Encrypted Deep Learning and Agricultural Yield Forecasting

UNIVERSITY OF
LINCOLN

George Onoufriou

School of Computer Science

College of Science

University of Lincoln

Submitted in partial satisfaction of the requirements for the
Degree of Doctor of Philosophy
in Computer Science

Supervisor Dr. Georgios Leontidis
Second Supervisor Prof. Marc Hanheide

2023-02-13

Declaration

I hereby declare that the work in this thesis has not been previous or currently submitted, in whole or in part, for any other degree or professional qualification. This thesis is composed solely by myself, and the work contained herein is my own except where explicitly stated otherwise.

Preface

In the face of the greatest epidemiological disaster since the Spanish flu of 1918, we now have Coronavirus disease 2019 (COVID-19); we have seen years of unprecedented disasters, restrictions, invasions, and human rights violations/ erosion as a consequence or as a dual factor. In this time we have seen:

(i) The collapse of One-Country-Two-Systems in Hong Kong, and the introduction of the so called Hong Kong national security law, after year long protests by the Hong Kong people. (2020-06-30)

(ii) The Beirut explosion due to negligence and corruption which devastated an already fragile country defaulting on its debt, and the subsequent absence of government for roughly a year hence. A failed state. (2020-08-04)

(iii) The fall of Myanmar and its democratically elected government to a military coup. (2021-02-01)

(iv) The capture of a journalist and critic, Roman Pratasevich, on Ryanair flight 4978 from Athens to Lithuania by Belarusian president Alexander Lukashenko, via bomb threat redirection. (2021-05-23)

(v) Refugees being exploited as weapons against Poland by Belarus likely due to the sanctions of the EU against Belarus for the aforementioned journalist capture, leading to a humanitarian crisis. (2021-08)

(vi) The invasion of Ukraine by Russian forces due to the increased westernisation of Ukraine as a consequence of the previous annexation of Crimea by Russia. (2022-02-24)

This is but to name a few, notably not including climate related issues. Tensions

across the world are high, abuse/ corruption by those in power is prevalent, and errosions of human rights are common place. This is catastrophic to trust, how can we trust each other when clearly we humans are capable and willing to harm ourselves and others. This is all without the looming dangers of AI and its misuse, which requires even more trust in usually very insular environments, often with little oversight. We need (Kerckhoffian) safeguards from ourselves and others to continue to use tools like these. This work is dedicated towards in some small part furthering science, helping directly in what little ways are possible by protecting data and subsequently people. AI is an incredibly useful tool towards almost any form of data processing but is also a very dangerous tool when applied to dystopian totalitarian use cases such as social credit scores by facial recognition/ association. AI is here to stay however we need ways to privately compute intelligence, for the ethical future of AI especially in an increasingly adversarial world and for the good of all who live in our interconnected world.

Towards this end, I have so much faith in both fully homomorphic encryption and my privacy-preserving work, that I have incorporated a business to take these ideas forward and manifest them towards Kerckhoffian encrypted deep learning for all.

deepcypher.me

"Cryptography is the ultimate form of non-violent direct action."

Julian Assange

"Cypherpunks write code."

Eric Hughes

"The true measurement of a person's worth isn't what they say they believe in, but what they do in defence of those beliefs. If you're not acting on your beliefs, then they probably aren't real."

Edward Snowden

"The Holocene has ended. What we do now, and in the next few years, will profoundly effect the next few thousand years."

David Attenborough

Acknowledgements

I would like to vehemently thank my family. We have had quite a journey thus far fraught with adversity that others would and have hardly believed. You are my heroes, I hope I can make you proud.

I would like to thank my supervisors; Georgios Leontidis and Marc Hanheide for their endless support and help.

I would like to thank my alma mater, for the freedoms and opportunities with which to grow.

Lastly I would like to thank the whole computer science, machine learning, privacy, homelab, and 3d printing communities for being so welcoming and helping me sustain this long PhD with fun and intrigue beyond purely the PhD.

Publications

George Onoufriou, Ronald Bickerton et al. (2019). 'Nemesyst: A hybrid parallelism deep learning-based framework applied for internet of things enabled food retailing refrigeration systems'. In: Computers in Industry 113, p. 103133

George Onoufriou, Marc Hanheide and Georgios Leontidis (2020a). 'The Augmented Agronomist Pipeline and Time Series Forecasting'. In: UKRAS 2020

George Onoufriou, Paul Mayfield and Georgios Leontidis (2021b). 'Fully homomorphically encrypted deep learning as a service'. In: Machine Learning and Knowledge Extraction 3.4, pp. 819–834

George Onoufriou, Marc Hanheide and Georgios Leontidis (2022a). 'EDLaaS: Fully Homomorphic Encryption over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting'. In: Sensors 22.21, p. 8124

George Onoufriou, Marc Hanheide and Georgios Leontidis (2022b). 'Premonition Net, A Multi-Timeline Transformer Network Architecture Towards Strawberry Tabletop Yield Forecasting'. In: arXiv preprint arXiv:2211.08177

Abstract

Food security is declining in the UK, and food insecurity is on the rise, even before the COVID-19 pandemic which only exasperated the issue (Pool and Dooris, 2021). This is largely due to global weather instability and lack of resources which has impeded our ability to forecast crop yields accurately. Subsequently this affects our ability plan around the diminishing availability of labour, and negotiate purchase orders ahead of time. One of the most difficult aspects of crops like strawberries is that they are highly perishable, and cannot be push-pulled like other products such as meats. Meats can be harvested earlier or later according to demand, strawberries in contrast will rot on the stem. Any overripe berries also contribute to the prevalence of pests like wasps which primarily feast on the overripe berries. Any shortfall in berries, such as from adverse weather, will likely have to be sourced overseas where the weather conditions are different, this is expensive and results in inferior produce due to the time delay importing has on the freshness of the berries. This naturally increases prices of fresh produce (FP) as the likelihood of this expensive scenario means prices must be passed on to the consumers. All of these considerations make it so incredibly important that we can forecast several weeks ahead, accurately, and in a low cost manner. This is where we believe, scarcely-applied, advanced machine learning (ML) forms of deep learning (DL) come in, to help accurately forecast yields to meet this need. In the pursuit of accurate and reliable ML the issue of data availability is ever present. Growers see little to no benefit to the arduous task of collecting such data, and of what data they do collect they are highly concerned with losing their competitive advantage. The advancement of fully homomorphic encryption (FHE) and deep learning we call *encrypted deep learning (EDL)*. *EDL* in this thesis serves to mitigate such concerns, encourage data sharing, and to open up more collaboration possibilities, towards a sustainable future for DL.

List of Terms

abelian We use this as another way of saying commutative. An abelian and a commutative group (ring) is such when any element in that group can be added or multiplied together to create another member in that group. I.E the set of integers $\mathbb{Z}$ is a commutative / abelian group since any two integers added or multiplied always results in another integer. Abelian compatibility is the conformity of some operation to this commutation.

certainty In the context of machine learning, certainty is the level of confidence in that some output will accurately represent the ground truth. Certainty can be expressed as a percentage or as a decimal in the range $[0, 1]$ representing the likelihood that some output is accurate to within some confidence interval / band.

complex Any number that is composed of a real and imaginary numbers.

cyphertext The output of some form of encryption, some form of encoded/ encrypted information that is unreadable without a proper key for decryption.

deep learning A subset of a broader family of techniques in machine learning that are specifically concerned with learning distributions using neural networks.

deep water culture A hydroponic technique which involves growing plants ina nutrient-rich water solution, without the use of soil. Roots of the plants are suspended directly into oxygenated water, to supply them with the necessary oxygen.

encrypted deep learning The process of using FHE encrypted data in compatible

/ abelian-based neural networks from end-to-end such that the neural networks have no context as to what they are processing.

encrypted deep learning as a service The a service model where encrypted deep learning is provided.

food insecurity The disruption of food intake or eating patterns due to lack of money or other resources.

food security A situation where all people, at all times, have physical, social and economic access to sufficient, safe and nutritious food that meets their dietary needs and food preferences for an active and healthy life.

general data protection Act A European Union law governing personal data collection, processing and use. It is intended to give individuals more control over their personal information and create sets of strict rules for businesses and other organisations to adhere to.

homomorphic A mathematical principle of a transformation which occurs on a set, whereby the second set maintains the operablility of the original set. Like addition and multiplication.

imaginary Any number multiplied by the imaginary unit i where $i = \sqrt{-1}$.

integer An integer in mathematics is any whole number. -200, 0, 1, 2, 3, 4, 5 are all integers. 2.45 is not an integer. The set of integers is every single whole number in existence.

lattice-based cryptography A form of cryptography that uses lattices for asymmetric crypographic primitives. It is thought to be quantum-decryption resistant since it does not rely on factorisation (RSA) or discrete logs (elliptic curve).

learning with errors A computational problem believed to be hard, or computationally infeasible, for certain types of algorithms and machines to solve. The

sub-problems are used as the basis for multiple cryprographic systems including FHE and latice-based cryptography.

loss A measure of how well a machine learning model is able to make predictions on a specific dataset. Loss is typically calculated by comparing the model predictions to the true / observed values in the dataset, and then summing the errors across all examples in the dataset. The goal of training a ML model is to minimise the loss, so that the model can make accurate predictions on new unseen data that was not used to train the ML model. There are various types of loss functions that can be used depending on the task, such as mean squared error for regression tasks and cross-entropy loss for classification tasks.

machine learning Algorithms that can automatically improve through the use of data and statistics.

modulo A mathematical operation which finds the remainder when one integer is divided by another. This is an extremely important operation in cryptography to encode information.

