure 5.1.

5.1.2 Case 2: Dissimilar Time Series

In this use case, the similarity type being tested is input-output. Two files are uploaded which are presumably dissimilar, one file is for the strawberry fresh produce which is retrieved from Santa Maria County, and the other file is for blueberry fresh produce which is retrieved from Ventura County. The chosen columns to find the similarity are: Temperature (K), Moisture (mm) for input parameters, and the output parameters representing the yield are called Strawberry and Blueberry respectively for each file. Then the similarity is computed. The results of executing this use case are visible in Figure 5.2.

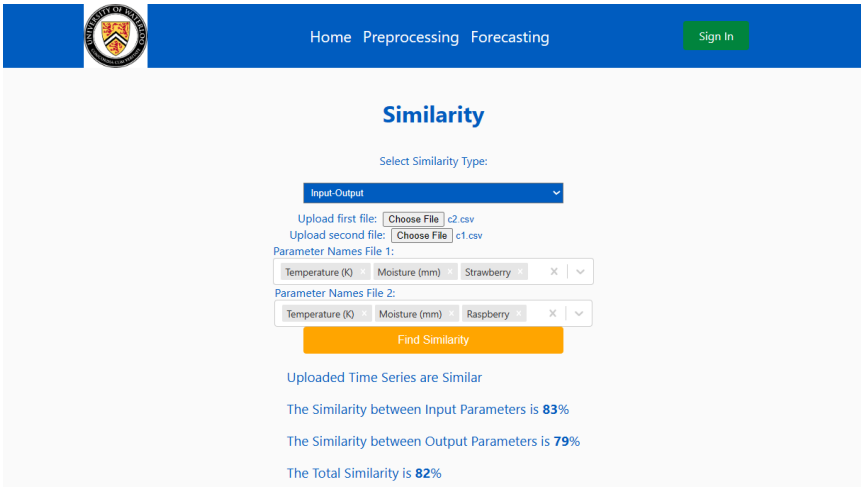

Figure 5.1: Similarity module use case of two similar time series

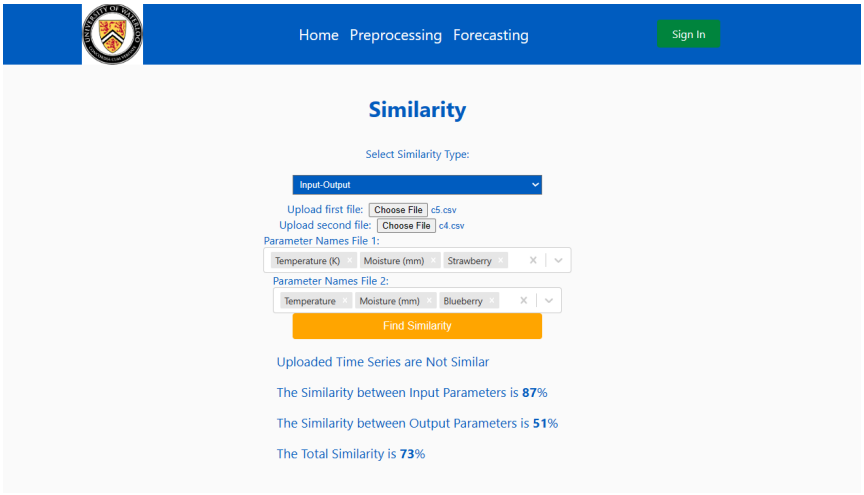

Figure 5.2: Similarity module use case of two dissimilar time series

5.2 Imputation Module

The imputation module is used to surpass the limitation of having missing values in time series which can affect the forecasting performance especially for daily time series. The imputation module uses established deep learning imputation techniques, with the addition of an interpolation technique that is found to be effective with time series with high rate of missing values that can exceed 50% as discussed in Section 3.1.

A sample imputation use case is presented here where the user is asked to: input the file that has the missing data then select the column representing the parameter that needs imputation; selecting a second column is required for the TL imputation. When done, a suitable imputation technique is applied based on the type of the dataset along with the nature and number of missing values. In this use case, the input dataset is the NDVI retrieved from Landsat Satellite with missing percentage greater than 50%, including random missing data and missing chunks. Hence, the interpolation is expected to run with transfer learning. Therefore, the NDVI parameter retrieved from MODIS Satellite is uploaded by the user as a base dataset. The results of the imputation are visible in Figure 5.3.

5.3 Forecasting Module

The forecasting module is used to enable the end user to forecast the fresh produce yield and price. The forecasting model is generalized based on the transfer learning framework shown in Figure 3.2, and this is implemented by embedding a database. The database consists of three tables: Fresh Produce Table, Model Table, and County Table. The County Table is predefined to have 229 counties which are all the available counties in US Census Counties retrieved from GEE [19]. The Entity Relationship Diagram (ERD) of the database is visible in Figure 5.4.

The forecasting module allows the user to: select the output intended to forecast whether yield or price, select the model type whether forecasting using Station based data, Satellite based data, or the combination of both defined as Hybrid, select the fresh produce name if available or enter new fresh produce, and select the state and county where the forecast should occur. One of the following cases occurs:

1. If there are no models yet in the system, then the models can be trained from scratch by asking the user to: upload a file containing the output parameter, and if the model

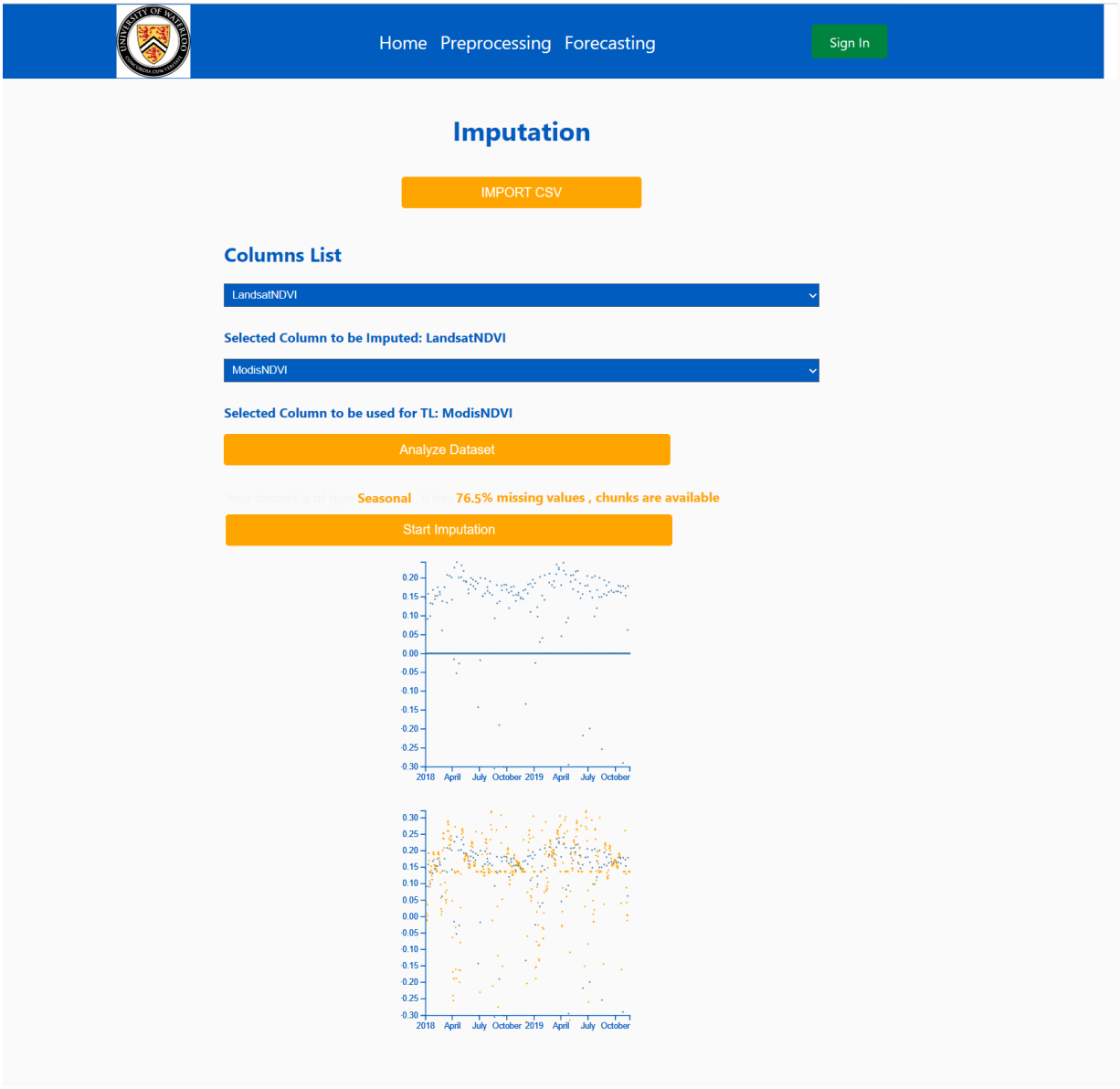

Figure 5.3: Imputation module result, the first plot shows the plot with missing values blue line in the 0, and in the second plot the result after imputation is shown where the orange dots are the imputed values

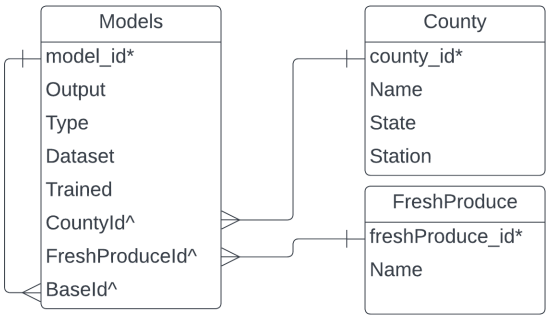

Figure 5.4: Database tables and fields ERD diagram

type is selected as station or hybrid then the station based input parameters for the selected fresh produce and its location are also needed as parameters in the file.

2. If there is a readily available model for the entered fresh produce and county, the user is asked to input the forecasting dates.

3. If the model is not available but there are other models for other fresh produce in the system then one of the following cases is considered:

   - Fresh produce found in the same location, the user is asked to input a file containing the yield of the new fresh produce and similarity is computed based on the similarity between the input parameters and yield of the fresh produces.
   - Fresh produces data is in different location, the user is asked to input a file having the yield of the new fresh produce, the input parameters are retrieved using satellite data, and then compared to find a similar time series from the available time series of base models.

   If two fresh produces are found similar finding a base model (BM) trained for the already available from produce FPB, transfer learning is applied. If no similar fresh produces found, then model is trained from scratch as a base model alone.

Three use cases are articulated for the forecasting module and discussed in subsections 5.3.1, 5.3.2 and 5.3.3.