neural network A neural network is a ML model loosely inspired by the structure and function of human neurons. It consists of layers of interconnected "neurons," which process and transmit information. Neural networks can learn to perform a variety of tasks by adjusting the strengths of the connections between neurons, known as weights. They are capable of learning to recognise patterns, classify data, and make decisions based on input data.

nutrient film technique A hydroponic technique which involves continuously coating the roots in a small nutrient film. Nutrient solution is stored in a reservoir at the lowest point, and pumped up to the highest point to allow gravity to pull the water down thought the root system.

polynomial An expression consisting of variables and coefficients, that involves only the operations of addition, subtraction, and multiplication, and does not involve any division by variables. Polynomials can have constants and variables

with positive integer exponents. For example, $3x^2 + 2x - 4$ is a polynomial. The highest exponent in a polynomial is called the degree of the polynomial. The degree of the polynomial above is 2, since it is the highest exponent in the expression.

rational Any number that can be represented in fractional form $\frac{a}{b}$. where both a and b are non-zero integers

rational agent An entity which always aims to perform optimally in its decisions based on currently known information.

real A non-imaginary, non-infinite number.

ring learning with errors A variant of the learning with errors (LWE) problem. LWE uses a vector as its secret key, wheras ring learning with errors (RLWE) uses a polynomial representation. The polynomials are defined over a ring, and thus allow for additiona and multiplication of the polynomials. This makes computations more flexible and efficient for cryptographic uses.

set A set as per set-theory is a collection of elements. These elements can be of any mathematical form, but are usually related. For example Integers are a well known set.

subset A collection of elements that belong to a larger set. The subset can contain any number of elements, including zero elements (the empty set), or all of the elements in the larger set. Every element in the subset must also be an element of the larger set.

superset A set that contains all of the elements of another set, as well as potentially additional elements. If a set A is a subset of a set B, then B is a superset of A. For example, if set A is the set of all positive *even* integers, and set B is the set of all positive integers, then set B is a superset of set A. Every element in set A is also an element of set B, but set B has additional elements that are not in set A.

uncertainty In the context of machine learning, uncertainty is the opposite / complement to certainty. $1 - \textit{certainty} = \textit{uncertainty}$.

zero-knowledge proof A method in which a prover can prove to a verifier that a given statement is true without conveying any additional information other than that the statement is indeed true, including not sharing the information itself.

List of Acronyms

ADAM adaptive moment estimation

ANN artificial neural network

BGG Berry Gardens Growers

CCE categorical cross-entropy

CKKS Cheon, Kim, Kim, and Song

CNN convolutional neural network

COVID-19 Coronavirus disease 2019

CSV comma-seperated values

CTP collaborative training partnership

DL deep learning

DWC deep water culture

EDL encrypted deep learning

EDLaaS encrypted deep learning as a service

FHE fully homomorphic encryption

FP fresh produce

GDPR general data protection Act

GPT general-purpose transformer

GPU graphics processing unit

LTSF long-term time series forecasting

LWE learning with errors

ML machine learning

MS-EVA Microsoft encrypted vector arithmetic

MS-SEAL Microsoft simple encrypted arithmetic library

MSE mean squared error

NFT nutrient film technique

NN neural network

PP privacy-preserving

PPDL privacy-preserving deep learning

PPML privacy-preserving machine learning

PPT privacy-preserving technology

ReLU rectified linear unit

RLWE ring learning with errors

ROS robot operating system

SARS-CoV-2 severe acute respiratory syndrome Coronavirus 2

TSF time series forecasting

UoL University of Lincoln

ZKP zero-knowledge proof

List of Tables

3.1 Seasonal data collection outcomes on the Riseholme strawberry tabletop. This table shows how the data collection varied between seasons as better techniques were found. 2022 data, due to changes in management, is only partially available to us and our use. 33

3.2 Time series forecasting of yield by number of punnets from the original 2019 dataset. 35

4.1 Table of predictive results of the constellation network (Figure: 4.11) predicting strawberry yield. 66

5.1 Expected errors by forecasting source. All models are from our previous work trialling different methods on the same dataset. † : These are estimates and may not be representative of any grower or agronomist specifically but are instead ballpark figures for illustration based on our information from our industry partners. 86

List of Figures

2.1 Non-linear, errored, learning problem, where g (blue), s (red), e (green) are randomly generated with a modulus of p we can further calculate t (orange) based on these prior generated values also with a modulus of p. This serves as the base for LWE based problems. 12

3.1 RASberry data distribution and aggregation pipeline, consisting of robot operating system (ROS), edge, database, and deep learning layers. The ROS layer is responsible for robot control. The edge layer is for edge compute and data capture. The DB layer is for aggregating data bateween multiple sites and Thorvalds. The back-end layer is for scaleable ML over all aggregated data (Onoufriou, 2019). 20

3.2 Strawberry seedling post transplantation into a grow bag cut-out. This grow bag sits inside a hydroponic basin that connects a row of grow bags end-to-end under a polytunnel. 24

3.3 Riseholme strawberry tabletop site. Depicting a Thorvald robot (running Ubuntu 18 at the time) with our initially mounted sensors and cameras. This image also shows some of the stationary cameras on a tripod, the other stationary camera cannot be seen as it is in the opposite tunnel. 26

3.4 2021 strawberry yields 7 day rolling average over ISO day of the year for the years 2020 and 2021. This graph overlays the two seasons to show the similarity between the two seasons and how the historic performance of yield is very indicative of subsequent years of yields where the difference is primarily dictated by mitigating or unusual conditions. 27

3.5 Rip, overripe, and underripe strawberries in the Riseholme tabletop site. This depicts an overripe strawberry (slightly purple tint) being eaten by a wasp which was and is a significant pest for strawberries when they are left on the the stem for too long. 28

3.6 2021 strawberry data line plot with temperatures, humidity, yield, soil temperature, and wind speed, over ISO days. This image serves to visually depict how the core strawberry forecasting data varies over any given season. 29
3.7 2020, and 2021 strawberry soil temperature averages (in degrees Celsius), showing how the soil (grow bags) temperature changes over the year by ISO date. 30
3.8 2021 strawberry yields by strawberry variety; Katerina and Zara. Both of which are industrial varieties. Interestingly we are informed that Zara is a very popular variety in spite of its smaller yield outputs in our Riseholme site. This may indicate that either in industrial scenarios a proper analysis is not done when selecting varieties, or that our findings here are not properly representative of these larger industrial sites. 36
3.9 2021 and 2020 strawberry yields and temperatures per ISO day, showing the relation between variety, temperature, and yields. This also shows how erratic conditions can be and how much other factors may also contribute. 36

4.1 Trends of privacy (red), Edward Snowden (orange), and Cambridge Analytica topics (green) on Google trends since 2010 showing a slow but steady increase in the interest of privacy, and particular peaks around events such as the Cambridge Analytica scandal and smaller peaks roughly correlated to Julian Assange. (Google, 2021) 39
4.2 Overview of distinct FHE cyphertext stages in computation and properties (Onoufriou, 2021). 44
4.3 Example automatic FHE parameterisation problem, over a multi-directed graph. Sources are where data becomes a cyphertext. Sinks are where cyphertexts become plaintexts. Computation nodes are generic nodes that represent some operation that can be applied to both cyphertexts and plaintexts. Explicit rotation nodes are where a cyphertexts keys are rotated, either to refresh them, or to change the form of the cyphertext, potentially into multiple smaller cyphertexts. Please note this does not necessarily follow the colour coding of our other automatically-generated graphs (Onoufriou, 2021). 55
4.4 Fashion-MNIST sample showing examples of data such as: boots, bags, jumpers, and trousers (Xiao, Rasul and Vollgraf, 2017). 56

4.5 Fashion-MNIST computational graph we call "sphira", showing the colour coded graph and the respective nodes used to train/ compute Fashion-MNIST using our neuronal-firing algorithm. Blue represents the input and input transformation circuit that deals with passing the signals into the neural network in a way it is expecting them. Yellow represents the convolutional neural network components where one filter neuron passes multiple output cyphertexts to a plethora of summing nodes. Pink represents the fully connected dense layer for each class. Purple represents the loss calculation circuit necessary for backpropagation. Orange represents the output/ prediction circuit. Red represents the generic glue operations necessary to bind components together. Green represents the encryption specific nodes like decryption, rotation, encryption. An interactive version of this graph is available in our source code documentation so that clusters of nodes can be peeled apart for investigating individual nodes and connections. (Onoufriou, 2021) 57

4.6 Encrypted convolutional neural network (CNN), this is a particular unusual implementation since there can be no summing of the filters, and instead this sum is commuted in the case where the filter operates on an input that is a single cyphertext (i.e. not a composite of multiple cyphertexts). Please see our documentation for closer detail (Onoufriou, 2021). 58

4.7 Merged mask and kernel together to create a single sparse kernel which zeros undesired components in the cyphertexts polynomial of values using Hadmard products. Please see our documentation for closer detail (Onoufriou, 2021). 60

4.8 Encrypted variant of an artificial neural network (ANN)/ dense neural network, usually used in our case to merge divergent times/ branches/ filters back together into a single output. Please see our documentation for closer detail (Onoufriou, 2021). 61

4.9 Model performance using different activation functions in the sphira network on the fashion-MNIST dataset. All activations here are their FHE compatible approximations unless otherwise specified. Each dot is a different network, or the same network with a different data type(cyphertext, plaintext) (Onoufriou, 2021). 64

4.10 Model inference time, by different types. Plaintext types mean where the graph is run using plaintext data. Cyphertext types mean where the graph is run using cyphertext data. Both plaintext and cyphertext data conforms to the same NumPy API, meaning they can be used interchangeably. Each dot is a different network (i.e. differently initialised weights but the same structure), or the same network with a different data type (cyphertext, plaintext) (Onoufriou, 2021). 64
4.11 Strawberry yield/ regression computational graph we call 'constellation', showing the colour coded graph representation and nodes used to train on strawberry yield, based on environmental factors. Blue are input/ encryption nodes. Yellow are convolutional-related nodes. Green are operational nodes necessary to "glue" the network together. Pink are dense/ ANN nodes. Orange is the output prediction node. Red is the loss calculation node. Purple is an FHE specific node used for decryption of the input data. Please see our documentation for closer detail (Onoufriou, 2021). 65

5.1 Past (purple-pink), present (blue) and premonition (yellow) timelines/ windows overlaid on a depiction / rough reference of strawberry yields through the years of 2020 and 2021 along with temperature. Depicting the point of prediction relative to (at the seam of) horizon and history. 75
5.2 Seven day rolling average line-plot of the strawberry yields of both the 2020 and 2021 seasons. 79
5.3 Yield performance of the Katerina and Zara strawberry varieties over the 2021 growing season. 80
5.4 a) Mutil Timeline Transformer (MTT) architecture wherebye three single transformers that each process different data streams, are merge by a learned dense layer to weight their significance. b) A full single transformer architecture comprised of fixed positional encoding, encoder, decoder, and linear layers notably missing Softmax. c) Multi-head attention mechanism with query, key, and value matrices. This is a sub-components of transformer encoder and decoders with optional masks to maintain the temporal blindness when processing all the data simultaneously. d) Scaled dot-product attention showing the various matrix operations necessary to compute. this is a sub component of multi-head attention. 83

5.5 Three timeline transformer loss training, validation and testing sets, per epoch of training. Beyond 62 epochs (pink vertical line) validation and testing loss steeply increases again. 85
5.6 Ordered forecasts of single MTT compared to ground truth with a horizon of 3 weeks and a history of 12 weeks. 87

Table of Contents

List of terms **xiv**

List of Acronyms **xix**

1 Topic Introduction **1**

1.1 Overview 1
1.2 Aims and Objectives 4
1.3 Thesis Structure 5
1.4 Contributions 6
1.5 Knowledge Dissemination Events 9

2 Background **10**

2.1 Rings and Fields 10
2.2 Learning With Errors 12
2.3 Ring Learning With Errors 13
2.4 Deep Learning 14
2.5 Privacy-Preserving Machine Learning 15
2.6 Food Security 16
2.7 Agricultural Supply Chains 16
2.8 Yield Forecasting 17

3 Data Collection and Data Pipelines **19**

3.1 Introduction 19
3.2 Contributions 20
3.3 Background 21
3.4 Material and Methods 25
3.4.1 Third-Party Data 25
3.4.2 Riseholme Data 26
Yield Data 26
Environmental Data 29
Hydroponics and Bag Data 30

Image Data . . . 31
3.4.3 Data Aggregation . . . 31
3.5 Results . . . 32
3.6 Discussion . . . 35
3.7 Conclusions . . . 37

4 Privacy Preservation and Fully Homomorphic Encryption 38
4.1 Introduction . . . 38
4.2 Contributions . . . 41
4.3 Related Work . . . 42
4.3.1 FHE Background . . . 42
4.3.2 Related Works . . . 44
Encrypted Deep Learning . . . 44
FHE Graph Parameterisation . . . 45
4.3.3 Threat Model . . . 46
4.4 Basic Concepts . . . 46
4.5 Material and Methods . . . 47
4.5.1 FHE parameterisation . . . 50
4.5.2 Open Data Fashion-MNIST . . . 56
Data Wrangling and Inputs . . . 58
CNN . . . 58
Dense/ ANN . . . 62
Prediction . . . 62
Loss . . . 62
4.5.3 Strawberry Yield Data . . . 63
4.5.4 Equations . . . 66
Sigmoid . . . 66
ReLU . . . 67
Sigmoid-Approximate . . . 67
ReLU-Approximate . . . 67
Sigmoid-Derivative . . . 67
ReLU-Derivative . . . 67
Sigmoid-Approximate-Derivative . . . 67
ReLU-Approximate-Derivative . . . 67
4.6 Results . . . 67
4.7 Discussion . . . 70
4.8 Conclusions . . . 71

5 Yield Forecasting and Premonition **73**
5.1 Introduction . . . 73
5.2 Contributions . . . 75
5.3 Related Work . . . 76
5.4 Material and Methods . . . 79
5.4.1 Data Wrangling . . . 80
5.4.2 Architecture . . . 82
Encoder and Decoder . . . 82
Dense . . . 83
Weight Initialisation . . . 84
Loss Function . . . 84
5.4.3 Models . . . 85
5.5 Results . . . 86
5.6 Discussion . . . 87
5.7 Conclusions . . . 90

6 Conclusions **91**
6.1 Limitations . . . 93
6.2 Implications Insights and Future Perspectives . . . 94
6.2.1 Yield Forecasting . . . 94
6.2.2 Deep Learning . . . 95
6.2.3 Fully Homomorphic Encryption . . . 96
6.3 Funding . . . 97
6.4 Future Beyond PhD . . . 97

Chapter 1

Topic Introduction

1.1 Overview

Privacy. *As sure as day follows night, knowledge follows data, and with knowledge comes understanding.* This chain of data, knowledge, then understanding is the central maxim of this PhD. This is true for computing as much as it is for any science, and daily life. Knowledge is formed from an accruement of data, and with this knowledge we can begin to hypothesise, rationalise, and come to an understanding. We might observe that an apple has fallen from a tree, then over time we come to know or even expect that they do so after observing countless numbers of apples under and falling from trees. Then over a longer period of time with more observations and seeking to find conflicting observations do we begin to understand the *why*, that the apple falls from the tree due to a force. So too do machines accrue knowledge from data, to model the world, which is called machine learning (ML). Privacy is a choice, or more specifically the ability to choose with whom to share knowledge and understanding, and by extension data. Clearly this choice is directly related to trust, as to be trusted is a prerequisite to be chosen.

Trust, A critical component in the day to day functioning of society. We trust that our fiat currency still holds value when we go to exchange it for goods and fresh produce (FP) at the store. We trust that the store with which we purchase our FP continues to do so, with enough availability for us. We trust that companies know

enough about us to enable us to purchase and use their services, but not so much or for so long as to become potentially harmful to us.

To function in our modern society is to require trust, but what is trust? Is it some belief in the truth of something, is it the hope or contingency of some state or outcome, maybe it is some charge with which one entrusts with confidence. For our purposes here trust shall mean; the confidence in the reliability, accuracy, and truth of some process. We narrow trust to the scope of processes as trust as a whole is too broad for the scope of this PhD.

Sometimes it is possible and even necessary to engage with a yet untrusted party, in particular with first engagements. Usually this is in some limited capacity with heightened observably such that any problems of concern or damages that do arise are minimal and can be quickly remedied. We refer to that here as trust building. Usually trust building takes a long period of time, unless concrete proofs can be provided that can expedite the process by convincing another party of some processes steadfast reliability. ML models and in particular the more advanced subset of deep learning (DL) models do not provide such concrete proofs, it is difficult to inspect their inner workings and indeed their understandings. Even with verifiable proofs it can still be difficult to convince parties which could be harmed by the act of trusting to trust in a new process, as in our primitive nature we are resistant to change. However a lack of change, variety, and growth as we know leads to extinction, so to would inflexible industries and ventures that do not change with the times.

There are many cases where trust building is not readily possible, meaning it is difficult to penetrate and innovate. Areas which resist trust building are areas where high sensitivity is a barrier. Good examples of highly sensitive areas include but are of course not limited to: Medicine and its highly sensitive patient data, military and its highly sensitive operational data, commercial trade secrets and their highly sensitive processes. One area we keenly focus on during this PhD is the aforementioned FP, as it is both vital for any state but also a highly competitive domain, not necessarily amongst rational agents.

Furthermore this PhD focuses on a subset of agricultural produce, namely strawberry

produce. Strawberry produce was chosen due to the local availability / constraints of this research taking place in Lincoln, and also due to existing relationships and projects ongoing between the University of Lincoln (UoL) and Berry Gardens Growers (BGG) in the form of collaborative training partnership (CTP) (see Section 6.3). We have thus been given access to both agronomist expertise and operational information which was critical for our understanding so that we could conceptualise the issues, outcomes, and constraints properly towards producing accurate yield forecasting weeks ahead of the yields. This is to enable the various stages of the FP supply chain (Section 2.7) to operate. Furthermore due to a plethora of projects with overlapping requirements we also gained access to a research centre in Riseholme adjacent to Lincoln which grows strawberries in industry like polytunnel tabletop conditions. This is important as stakeholders are difficult to contact and even less willing to share data, due to a myriad of real and perceived sensitivity reasoning, not least of which are complex webs of contractual obligations that often restrict the free movement of data needed to generate impactful forecasting models. Our Riseholme strawberry tabletop allows us to avoid such issues, by allowing us to generate our own real data, while also providing us with an extremely rare and fruitful opportunity to create real models from local data sources rather than the remote sensing datasets typically used. This funding was put in place due to real needs and problems that stakeholders are facing. They are struggling to forecast yields ahead of time, and this is having consequences to food waste, price increases, and increased carbon emissions. To solve these problems we propose new and encrypted forms of DL using fully homomorphic encryption (FHE) as encrypted deep learning (EDL) for agricultural yield forecasting, solving different and difficult problems such as privacy, trust, and performance. In tern, with better more private forecasting we can optimise and reduce food waste, carbon emissions, and improve prices, all while increasing the availability of data with which to create better forecasting models.

1.2 Aims and Objectives

The aim of this PhD work is:

To provide automated agronomy support for agronomists at scale using machine/ deep learning techniques for yield prediction, to minimize costs, and maximize specialist human time in areas that require the most attention, from high dimensional spatio-temporal data. Including providing reasonable security to protect both the data owner, and neural networks.