5.3.1 Case 1: Forecasting without Available Models

In use case 1 the Hybrid model is selected, and it is assumed that no fresh produce is yet available in the system, hence the user enters the name of the fresh produce; which is Strawberry in this case. The Strawberry Santa Maria input parameters and yield are uploaded by the user, and two models are trained in the background. The reflection of this use case on the database is visible in Figure 5.5, where the blue rows are the newly created database entries of the Strawberry fresh produce, and the two models created for station and satellite forecasting. Finally, the user enters the forecasting dates; the result of forecasting after training is visible in Figure 5.6.

Model

model_id*	Output	Type	Dataset	Trained	CountyId^	FreshProduceId^	BaseId^
1	Yield	Station	StrawberrySantaBarbaraStation.csv	True	229	1	N/A
2	Yield	Satellite	StrawberrySantaBarbaraSatellite.csv	True	229	1	N/A

Fresh Produce

freshProduce_id*	Name
1	Strawberry

County

county_id*	Name	State	Station
1	St. Louis	Missouri	
. .			
194	Ventura	California	
. .			
229	Santa Barbara	California	

Figure 5.5: Database table entrees after adding a new fresh produce and creating a hybrid model.

5.3.2 Case 2: Forecasting Similar Fresh Produce

In case 2 the Station forecasting model is used for Raspberry in Santa Maria, which is found to be similar to Strawberry Santa Maria. The user input: the yield data for Raspberry, where similarity is computed and Raspberry is clustered with Strawberry Santa Maria as its base model as visible in Figure 5.7, where Raspberry Model with ID 3 has base model Strawberry with ID 1. Then the transfer learning process from Strawberry Santa Maria to Raspberry is applied then the yield is forested for the entered dates. The forecasting results are visible in Figure 5.8.

Figure 5.6: Result of use case 1 forecasting strawberry yield

Model

model_id*	Output	Type	Dataset	Trained	CountyId^	FreshProduceId^	BaseId^
1	Yield	Station	StrawberrySantaBarbaraStation.csv	True	229	1	N/A
2	Yield	Satellite	StrawberrySantaBarbaraSatellite.csv	True	229	1	N/A
3	Yield	Station	RaspberrySantaBarbaraStation.csv	True	229	2	1

Fresh Produce

freshProduce_id*	Name
1	Strawberry
2	Raspberry

County

county_id*	Name	State	Station
1	St. Louis	Missouri	
. .			
194	Ventura	California	
. .			
229	Santa Barbara	California	

Figure 5.7: Database table entrees after adding raspberry and applying transfer learning from strawberry station model

5.3.3 Case 3: Forecasting Dissimilar Fresh Produce

In case 3 the Hybrid forecasting model is used for estimating the yield of Blueberry in Ventura, which is found to be not similar to any of the available base models (assuming one model is available for Strawberry Santa Maria). The user input the yield data for Blueberry, the input parameters for Ventura are retrieved from satellite data, then the similarity is computed between Blueberry Ventura and Strawberry Santa Maria, which are found to be not similar. Hence, a new model named Blueberry Ventura is trained from scratch and labelled as a BM as visible in Figure 5.9 where the entry for Blueberry models has BaseID of N/A, and then used for forecasting the required dates as visible in Figure 5.10.

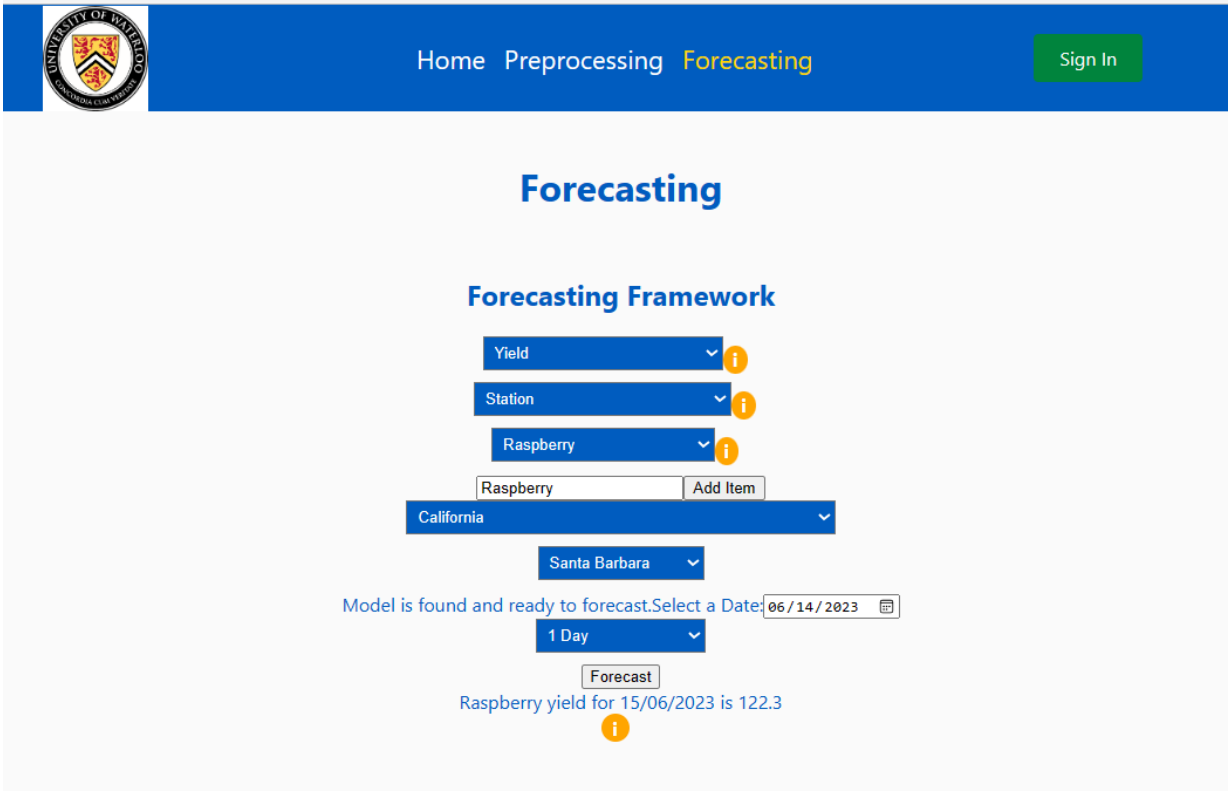

Figure 5.8: Result of use case 2 forecasting raspberry yield

Model

model_id*	Output	Type	Dataset	Trained	CountyId^	FreshProduceId^	BaseId^
1	Yield	Station	StrawberrySantaBarbaraStation.csv	True	229	1	N/A
2	Yield	Satellite	StrawberrySantaBarbaraSatellite.csv	True	229	1	N/A
3	Yield	Station	RaspberrySantaBarbaraStation.csv	True	229	2	1
4	Yield	Station	BlueberryVenturaStation.csv	True	194	3	N/A
5	Yield	Satellite	BlueberryVenturaSatellite.csv	True	194	3	N/A

Fresh Produce

freshProduce_id*	Name
1	Strawberry
2	Raspberry
3	Blueberry

County

county_id*	Name	State	Station
1	St. Louis	Missouri	
. .			
194	Ventura	California	
. .			
229	Santa Barbara	California	

Figure 5.9: Database table entrees after adding the blueberry and training the models from scratch

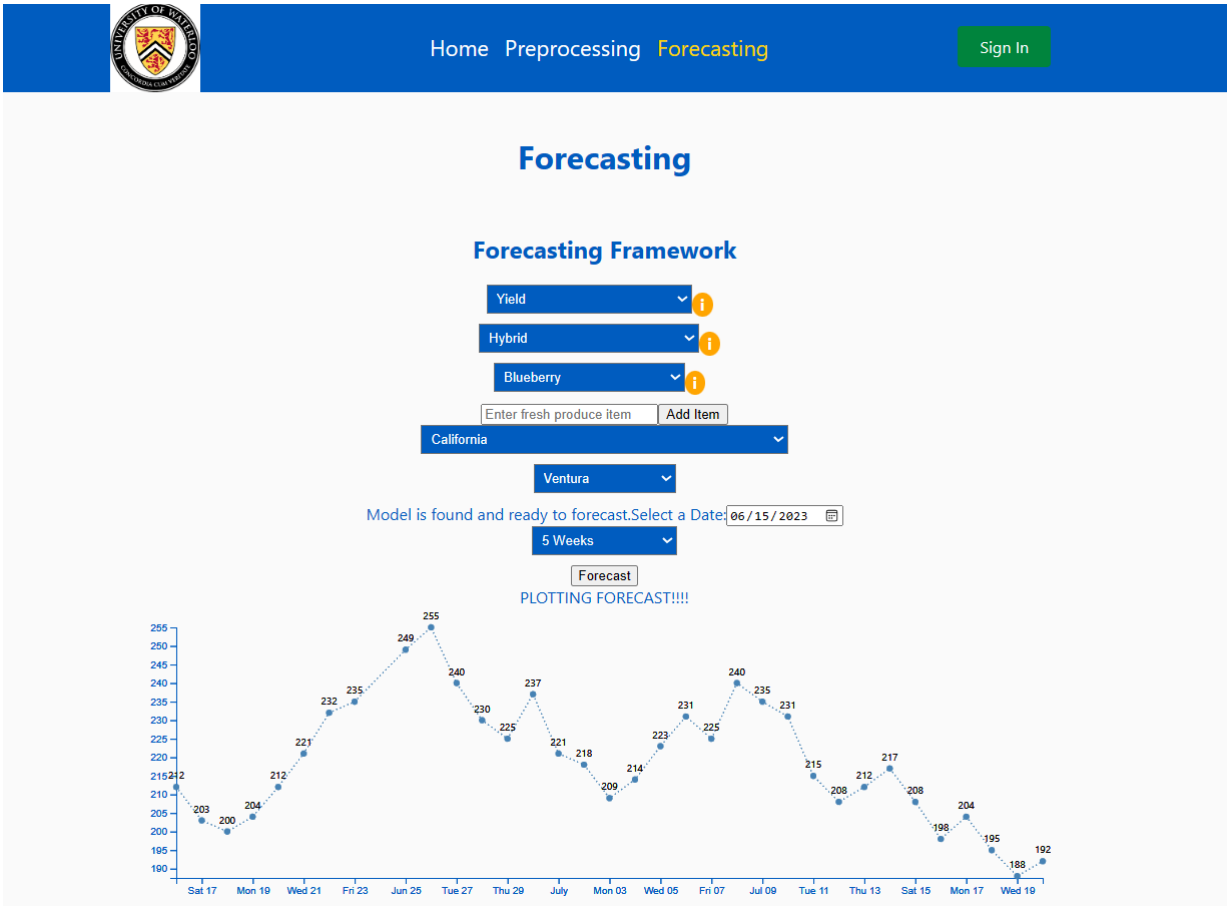

Figure 5.10: Result of use case 3 forecasting of blueberry yield

Chapter 6

Conclusion and Future Work

Accurate estimates of fresh produce yields and prices are crucial for having fair bidding prices by retailers along with informed asking prices by farmers, leading to the best prices for customers, hence the ability to forecast fresh produce accurately is presented in this thesis.