Over the course of the PhD to achieve this aim the following milestones / goals were outlined:

(i) **Create an autonomous data collection system**, to make such an augmented agronomist possible as in the aim. Hand collecting data at scale would be infeasible due to both time and cost investments being too high while also providing inconsistent results, meaning we need to create some form of repeatable and autonomous data collection platform so that we can collect our spacio-temporal data for yield, and uncertainty prediction, consistently and at some scale.

(ii) **Create a data aggregation, and utilization pipeline** to be able to handle distributed autonomous data collections, since this will be the most likely scenario in practice.

(iii) **Deploy an agronomy assistive ML model to predict plant yield ahead of time**, such that deviances can indicate an area of interest/ concern for both the Thorvald and the agronomists.

(iv) **Assess viability of privacy-preserving machine learning (PPML)** , to improve the security, and privacy of the data if the system were to be used in an industrial setting. In particular this could include investigating FHE encrypted data in this system, which can ensure data security being both quantum resistant, and encrypted-during-computation.

1.3 Thesis Structure

This thesis is organised broadly with connective tissue that serves to frame the broad problem and roughly chronologically ordered sub-topics that are necessary stepping stones to achieving this projects aims. As such chapters 3, 4, 5 are relatively self contained derivatives from existing publications as per the following:

(i) A top level hierarchy that serves as the primary connective tissues between all sub-topics. This includes this very text and the broader related works.

(ii) The earliest and first sub-topic pertaining to data, and data pipelines. This sub-topic is an aggressively expanded form of two of our published papers, where we sought to accomplish our first two goals around data (Section: 1.2). This is due to their timing with respect to the PhD meaning they do not have many of the improvements we have since made to our techniques:

   (a) The Augmented Agronomist Pipeline and Time Series Forecasting (Onoufriou, Hanheide and Leontidis, 2020a)

   (b) Nemesyst: A hybrid parallelism deep learning-based framework applied for internet of things enabled food retailing refrigeration systems (Onoufriou, Bickerton et al., 2019)

(iii) The cornerstone sub-topic of privacy preservation using FHE. This sub-topic is an expanded form of a further two of our published papers. This primarily pertains to the most tricky goal 4, towards assessing the viability of privacy-preserving neural networks.

   (a) EDLaaS:Fully Homomorphic Encryption over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting (Onoufriou, Hanheide and Leontidis, 2022a)

   (b) Fully Homomorphically Encrypted Deep Learning as a Service (Onoufriou, Mayfield and Leontidis, 2021b)

(iv) The final sub-topic which of yields, and yield forecasting using various neural network techniques from traditional RNNs to multi-timeline transformers. This

sub-topic is comprised of multiple works and papers over the full length of the PhD. This sub-topic seeks to tackle goal 3.

(a) Premonition Net, A Multi-Timeline Transformer Network Architecture Towards Strawberry Tabletop Yield Forecasting (Onoufriou, Hanheide and Leontidis, 2022b)

(b) The Augmented Agronomist Pipeline and Time Series Forecasting (Onoufriou, Hanheide and Leontidis, 2020a)

1.4 Contributions

The methods, material, and ideas presented in chapter 3 is an extension of published works by the thesis' author (Onoufriou, Hanheide and Leontidis, 2020a; Onoufriou, Bickerton et al., 2019).

(i) We conceptualise, implement, define, and exploit a novel data acquisition pipeline for strawberry tabletop from mixed data sources, including from robotic traversal, stationary cameras, environment sensors, weather vanes, and irrigation / hydroponic related data. This improves data quantity through increased data acquisition sources and their frequency. This improves data quality through the repetitiveness of robotic traversal, and automated sensor collections over a known site. This improves speed, and reduces human time necessary at the human acquisition stage, and thus reduces cost.

(ii) Pairing with our data acquisition pipeline we also provide a unified data aggregation and utilisation pipeline, to stream the acquired data, where it is needed, in the format it is needed in, in a near-real-time manner. This reduces the time from observation to forecast, as neural networks can attain the data they need in near-real-time. This affords shareholders more time to plan, organise, and negotiate the labour, contracts, and logistics necessary to get as much of the crop yields to market as possible for the best price. This also allows many more varied approaches to be taken since the data is very easily

and ephemerally tansformable for different neural network tasks, as it is being streamed to neural networks (NNs).

The methods, material, and ideas presented in chapter 4 is an extension of published works by the thesis' author (Onoufriou, Hanheide and Leontidis, 2022a; Onoufriou, Mayfield and Leontidis, 2021b).

(i) We propose a new block-level automatic cyphertext parameterisation algorithm, which we call autoFHE. We also seek to showcase autoFHE in both regression and classification networks, which still appears to be a misunderstood and ongoing problem (Falcetta and Roveri, 2022).

(ii) We provide and showcase open-source encrypted deep learning with a reproducible step-by-step example on an open dataset, in this case Fashion-MNIST, achieved through a dockerised Jupyter-lab container, such that others can readily and easily explore FHE with DL and verify our results.

(iii) We show a new application for encrypted deep learning to a confidential real-world dataset. This can be used in conjunction with our open example dataset to evaluate the performance of EDL when applied to various tasks in classification and regression.

(iv) We demonstrate how neuronal firing in multi-directed graphs can be achieved in our different approach. This neuronal firing algorithm is very different to standard NN approaches since it has to account for computational depth experienced by cyphertexts allowing us to go deeper, faster, and with more certainty in the integrity of the cyphertexts.

(v) We show and detail precisely the computational graph of how a convolutional neural network (CNN) can be constructed using FHE in particular how handling of the sum-of-products can occur. This along with our easily reproduced example, should help clarify many otherwise omitted details from previous works that hinder their application by new researchers to this new field.

(vi) We show recent advancements in FHE compatibility like acrrelu approximations in greater detail along with problems/ considerations as part of a whole

computational graph. We also backpropogate the dynamically approximate range of rectified linear unit (ReLU). With ReLU we are much more able to approximate current research results which also use this same, extremely popular, activation function.

The methods, material, and ideas presented in chapter 5 is an extension of published works by the thesis' author (Onoufriou, Hanheide and Leontidis, 2022b).

(i) We propose a new multi-timeline transformer NN architecture, towards forecasting over multiple growing seasons with varying contexts for the past, present, and the premonition of the future. Our method allows a transformer to model the relationship between what we have seen before, what we have seen in the current season, and what we expect to see in the future given our current understanding. This reduces the start-of-season forecasting issues and improves season-wide performance significantly when compared to previously published techniques.

(ii) We provide several solutions and techniques necessary to overcome real world problems in out data. This includes skipping windows, resampling for synchronisation, and detailed training and architectural decision making. Our techniques allow complex transformers such as our multi-timeline transformers to ingest data regardless of inevitably varying picking schedule and quality, between seasons where data may not align. This expands the repertoire of applicable data, to allow for deeper training of more complex networks.

(iii) We apply our methods to a real functioning strawberry tabletop site, which suffers from various issues such as pests, labour shortages. This provides a real world baseline for future comparison, albeit on a smaller site, more intensive site.

1.5 Knowledge Dissemination Events

Following are several of the events attended and presented, whereby knowledge on various topics of this PhD have been disseminated by the PhD.

(i) Conference: Internet of Food Things 2019 (2019-09-17)

(ii) Event: Collaborative training partnership knowledge dissemination events (2022-07-19, 2021-11-03, 2020-11-25, 2019-10-30)

(iii) Paper: Nemesyst: A Hybrid Parallelism Deep Learning Framework (2019-12)

(iv) Paper: The Augmented Agronomist Pipeline and Time Series Forecasting (2020-05-06)

(v) Conference: New Scientist Live Presentation: Future of Food and Agriculture (2020-11-28)

(vi) Conference: Internet of Food Things Network Conference 2021 (2021-03-01)

(vii) Event: FHE.org: Running Numpy Programs Homomorphically (2021-09-30)

(viii) Paper: Fully Homomorphically Encrypted Deep Learning as a Service (2021-10-13)

(ix) Lecture: Guest Lecture: AI Containerisation (2021-11-16)

(x) Paper: EDLaaS:Fully Homomorphic Encryption over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting (2022-10-24)

(xi) Paper: Premonition Net, A Multi-Timeline Transformer Network Architecture Towards Strawberry Tabletop Yield Forecasting (In review)

Chapter 2

Background

The goal of this background is to introduce various topics that are necessarily built upon each other, such that the reader can have a sufficient understanding of the underlying techniques, principles, and concepts presented through this thesis. Furthermore this background also serves to highlight certain deficiencies and gaps in current works that we seek to remedy. As such it is necessary to briefly cover some topics that the subsequent chapters will rely upon such as learning with errors (LWE), and ring learning with errors (RLWE) which is necessary to understand why fully homomorphic encryption (FHE) must be abelian compatible, which leads to subsequent consequences throughout. We also discuss the background of yield forecasting, its effects on argicultural supply chains, and the consequences it holds for food security.

2.1 Rings and Fields

Commutative rings are sets in which it is possible to add, subtract (via the additive inverse), and multiply, and still result in a member of the set. This includes the sets:

(i) $\mathbb{Z}$; integer, e.g.: $(-1, 0, 1, 2, ...)$ Formally: An integer is any number that has no fractional part (not a decimal).

(ii) $\mathbb{Q}$; rational numbers, e.g.: $(5, 1.75, 0.001, -0.1, ...) = (\frac{5}{1}, \frac{7}{4}, \frac{1}{1000}, \frac{-1}{10}, ...)$ Formally; a rational number is a number that can be in the fractional form $\frac{a}{b}$ where a and b are integers and b is non-zero.

(iii) $\mathbb{R}$; real numbers, e.g.: $(0, -1.5, 3/7, 0.32, \pi)$ Formally; a real number is any non-imaginary, non-infinite number.