This work started by discussing the contributions to time series forecasting available in literature. This is followed by a proper background covering: The use of station and satellite-based data for FP yield and price forecasting. The available data preprocessing approaches and the deployed imputation techniques from literature for estimating the missing values found in datasets retrieved from those two sources. The available highly performing deep learning forecasting models. The time series similarity and clustering techniques. Finally, the transfer learning and quality assurance techniques for machine learning models.

A deep learning model for fresh produce forecasting is proposed in this work , where the proposed model of DFNNGRU-ADGRU Ens is found to outperform the literature yield forecasting models in [2]: by 24% AGM enhancement over the CNN-LSTM Ens Station based forecasting model and 20% over the SAE-CNNLSTM Ens satellite-based model for yield forecasting. An enhancement of 38% is found for the station-based model price forecasting. Additionally, an efficient preprocessing technique is introduced for satellite data representing each satellite image with the overall average of its pixels is found to have 12.5% improvement in AGM compared to finding the prelagged histogram which is recommended in [18]. The deployed interpolation technique, cubic spline, is found to improve the forecasting AGM by 44% compared to replication method. Finally, adding the vegetation index parameter NDVI improves the AGM up to 12.5%. However, the

NDVI is found to have a high rate of missing values that can cause a problem for the daily forecasting, therefore an interpolation technique is tailored and proposed to estimate those NDVI missing values. All the forecasting models are trained on forecasting 5 weeks ahead, however this work could be extended to other duration by changing the lag based on the best lag found using statistical methods.

The generalization is achieved through proposing and applying a transfer learning framework to the found deep learning model to facilitate its utilization with different FPs in various locations. First, a similarity technique is applied to find the percentage similarity among time series and clustering them accordingly. The similarity technique is compared with the input similarity model proposed in [67]. The results are found to be accurate with the advantage of being faster and discarding the arbitrarily decided thresholds.

The best combination of freezing, finetuning and retraining model layers is then decided to apply the transfer learning, where an AGM percentage improvement of 14% is found over retraining the model from scratch. Finally, a quality assurance method is proposed and applied to the DFNNGRU-ADGRU Ens model and the data used in training where metamorphic relations and tests relevant to the problem in hand are articulated and applied successfully. The quality assurance of the found model is done using metamorphic testing; 11 metamorphic relations with their corresponding tests are designed and executed. The results of all metamorphic tests are found to match the expected results therefore all the metamorphic relations are fulfilled by the model which assures the high quality of the model.

An application is successfully built using Flask backend and react front-end to allow users to have friendly access to the built modules: The preprocessing tab which includes the imputation and similarity modules. The deep learning models for fresh produce forecasting are covered under the forecasting tab along with the transfer learning framework.

Future work should consider: exploring the effect of transformers on time series forecasting as it is gaining popularity and researchers attention. Additionally, the comparison of creating a 1 week forecasting model with a 5 weeks ahead forecasting needs to be tested for the same expected output dates, then compared with using PCA as a reduction technique to achieve the same dimensions for different lags, based on the horizon, and comparing the results. Furthermore, improving the satellite image forecasting model by enabling it to extract the yield of any FP when given the fresh produce planting location as input from the user instead of relying on the user external sources like station-based data to get the yield. Finding a reliable imputation technique suitable for datasets with large missing data ratios that can exceed 50%. Testing the effect of using metamorphic tests in other domains different from time series forecasting. Adding a method of finding the closest

weather station to enable the automatic loading of available station data rather than taking it as an input from the user. Finally, enhancing the Google Earth Engine satellite images deployment by using batch tasks to retrieve the images rather than running all in one task, which will help in faster image retrieval, in addition to paralleling the satellite tasks on server so that all images are obtained at once.

References

[1] Allyson Fradella, "Behind the Number: What's Causing Growth in Food Prices", Statistics Canada Website, last accessed 23/05/23.

[2] M. Chaudhary, M. Gastli, L. Nassar and F. Karray, "Deep learning approaches for forecasting Strawberry yields and prices using satellite images and station-based soil parameters," AAAI-MAKE 2021.

[3] C.L. Wiegand, A.J. Richardson, D.E. Escobar, A.H. Gerbermann,"Vegetation indices in crop assessments, Remote Sensing of Environment," Volume 35, Issues 2–3, 1991, Pages 105-119.

[4] S. Panda, D. Ames, P. Suranjan. "Application of Vegetation Indices for Agricultural Crop Yield Prediction Using Neural Network Techniques." Remote Sensing, 2010.

[5] Chaudhary, M., Gastli, M. S., Nassar, L., & Karray, F. (2021, July). Transfer learning application for berries yield forecasting using deep learning. In 2021 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[6] B. Mateus, J. Farinha, R. Assis and A. Cardoso, "Comparing LSTM and GRU models to predict the condition of a Pulp paper press," Energies, 2021.

[7] M. Chaudhary, L. Nassar and F. Karray, "Deep Learning Approach for Forecasting Apple Yield using Soil Parameters," 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 2021, pp. 844-850, doi: 10.1109/SMC52423.2021.9658804.

[8] Kirthiga, S. M., & Patel, N. R. (2022). In-Season Wheat Yield Forecasting at High Resolution Using Regional Climate Model and Crop Model. AgriEngineering, 4(4), 1054–1075.