(iv) $\mathbb{C}$; complex numbers, e.g.: $(1+i, 32+-2.2i, 5, -6i)$ Formally: A number which is a combination of real and imaginary numbers, where either part can be zero.

This does not include the sets:

- $\mathbb{I}$; imaginary numbers, e.g.: $where : i = \sqrt{-1}, (i, -i, 39.8i, ...)$ Formally: Imaginary numbers are any numbers which are multiplied by the imaginary unit i.

R for (commutative) ring shall henceforth be one of the four sets $\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$. In contrast a *field (F)* is any *commutative ring (R)* which may also perform division and still result in elements from that ring. This includes only the sets $\mathbb{Q}, \mathbb{R}, \mathbb{C}$ as not all elements in the set of integers ($\mathbb{Z}$) can be divided by another integer and still result in an integer. (Ershov, 2015) These rings are used through polynomial expressions instead of discreet matrices in LWE (see Section: 2.2) thus RLWE (see Section: 2.3). For formalisation if all of the following axioms are fulfilled then the resulting set is called a field:
addition axioms;
$given : (x, y, z \in R), then :$

$$\begin{aligned} (unity)\ 0 \in R \\ (closed)\ x + y \in R \\ (inverse)\ x, -x \in R \\ (commutative)\ x + y = y + x \\ (associative)\ (x + y) + z = x + (y + z) \end{aligned} \tag{2.1}$$

multiplication axioms;
$given : (x, y, z \in R), then :$

$$\begin{aligned} (unity)\ 1 \in R \\ (closed)\ x \cdot y \in R \\ (inverse)\ x, x^{-1} \in R \\ (commutative)\ x \cdot y = y \cdot x \\ (associative)\ (x \cdot y) \cdot z = x \cdot (y \cdot z) \end{aligned} \tag{2.2}$$

multiplicative additive axioms;

$given : (x, y, z \in R), then :$

$$(distributivity)\ (x + y) \cdot z = x \cdot z + y \cdot z \tag{2.3}$$

If all but multiplicative-inverse then this is a commutative ring with 1, if this also does not fulfil multiplicative-unity then this is just a commutative ring. (Ershov, 2015)

2.2 Learning With Errors

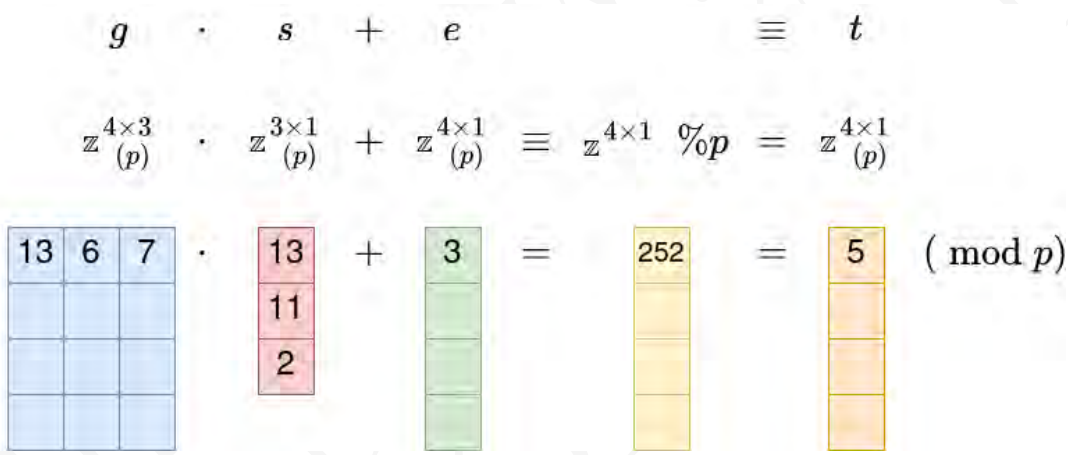

Figure 2.1: Non-linear, errored, learning problem, where g (blue), s (red), e (green) are randomly generated with a modulus of p we can further calculate t (orange) based on these prior generated values also with a modulus of p. This serves as the base for LWE based problems.

LWE as depicted in Figure 2.1 is simply a problem framing. Using this frame, we can ommit one or more parts to form the basic problems that give it its difficulty:

- LWE search problem; Given g (blue) and t (orange), can we find s (red). (Figure 2.1)
- LWE decision problem; Given g (blue), distinguish t (orange) from random. (Figure 2.1)

We can use this problem for strong and lightweight public-key cryptography, which is generally difficult to solve due to the error component adding a difficult to determine bias, that prevents solving by gaussian elimination.

First we generate our secret key (s), error (e), and coefficient (g may also be known as A) which we use to calculate (t may also be known as B). All with modulus p as integer matrices on various sizes dictated by g of size m by n. m represents the number of samples.

$$\begin{aligned} q \in \mathbb{Z} : q > 0 \\ m \in \mathbb{Z} : m > 0 \\ n \in \mathbb{Z} : n > 0 \\ g \in \mathbb{Z}^{m \times n}_{(\bmod p)} \\ s \in \mathbb{Z}^{n \times 1}_{(\bmod p)} \\ e \in \mathbb{Z}^{m \times 1}_{(\bmod p)} \\ t = (g \cdot s + e)\%p \end{aligned} \tag{2.4}$$

Our public key is thus t and g, and our secret key is s. We are now able to use these keys for encryption, decryption, signing, and verifying signatures.

This problem framing and the decision problem in particular are used as the basis in FHE. More specifically using RLWE

2.3 Ring Learning With Errors

RLWE is much the same as LWE discussed in Section 2.2 however as eluded to in Section 2.1 where the superset category of problems in LWE differ from the subset of RLWE problems is that instead of matrices we use polynomials over a finite field.

RLWE is specifically used in FHE and is what makes it possible to add and multiply the similarly-encrypted cyphertexts together. This commutative / abelian nature is what allows FHE to be used in deep learning (DL). However while there has been some research into combining DL and FHE, more complex activation functions, deeper complex networks, generally FHE compatible neural network algorithms and optimisations were still missing in large part in literature. Solutions to these problems are necessary to make encrypted deep learning (EDL) possible.

2.4 Deep Learning

DL is a subset of the broader machine learning (ML) set of algorithms. DL differs from ML in that DL is principally concerned with the use of neural networks (NNs), wheras ML uses classical and usually statistical models like linear, polynomial, logistic, Poisson regression etc, but is also commonly known to include models like K-nearest neighbours, random forests etc. ML has in the past been used for a vast array of different use cases from unsupervised clustering type problems like recommender systems, to supervised classification like diagnosis, to reinforcement learning like game AI. DL is a powerfull tool that has also become state-of-the-art in related time series forecasting (TSF) works (Zeng et al., 2022; Zhu et al., 2022; Minhao et al., n.d.; H. Zhou et al., 2021). many other adjacent fields so long as sufficient data is available to train them; Reinforcement learning (Cobbe et al., 2021), machine translation (Takase and Kiyono, 2021), sentiment analysis (Raffel et al., 2020), image classification (Yu et al., 2022), object detection (Q. Chen et al., 2022), and so many more. This is largely thanks to its flexibility, and powerful near-noetic ability to model the world through complex and interconnected distributions of data. Many of the state-of-the-art TSF approaches across multiple of the aforementioned domains use specifically transformers including for TSF. DL and in particular transformers have markedly increased the performance in multiple domains, and have had real world impact such as general-purpose transformer (GPT)-3 (Brown et al., 2020) in tools like ChatGPT. Transformers however have some debate with regards to their efficiency compared to simpler NN models in TSF (Zeng et al., 2022), and they may not be suitable to small datasets as a consequence. Lastly DL models having achieved unprecedented accuracy has lead to a glut of commercial companies collecting user data, on large scales, towards mass exploitation of this data. This presents obvious privacy issues, and is why for the sustainable future of DL we must introduce privacy-preserving machine learning (PPML) techniques like FHE to temper the consequences to privacy (Shokri and Shmatikov, 2015).

2.5 Privacy-Preserving Machine Learning

PPML refers to the development and deployment of ML models in a way that protects the privacy of individuals whose data is used to train and evaluate the models. This can be achieved through a variety of techniques, including:

(i) Differential privacy: This is a mathematical framework for adding noise to the data in a way that preserves privacy, while still allowing the model to learn valuable insights from the data on aggregate.

(ii) Federated learning: This involves training a machine learning model on multiple decentralised devices, such as smartphones, without the need to centralise the data on a single server. This means no one entity has access to the whole data, making it significantly harder to form knowledge of individuals.

(iii) Homomorphic encryption: This is a type of encryption that allows mathematical operations to be performed on encrypted data, without the need to decrypt it first. This can be used to perform machine learning on encrypted data, preserving the privacy of the individuals whose data is used. Homomorphic encryption is often limited by computational depth, meaning only a certain number of mathematical operations can be applied to it before it becomes garbled by noise. FHE on the other hand has an unlimited computational depth through the additional operation called boostrapping.

(iv) Multi-party computation: This involves splitting the data and model across multiple parties, and using cryptographic techniques to allow the parties to compute on the data without revealing it to each other.

There are many other techniques for PPML, and the appropriate approach depends on the specific needs and constraints of the application. In our case we focus on FHE as it is not only one of the least explored, but one of the only universally palatable PPML techniques that can be a gateway for the others once stakeholders begin to see the benefits garnered by DL. There are relatively few works in EDL (J.-W. Lee et al., 2022), we believe we can have the most significant impact here. Notable omissions in the field prior to our work were activation functions like approximations for rectified

linear unit (ReLU) (J.-W. Lee et al., 2022), automatic parameterisation (Dathathri et al., 2020), NN traversal of (Dathathri et al., 2020). Without these it would be incredibly difficult to implement modern neural networks to an acceptable standard without a significant drop in performance. This is more concretely specifically elaborated in Section 4.3.

2.6 Food Security

Food security refers to the availability of food and an individuals' or a populations' access to it. It includes the ability to produce, store, and transport food, as well as the ability to purchase and consume it. Food security is an important issue because it affects the health and well-being of individuals and communities.

There are several factors that contribute to food security. One important factor is the availability of food, which is affected by factors such as climate, natural disasters, and economic conditions. Another factor is access to food, which can be affected by factors such as poverty, lack of infrastructure, and political instability.

Food insecurity can have serious consequences, including malnutrition, disease, and even death. It is important for governments, international organisations, and other stakeholders to work together to address food insecurity and ensure that everyone has access to sufficient, safe, and nutritious food.

2.7 Agricultural Supply Chains

Agricultural supply chains involve the movement of agricultural products (like fresh produce (FP)) from the point of production to the point of consumption. This can include a range of activities, such as planting, harvesting, processing, packaging, and transporting the products. Agricultural supply chains can be local, regional, or global in scope, and can involve a variety of different actors, including farmers, processors, distributors, retailers, and consumers.

There are several key stages in an agricultural supply chain:

(i) Production: This is the first stage of the supply chain, where agricultural products are grown, raised, or harvested. This can involve a range of activities, such as planting seeds, watering and fertilising crops, and caring for animals. This is the primary stage of concern with respect to this thesis.

(ii) Processing: After agricultural products have been harvested, they may need to be processed in some way before they are ready for distribution and sale. This can include activities such as grading, sorting, packaging, and storing the products.

(iii) Distribution: Once agricultural products have been processed, they need to be transported to the next stage in the supply chain. This can be done through a variety of means, including trucks, trains, planes, and ships.

(iv) Retail: At this stage, agricultural products are made available for sale to consumers. This can involve a range of activities, such as setting up a stall at a farmers market, selling products through a grocery store or online retailer, or distributing products directly to consumers through a subscription service.

(v) Consumption: The final stage of the agricultural supply chain is when the products are purchased and consumed by the end-users. This can involve preparing and cooking the products, or using them in other ways, such as feeding them to animals or using them in industrial processes.

It is necessary to have an understanding of the individual stages of the supply chain, as yield forecasting in the first production stage has consequences when planning and executing the subsequent stages.

2.8 Yield Forecasting

Yield forecasting of FP is the ability to forecast how much of any given FP like strawberries will be yielded at a given time, ahead of the yield often by weeks to allow time for the various agricultural supply chain stages. In our case we focus on strawberry yield forecasting. This information is useful for a range of stakeholders, including farmers, processors, distributors, retailers, and consumers.

For farmers, yield forecasting of FP can help them to plan for the future, by giving them an idea of how much of a particular crop they will be able to sell and thus at what price it can be negotiated for. For processors, yield forecasting of FP can help them to plan for the future demand for their products. For distributors and retailers, yield forecasting of FP can help them to plan their inventory and supply chain management.

Overall, yield forecasting is an important tool for helping to ensure that the agricultural supply chain operates efficiently and effectively, by providing stakeholders with the information they need to make informed decisions about the production, processing, distribution, and sale of FP. This results in minimised waste, improved quality, reduced overhead, and more stable cheaper prices.

Currently there are some, but few works with DL towards TSF the agricultural yields of FP. The works that do exist have a tendency to use classical ML methods or simple DL methods (Paudel et al., 2021; Nassar et al., 2020). Few of these other works relate to soft-fruit forecasting and most use remote sensing datasets which have had issues with robustness in particular for finer precision agriculture like those within polytunnels (Sartore et al., 2022; Baghdasaryan et al., 2022). To this end we collect our own local dataset, and apply more complex transformer models which are state-of-the-art in TSF problems (Zeng et al., 2022; Zhu et al., 2022; Minhao et al., n.d.; H. Zhou et al., 2021). This is more concretely elaborated in Section 5.3.

Chapter 3

Data Collection and Data Pipelines

George Onoufriou, Marc Hanheide and Georgios Leontidis (2020a). 'The Augmented Agronomist Pipeline and Time Series Forecasting'. In
Please note this has been significantly expanded due to the brevity and age of this original paper.

3.1 Introduction

Machine/Deep learning is becoming a bigger and more important part of our daily lives through the rise of an ever-increasing quantity of available data. 3rd-party services use machine learning in combination with user data for tasks ranging from natural language processing (Do et al., 2019), image recognition, diagnosis (Biswas et al., 2019), detection, classification (Fawaz et al., 2019), generation, imputation, broadly prediction; medical diagnosis (Anderson et al., 2019), self-driving cars (Huval et al., 2015), facial recognition (Güera and Delp, 2018), etc. However one area in which deep learning has remained relatively stagnant is in agriculture, where data is scarce, low-quality, and of low-value forcing the use of remote sensing datasets or the like, as well as the existing research using classical techniques without many of the recent advances (Alvarez, 2009; Chlingaryan, Sukkarieh and Whelan, 2018; Prasad et al., 2006). We also found that there was a lack of willingness, and trust of the growers/ agriculturalists to release their potentially sensitive techniques latently in any data they provide. Thus if there is little to no data there can be little advancement with deep learning techniques, meaning prospective research will require self collected data to find any meaningful relations between the features and targets with

which to predict accurately and far enough ahead to facilitate timely and effective actions.

We contribute novel methods and results towards creation of a larger and more accurate plant yield prediction framework (Figure 3.1) which both helps automate but crucially improves upon current yield forecasting currently possible. Our work is facilitated by the RASberry research programme[1], which is a collaboration effort between UoL, Saga Robotics, and BerryGardens, funding autonomous strawberry data collection, under our direct control. This involves the generic expandable Thorvald platform, which is an autonomous robot ready for use in many terrains. Thorvald is an ideal candidate platform to use for our own experiments thanks to its autonomy, and available resources. The only drawback of using strawberries is that they are only grown from late June to early October.

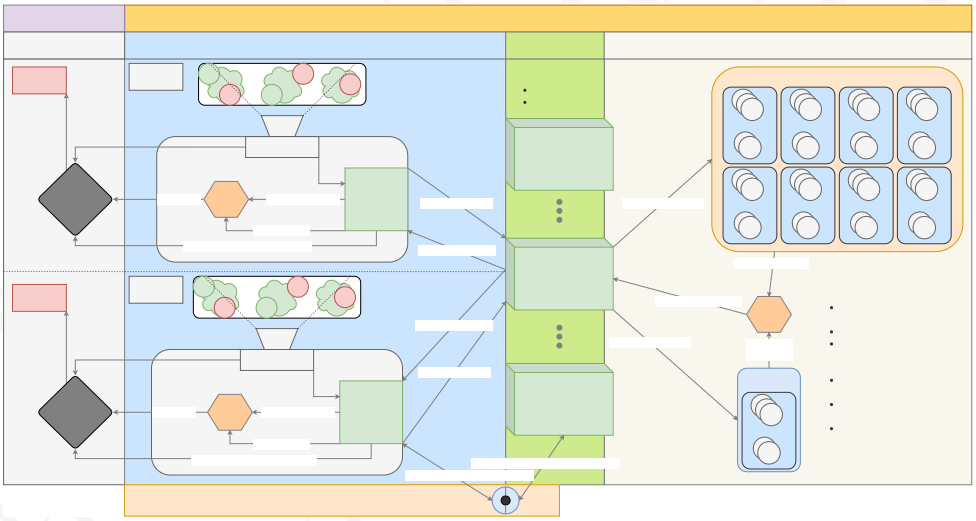

Figure 3.1: RASberry data distribution and aggregation pipeline, consisting of robot operating system (ROS), edge, database, and deep learning layers. The ROS layer is responsible for robot control. The edge layer is for edge compute and data capture. The DB layer is for aggregating data bateween multiple sites and Thorvalds. The back-end layer is for scaleable machine learning (ML) over all aggregated data (Onoufriou, 2019).

3.2 Contributions

(i) We conceptualise, implement, define, and exploit a novel data acquisition pipeline for strawberry tabletop from mixed data sources, including from ro-

[1] `https://rasberryproject.com`

botic traversal, stationary cameras, environment sensors, weather vanes, and irrigation / hydroponic related data. This improves data quantity through increased data acquisition sources and their frequency. This improves data quality through the repetitiveness of robotic traversal, and automated sensor collections over a known site. This improves speed, and reduces human time necessary at the human acquisition stage, and thus reduces cost.

(ii) Pairing with our data acquisition pipeline we also provide a unified data aggregation and utilisation pipeline, to stream the acquired data, where it is needed, in the format it is needed in, in a near-real-time manner. This reduces the time from observation to forecast, as neural networks can attain the data they need in near-real-time. This affords shareholders more time to plan, organise, and negotiate the labour, contracts, and logistics necessary to get as much of the crop yields to market as possible for the best price. This also allows many more varied approaches to be taken since the data is very easily and ephemerally tansformable for different neural network tasks, as it is being streamed to neural networks (NNs).

3.3 Background

Strawberries in the UK are grown using various mediums. They can be grown at ground level in soil, on trellises, or on strawberry tabletop. In many of the sites we have visited we observed that Berry Gardens Growers (BGG) farms tended to grow their strawberries in strawberry tabletop. Strawberry tabletop is a raised single-level row of coconut coir grow bags. Strawberry tabletop is then arranged in flat rows so that strawberries between rows do not impeded on the sunlight afforded to others. Strawberry tabletop rows are spaced out enough for humans to pass down the rows without damaging the crop even as it grows bushy. Strawberry tabletop is a convenient medium for farmers as it also tends to be supported by a hydroponic system to feed the bags, which they can regulate centrally, but also brings the crop off of the floor and at an ergonomic level, making it easier for humans to maintain, pick, and observe the strawberries. Growing strawberries in tabletop in this manner

also has the benefit of reducing the prevalence of weeds (less open soil for seeds to germinate), insects (most insects fly low level and crawling insects must climb up), and disease which strawberry fruits are particularly prone to when making contact with soil.