[9] A. Sagheer, H. Hamdoun and H. Youness,"Deep LSTM-Based Transfer Learning Approach for Coherent Forecasts in Hierarchical Time Series," Sensors 21(13), 2021.

[10] I. Nasr, L. Nassar, and F. Karray, "Transfer Learning Framework for Forecasting Fresh Produce Yield and Price", International Joint Conference on Neural Networks, IJCNN'22, IEEE.

[11] M. Saad, L. Nassar, F. Karray and V. Gaudet, "Tackling Imputation Across Time Series Models Using Deep Learning and Ensemble Learning," 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON, Canada, 2020, pp. 3084-3090.

[12] National Oceanic and Atmospheric Administration, "url:https://www.noaa.gov/", last visited on 2021-10-17.

[13] The California Strawberry Commission, url:"https://www.calstrawberry.com/en-us/", last visited on 2021-10-17.

[14] Lanza, L., Conti, M., "Cloud tracking using satellite data for predicting the probability of heavy rainfall events in the Mediterranean area." Surv Geophys 16, 163–181 (1995).

[15] H. V. Alvan and F. H. Azad, "Satellite remote sensing in earthquake prediction. A review," 2011 National Postgraduate Conference, 2011, pp. 1-5, doi: 10.1109/NatPC.2011.6136371.

[16] Subash S.P, Rajeev Kumar, Aditya K S. "Satellite data and machine learning tools for predicting poverty in rural India." Agricultural Economics Research Review. 31. 231-240. 2019.

[17] Sohaib Abujayyab,˙Ismail Karas¸. "Automated Prediction System for Vegetation Cover Based on MODIS-NDVI Satellite Data and Neural Networks." ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. XLII-4/W19. 9-15. 2019.

[18] M. S. Gastli, L. Nassar and F. Karray,"Satellite Images and Deep Learning Tools for Crop Yield Prediction and Price Forecasting," 2021 International Joint Conference on Neural Networks (IJCNN), 2021, pp. 1-8.

[19] Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., and Moore, R. "Google Earth Engine: Planetary-scale geospatial analysis for everyone." 2017.

[20] Qi Wang, Jiancheng Li, Taoyong Jin, Xin Chang, Yongchao Zhu, Yunwei Li, Jiaojiao Sun, and Dawei Li. “Comparative Analysis of Landsat-8, Sentinel-2, and GF-1 Data For Retrieving Soil Moisture Over Wheat Farmlands.“ Remote Sensing, 12(17):2708, pp. 2-3, Aug 2020.

[21] Xiao-Peng Song, Wenli Huang, Matthew C. Hansen, Peter Potapov, “An evaluation of Landsat, Sentinel-2, Sentinel-1 and MODIS data

[22] J. You, X. Li, M. Low, D. Lobell, and S. Ermon, “Deep Gaussian Process for Crop Yield Prediction Based on Remote Sensing Data,” in ThirtyFirst AAAI Conference on Artificial Intelligence, 2017, p. 4559–4565.

[23] “NASA-USDA global soil moisture data,” https://earth.gsfc.nasa.gov

[24] “LP DA AC MODIS land products,” http://lpdaac.usgs.gov/, accessed 16 January 2023.

[25] H. Fang, S. Liang, in “Reference Module in Earth Systems and Environmental Sciences,“ 2014.

[26] C.L. Wiegand, A.J. Richardson, D.E. Escobar, A.H. Gerbermann,“Vegetation indices in crop assessments, Remote Sensing of Environment,“ Volume 35, Issues 2–3, 1991, Pages 105-119.

[27] S. Panda, D. Ames, P. Suranjan. “Application of Vegetation Indices for Agricultural Crop Yield Prediction Using Neural Network Techniques.“ Remote Sensing, 2010.

[28] Qi Qin, Dawei Xu, Lulu Hou, Beibei Shen, and Xiaoping Xin. “Comparing Vegetation Indices from Sentinel-2 and Landsat 8 Under Different Vegetation Gradients Based on a Controlled Grazing Experiment.“ Ecological Indicators, 133:108363, 2021, pages 2.

[29] GISGeography, “Spectral Signature Cheatsheet – Spectral Bands in Remote Sensing.“ October-2021. Last accessed 25 April 2022.

[30] Irfan Pratama, Adhistya Permanasari, Igi Ardiyanto, Rini Indrayani.“A Review of Missing Values Handling Methods on Time-series Data.“ pp. 1-6. 2016.

[31] Shin-Fu Wu, Chia-Yung Chang, Shie-Jue Lee. “Time Series Forecasting with Missing Values.“ EAI Endorsed Transactions on Cognitive Communications. 2015.

[32] P. Bansal, P. Deshpande,S. Sarawagi.“Missing Value Imputation on Multidimensional Time Series.“ 2021.

[33] Schafer JL, Graham JW. “Missing data: our view of the state of the art. Psychol Methods.“ 7(2): pp. 147-77. 2002.

[34] P. Nickerson, R. Baharloo, A. Davoudi, A. Bihorac,P. Rashidi, “Comparison of Gaussian Processes Methods to Linear methods for Imputation of Sparse Physiological Time Series“. Conf Proc IEEE Eng Med Biol Soc. 2018 Jul; pp. 4106–4109.

[35] Mohamed Noor et al. “Comparison of Linear Interpolation Method and Mean Method to Replace the Missing Values in Environmental Data Set.“ Materials Science Forum. 803. pp. 278-281. 2014.

[36] C. Caruso, F. Quarta. “Interpolation Methods Comparison.“ Computers and Mathematics Applications vol. 35, pp. 109-126, June 1998.