Hydroponics is a method of growing plants using nutrient-rich water solutions, instead of soil. It is a soil-less form of agriculture that allows plants to be grown in a controlled environment, such as a greenhouse, polytunnel, or indoor facility.

In a hydroponic system, plants are grown in containers filled with an inert growing medium, such as the aforementioned coconut coir or perlite, which provides support for the plants but does not supply any nutrients. Instead, the plants are nourished by a nutrient-rich water solution that is delivered directly to the roots. The water solution is typically enriched with a mixture of essential nutrients, including minerals and trace elements, which are necessary for the plants to grow and thrive.

There are several different types of hydroponic systems, including nutrient film technique (NFT), deep water culture (DWC), aeroponics, and others. Each type of system has its own set of advantages and disadvantages, however on the sites we visited and maintained the NFT technique was frequently used and the most suited to long rows of strawberry tabletop.

Hydroponics have several benefits over traditional soil-based agriculture. It allows for precise control over the growing environment, including temperature, humidity, and nutrient levels, which can result in faster growth and higher yields. It also allows for year-round production if plants are grown indoors in a controlled environment. Most growing sites for strawberries that we visited use polytunnels in outdoor environments, especially for June bearing strawberry varieties. Hydroponics can also be more water-efficient than soil-based farming, as the water and nutrients can be recycled and reused, reducing the amount of water and fertiliser needed. However, it requires a consistent and reliable source of water and electricity, as well as a high level of technical expertise and knowledge to set up and maintain a successful hydroponic system.

A polytunnel is a type of greenhouse that is made of a plastic or polyethylene cover

stretched over a metal or plastic frame. It is a low-cost and easily accessible way to create a protected growing environment for plants, and is frequently used over strawberry tabletop.

Polytunnels are designed to provide plants with a controlled environment, including warmth, moisture, and light, which can be adjusted within bounds as needed to suit the specific needs of the plants being grown. They can be used to extend the growing season, allowing plants like strawberries to be grown earlier in the spring and later in autumn, or to grow plants in areas with a climate that is not conducive to their growth like our cold wet climate.

To care for strawberries in a polytunnel, we need to monitor and control the temperature, humidity, and light levels inside the tunnel, as well as provide the plants with adequate water and nutrients through our central NFT hydroponic system. We also need to control pests and diseases, and prune or train the plants as needed.

Strawberries in particular are often germinated off-site in closely controlled conditions sometimes from abroad and transplanted to the grow bags on site. This makes the cold-start problem of new seedlings (like Figure 3.2 in our Riseholme campus) harder since the environmental data leading up to the plants transplantation holds little to no relation to the environment they were actually germinated in.

As it stands there are many existing methods that have been used to attempt to predict crop yield, using data such as remote sensing (You et al., 2017; Chlingaryan, Sukkarieh and Whelan, 2018), satellite image, climate conditions, geolocation data, etc (Liakos et al., 2018). However, there is high variation in the type, quality, and quantity in the datasets used, with very little from a standard dataset with which to use (Rahnemoonfar and Sheppard, 2017; You et al., 2017; A. X. Wang et al., 2018). The vast majority of papers use remote models relying primarily on: temperature, humidity, precipitation, and soil moisture. Some others attempt image based approaches but lack of data is a serious problem for them (Prasad et al., 2006). This means as far as yield prediction is concerned it is necessary to create a consistent, and granular dataset (Rahnemoonfar and Sheppard, 2017). All these papers use many different techniques, with a wide variety of data types such that

Figure 3.2: Strawberry seedling post transplantation into a grow bag cut-out. This grow bag sits inside a hydroponic basin that connects a row of grow bags end-to-end under a polytunnel.

they only marginally narrow the focus for our data collection efforts to things such as climate conditions, (Niedbała, 2019) meaning we will have to collect a large variety of data and thereafter assess the correlation to achieve the best results.

It has been highlighted in various works (Shafiq et al., 2021) that light intensity, light quality, CO2 levels, temperature, humidity, and water levels are key factors affecting any given plants growth beyond pure soil quality and suitability. In particular there are interesting relationships between light intensity and CO2 levels whereby a plant's growth rate increases with more light, up until some plant specific threshold whereby it becomes light stressed and its growth sharply declines. This threshold is affected by many factors but one of the most effectual is CO2 levels, the higher the CO2 levels the lower the light stressed threshold but also the plant grows faster below this threshold for lower intensity light. While in our work we do not seek to feature engineer this relationship, we are keen to see if the deep learning (DL) approach we developed will find this relationship significant enough in normal scenarios to take into account for yield forecasting.

3.4 Material and Methods

Towards training deep learning models capable of achieving less than 15% error over a 3 week forecasting horizon we require data. This data needs to show us the ground truth of what we are forecasting (strawberry yields), along with associated (causal) observations leading up to those forecasted outcomes. In this way we can use back-propagation to associate cause and effect through a models weights to model the relationship. It must be that the data sample that we inevitably train on is representative of the broader distribution of data otherwise our models become biased, and are unfit for inference in scenarios outside of those represented by the sample of data we do have. As highlighted previously however there is a distinct lack of available strawberry yield data (see 3.4.1). In this materials and methods section we outline what data we could gather external and how we process it, along with our own collected data using the Risehomle tabletop site.

3.4.1 Third-Party Data

Firstly we collected what data we could find openly available, by reference in other such yield forecasting papers. We found mention primarily of the California strawberry dataset and associated California weather data (CIMIS) over the same period. However as we began our search of this data we found that many of the resources pointed to had either experienced link-rot or were no longer available. What we could find was held behind authentication barriers, some of which we could pass through. In the end of the strawberry dataset we could find the California "pink sheets" which are PDF based outlines of the aforementioned yields for a subset of the years. This would be very arduous to convert into a usable form due to the PDF format and style of the content. We also found an excerpt of some of the weather data from Kaggle as the origin for this data had seemingly been offline for months at the point of investigation.

We also approached our industry partners for data, they unfortunately were prohibitively concerned with their data, making collaboration on their existing sites very difficult. We did manage to get a portion of yield data after a few years of requests,

Figure 3.3: Riseholme strawberry tabletop site. Depicting a Thorvald robot (running Ubuntu 18 at the time) with our initially mounted sensors and cameras. This image also shows some of the stationary cameras on a tripod, the other stationary camera cannot be seen as it is in the opposite tunnel.

however this yield data was horribly malformed, inconsistent, and had no contributing features that we could correlate to this yield. This made this data source untenable.

3.4.2 Riseholme Data

To alleviate the plethora of issues around data availability that we experienced, we collected our own data from our Riseholme strawberry tabletop site (as depicted in Fig 3.3). We used several different data streams to create as representative a dataset as possible, that described the environment the strawberries experienced, and the performance of the strawberries given those prior conditions.

Yield Data

We collected yield data as it was critical that our methods had some observed values for outcomes. This way the neural network could relate these observes outcomes / outputs to come observed inputs. This way we can build a model from the historic data that we know. This also affords us the ability to compare our trained models with data they have not seen before. This means we can evaluate the performance of our models and select the ones we deem to be the best.

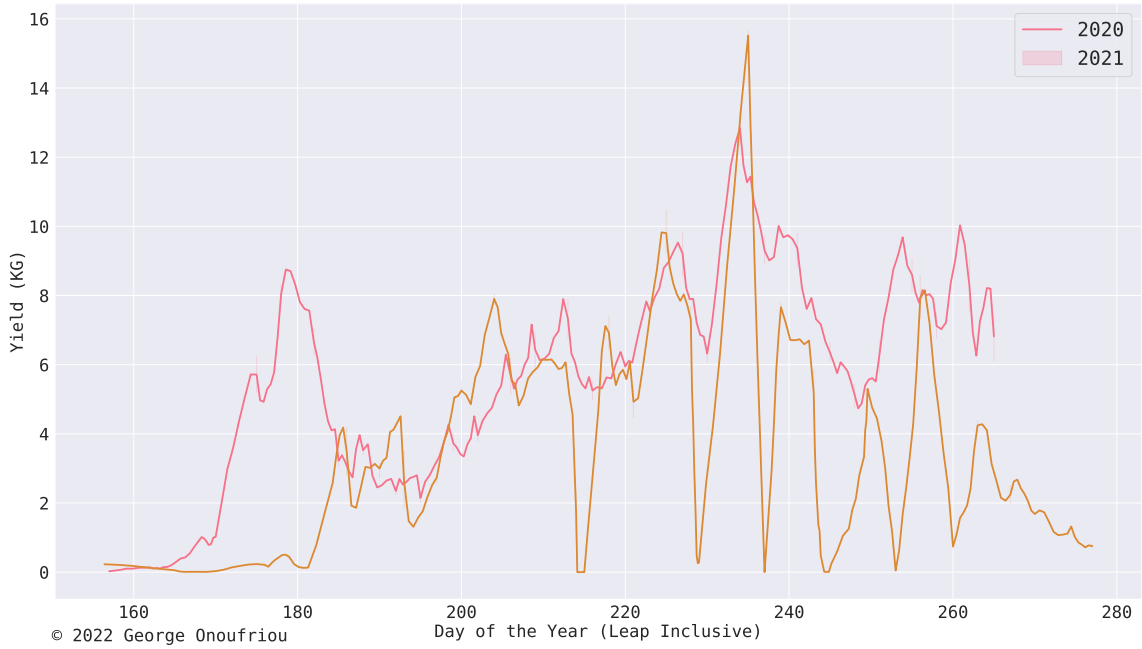

Figure 3.4: 2021 strawberry yields 7 day rolling average over ISO day of the year for the years 2020 and 2021. This graph overlays the two seasons to show the similarity between the two seasons and how the historic performance of yield is very indicative of subsequent years of yields where the difference is primarily dictated by mitigating or unusual conditions.