[37] Baluev, R. V. “Assessing the Statistical Significance of Periodogram Peaks.“ Monthly Notices of the Royal Astronomical Society, 385(3), pp. 1279–1285. 2008.

[38] M. Lepot, J. Aubin, F.H. Clemens. “Interpolation in Time Series: An Introductive Overview of Existing Methods, Their Performance Criteria and Uncertainty Assessment.“ Water, 9, 796. 2017.

[39] J. Gauthier, D. Prandi. “Generalized Fourier-Bessel operator and almost-periodic interpolation and approximation.“ 2016.

[40] Murli Gupta. “Numerical Methods and Software (David Kahaner, Cleve Moler, and Stephen Nash).“ Siam Review - SIAM REV. 33. 1991.

[41] Hornbeck, H. ”Fast Cubic Spline Interpolation.” 2020.

[42] William H., Saul A., William T., Brian P. ”Numerical Recipes The Art of Scientific Computing-3rd Edition.” 2007.

[43] T. Feng, S. Narayanan, ”Imputing Missing Data In Large-Scale Multivariate Biomedical Wearable Recordings Using Bidirectional Recurrent Neural Networks With Temporal Activation Regularization.” In Proceedings of the 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2019, Berlin, Germany, 23–27 July.

[44] W., Junger, and A.P., Leon. ”Imputation of missing data in time series for air pollutants. Atmospheric Environment”, vol. 102, pp. 96-104. 2015.

[45] R. Fu, Z. Zhang and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), Wuhan, 2016, pp. 324-328, doi: 10.1109/YAC.2016.7804912.

[46] X. Yang, Y. Zhang and M. Chi, "Time-aware Subgroup Matrix Decomposition: Imputing Missing Data Using Forecasting Events," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 1524-1533, doi: 10.1109/BigData.2018.8622436.

[47] L. Shen, Q. Ma & S. Li, "End-to-End Time Series Imputation via Residual Short Paths." Proceedings of The 10th Asian Conference on Machine Learning, in PMLR vol. 95, pp. 248-263, 2018.

[48] Mack C, Su Z, Westreich D. Managing Missing Data in Patient Registries: Addendum to Registries for Evaluating Patient Outcomes: A User's Guide, Third Edition [Internet]. Rockville (MD): Agency for Healthcare Research and Quality (US); 2018 Feb. Types of Missing Data. Available from: https://www.ncbi.nlm.nih.gov/books/NBK493614/

[49] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," arXiv preprint arXiv:1704.02971, 2017.

[50] H. Verma and S. Kumar, "An accurate missing data prediction method using LSTM based deep learning for health care", Proc. 20th Int. Conf. Distrib. Comput. Netw., pp. 371-376, 2019.

[51] Y. J. Kim and M. Chi, "Temporal belief memory: Imputing missing data during RNN training", IJCAI, pp. 2326-2332, 2018.

[52] S.J. Pan, Q. Yang, "A survey on transfer learning", IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359, doi:10.1109/TKDE.2009.191.

[53] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research Vol. 12, pp. 2825–2830, 2011.

[54] Karlis Gutans, "Business Days Time Series Weekly Trend and Seasonality," Engineering Proceedings Vol.5, 2021.

[55] I. Nasr, L. Nassar and F. Karray, "Enhancing Fresh Produce Yield Forecasting Using Vegetation Indices from Satellite Images", System Man Cybernetics Conference (SMC'22).

[56] V. Ingle and S. Deshmukh, "Ensemble deep learning framework for stock market data prediction (EDLF-DP)," Global Transitions Proceedings vol. 2, pp. 47–66, 2021.

[57] S. Gomez-Rosera, M. Capretz and S. Mir, "Deep neural network for load forecasting centered on architecture evolution," IEEE ICMLA, pp. 122–129, 2020.

[58] J. Runge and R. Zmeureanu, "A review of Deep learning techniques for forecasting energy use in buildings," Energies, Vol.14, 2021.

[59] J. Newling and F. Fleuret, "K-Medoids for K-Means seeding," NIPS, 2017.

[60] C. Sammut and G. Webb, "Root Mean Squared Error and Mean Average Error," In: Encyclopedia of Machine Learning, Springer, Boston, MA, 2021.

[61] D. Chicco, M. Warrens and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation," PeerJ Computer Science, 2021.

[62] I. Okwuchi, "Machine Learning based Models for Fresh Produce Yield and Price Forecasting for Strawberry Fruit," Thesis UWSpace, 2020.

[63] A. Steland, E. Rafajlowicz and O. Okhrin, "Stochastic Models, Statistics and Their Applications," Vol. 294, Springer Proceedings in Mathematics & Statistics, pp. 408–427, 2009.

[64] F. Zhao et al., "A similarity measurement for time series and its applications in the stock market," Expert Systems with Applications Vol. 182, 2021.

[65] A. Lahreche and B. Boucheham, "A fast and accurate similarity measure for long time series classification based on local extrema and dynamic time warping," Expert System with Applications, vol. 168, 2021.

[66] Y. Xie and B. Wiltgen, "Adaptive Feature Based Dynamic Time Warping," IJCSNS, vol. 10 no.1, 2010.

[67] F. Jafari, L. Nassar, and F. Karray, "Time Series similarity analysis framework in the Fresh Produce yield forecast domain," IEEE SMC, 2021.

[68] S. Gomez-Rosero, M. Capretz and S. Mir, "Transfer Learning by similarity-centered architecture evolution for multiple residential load forecasting," Smart Cities, vol. 4, pp. 217–240, 2021.

[69] A. Saxena et al., "A review of clustering techniques and developments," Neurocomputing, vol. 267, pp. 664–681, 2017.

[70] L. Kaufman and P. Rousseeuw, "Finding groups in data: an introduction to cluster analysis," pp. 88–125, 2005.

[71] J. Newling and F. Fleuret, "K-Medoids for K-Means seeding," NIPS, 2017.

[72] H. Fawaz, G. Forestier, J. Weber, L. Idoumghar and P. Muller, "Transfer learning for time series classification," CoRR, 2018.

[73] X. Bu, J. Peng, J. Yan, T. Tan, and Z. Zhang, "GAIA: A transfer learning system of object detection that fits your needs," CVPR 2021, pp. 274–283.

[74] J. Devlin, M. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL, pp. 4171–4186, 2019.

[75] Jun Ma et al., "Transfer learning for long-interval consecutive missing values imputation without external features in air pollution time series," Advanced Engineering Informatics Vol. 44, 2020.

[76] K. Weiss, T. Koshgoftaar and D. Wang, "A survey of transfer learning," Journal of Big Data Vol. 3, 2016.

[77] S. Segura, G. Fraser, A. Sanchez, A. Ruiz-Cortes. (2016). "A Survey on Metamorphic Testing". IEEE TSE. 42. 1-1.

[78] Tsong Yueh Chen et al. 2018. "Metamorphic Testing: A Review of Challenges and Opportunities." ACM Comput. Surv. 51, 1, Article 4 (January 2019), 27 pages.

[79] Srinivasan, M, Kanewala, U. "Metamorphic relation prioritization for effective regression testing." Softw Test Verif Reliab. 2022; 32.

[80] Aravind Nair, Karl Meinke, and Sigrid Eldh. 2019. "Leveraging mutants for automatic prediction of metamorphic relations using machine learning." MaLTeSQuE 2019. Association for Computing Machinery, New York, NY, USA, 1–6.

[81] Murphy, Christian & Kaiser, Gail & Hu, Lifeng & Wu, Leon. (2008). "Properties of Machine Learning Applications for Use in Metamorphic Testing." 867-872.

[82] Sebastiao H. N. Santos, Beatriz Nogueira Carvalho da Silveira, Stevao A. Andrade, Marcio Delamaro, and Simone R. S. Souza. 2020. "An Experimental Study on Applying Metamorphic Testing in Machine Learning Applications." In Proceedings of the 5th Brazilian Symposium on Systematic and Automated Software Testing. Association for Computing Machinery, New York, NY, USA, 98–106.

[83] A. Dwarakanath, M. Ahuja, S. Podder, S. Vinu, A. Naskar and M. Koushik, "Metamorphic Testing of a Deep Learning Based Forecaster," 2019 IEEE/ACM 4th International Workshop on Metamorphic Testing (MET), 2019, pp. 40-47.

[84] Chen, T.Y., Cheung, S.C., & Yiu, S. (2020). "Metamorphic Testing: A New Approach for Generating Next Test Cases."

[85] J. Tan,J. Yang, S. Wu, G. Chen, J. Zhao. (2021). "A Critical Look at the Current Train/Test Split in Machine Learning."

[86] Marijan, Dusica & Gotlieb, Arnaud & Ahuja, Mohit. (2019). "Challenges of Testing Machine Learning Based Systems." 101-102.

[87] Frances E. Allen (July 1970). "Control flow analysis". SIGPLAN Notices. 5 (7): 1–19. doi:10.1145/390013.808479.

[88] Li, C., Liu, J., Yang, X., Yan, S., and Li, M., "Metamorphic Relation Recognition Method Based on Control Flow Graph Features", in Journal of Physics Conference Series, 2022, vol. 2219, no. 1.

[89] U. Kanewala, "Techniques for Automatic Detection of Metamorphic Relations," 2014 IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops, 2014, pp. 237-238.

[90] I. Nasr, L. Nassar, F. Karray, "Enhanced Deep Learning Satellite-based Model for Yield Forecasting and Quality Assurance Using Metamorphic Testing", International Joint Conference on Neural Networks, IEEE IJCNN'23.

[91] Upulee Kanewala, Anders Lundgren, and James M. Bieman, "Chapter Seven - Automated Metamorphic Testing of Scientific Software", Software Engineering for Science, CRC Press, pages 149-174, 2016.

[92] Janiesch, C., Zschech, P. & Heinrich, K. "Machine learning and deep learning." Electron Markets 31, 685–695 (2021).

[93] Li, C., Liu, J., Yang, X., Yan, S., and Li, M., "Metamorphic Relation Recognition Method Based on Control Flow Graph Features", in Journal of Physics Conference Series, 2022, vol. 2219, no. 1.

[94] S. V. N. Vishwanathan, Nicol N. Schraudolph, Risi Kondor, and Karsten M. Borgwardt. 2010. "Graph Kernels." J. Mach. Learn. Res. 11 (3/1/2010), 1201–1242.

[95] Urry, Matthew J. ; Sollich, Peter. "Random Walk Kernels and Learning Curves for Gaussian Process Regression on Random Graphs." In: JOURNAL OF MACHINE LEARNING RESEARCH. 2013 ; Vol. 14, No. N/A. pp. 1801-1835.

[96] Benyamin Ghojogh, Mark Crowley, Fakhri Karray, Ali Ghodsi (2023). "Elements of Dimensionality Reduction and Manifold Learning."