To collect yield data we needed some measure of yield. In the first growing season (2019) we unfortunately had to use punnets due to limitations of availability, and shortness of time. Subsequent growing seasons (2020, 2021, 2022, also see Figure 3.4) used the actual weight in kg of strawberries, which gave us a much more consistent metric, as the size of berries and prevalence of berries inside a punnet can alter just how much mass a punnet contains making it a poor proxy for the mass of fruit produced by the strawberries.

Similarly in the first growing season (2019) we collected strawberry yield data at polytunnel level, we found this to be insufficient as it provided too few examples for a sufficiently performant and complex neural network to train with. The strawberries require time to grow, strawberries can only be picked in intervals of a few days otherwise too little would have changed to warrant the labour and cost. The strawberry growing season only holds for 10-12 weeks depending on the first frosts. If we harvest strawberries twice per week, for 10-12 weeks with two polytunnels and only kept the data at polytunnel level we would thus only have 48 yield values at best. Thus in subsequent seasons we collected this yield data much more granularly

Figure 3.5: Rip, overripe, and underripe strawberries in the Riseholme tabletop site. This depicts an overripe strawberry (slightly purple tint) being eaten by a wasp which was and is a significant pest for strawberries when they are left on the the stem for too long.

such that we knew which mass of strawberries came from which tabletop row. This is beneficial as it not only provides us orders of magnitude more examples to train with, but it also allows us to much closely correlate conditions experienced by the strawberries to output yields. In this way we could increase our yield values for training from 48 to 240 at best, given 5 rows per polytunnel.

Lastly we also collected waste data, in particular mechanical, pest, disease, and quality waste data. This is to allow us to model how much of the crop is likely to be lost to harvesting or mechanical loss, how much is lost to pests (like those depicted in Figure 3.5), how much is lost to various diseases, and finally how much is lost due to being of inadequate quality. Unfortunately this was only possible for the first growing season, which as previously mentioned was insufficient due to using punnet-based measures instead of weight.

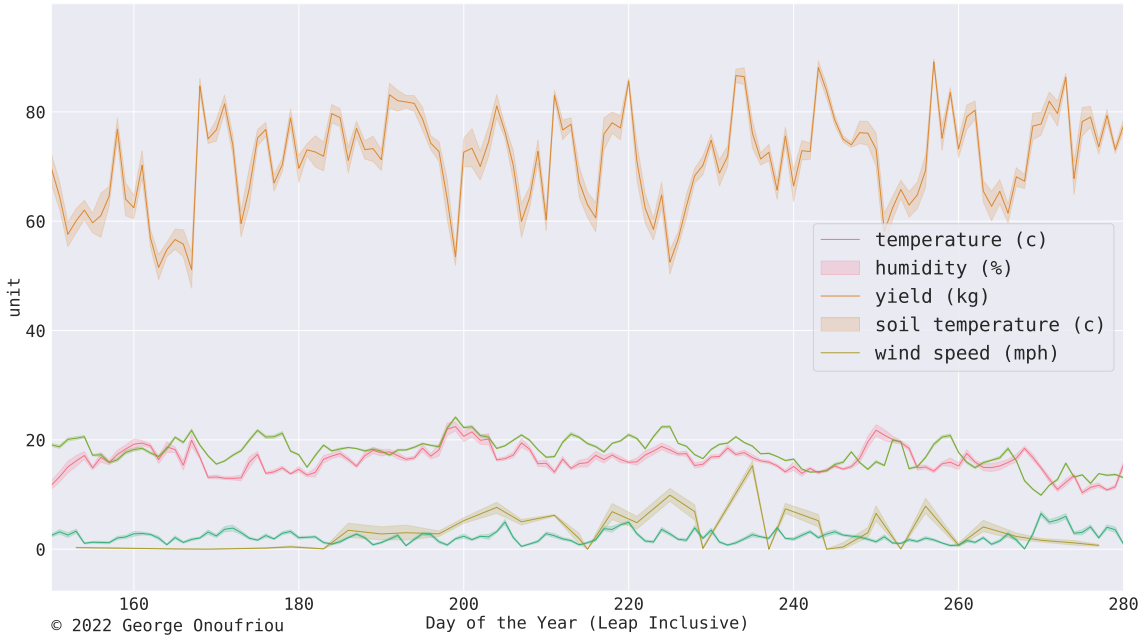

Figure 3.6: 2021 strawberry data line plot with temperatures, humidity, yield, soil temperature, and wind speed, over ISO days. This image serves to visually depict how the core strawberry forecasting data varies over any given season.

Environmental Data

We collected environmental data (as depicted in Figure 3.6) so that we could model the relationship between the environment experienced by the strawberries and how they affected the resulting yields. However it is important to note that much of the environment on our small site is shared between all the strawberries since it is not as large as industrial sites.

To collect environmental data we needed to identify what environmental features are significant. We based this on what BGG agronomists highlighted to us and what we reasonably expected to have an affect on the output yields. We collected as much supplementary data as possible but collected these aforementioned features as a priority. The features we focused on were light intensity, temperature (as shown in Figure 3.7), humidity, precipitation, CO2 levels, and pressure. We would have liked to have been able to measure light quality but due to the difficulty with equipment, the light being sunlight, and the scarcity of such equipment on industrial farms it would be difficult to relate this to real-world scenarios.

We collected environmental data using both a weather vane above the polytunnels

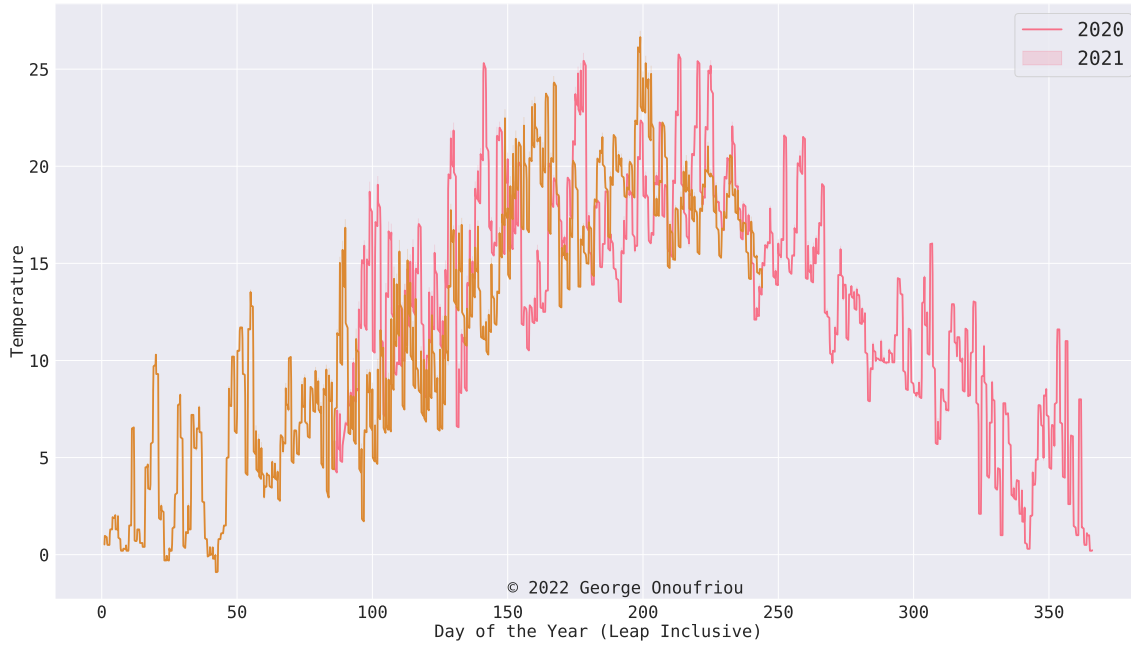

Figure 3.7: 2020, and 2021 strawberry soil temperature averages (in degrees Celsius), showing how the soil (grow bags) temperature changes over the year by ISO date.

and a sensor inside one of the polytunnels. This was initially to help distinguish the environment outside the polytunnel that would likely be seen by remote sensing datasets with the environment inside the polytunnels. We collected this data in 15 minute intervals to give us very granular information on the environment which if we wanted to we could resample to a lower sample rate. It would have been significantly less representative and harder to go to a higher sample rate if we had collected this data in large intervals.

Hydroponics and Bag Data

To pair with the above ground environmental data we also decided to gather the hydroponic and bag data. This helps to describe what the root system is exposed to which is clearly significant since the water and nutrients the plant uses to grow come from, and are ultimately limited by the root system and its size, and its access to said water and nutrients. While it is difficult to model the root system itself since this would be somewhat destructive to our active strawberry tabletop, a NN should be aptly capable of modeling this hidden factor / coefficient for root systems given yields and backpropagated errors.

The soil features we garnered are moisture, irrigation input levels, irrigation run-off levels, along with temperature. The underlying pre-built system these trusses use collects data with a 2 minute sample rate. This creates exorbitantly high density data which necessarily needs to be re-sampled to align with other samples, we store it with the original sample rate to ensure we have no loss for varying our experiments.

Image Data

Lastly the final form of data we collected was image data. We had various such image data streams broadly in two categories; stationary camera data, mobile camera data. The stationary cameras are any cameras that stay on site and capture single images in intervals over a prolonged period of time, for a single section of the growing site. Mobile camera data are any cameras that seek to capture a whole row or rows of strawberry tabletop, by taking frames from different sections of the growing site over time. We had multiple mobile cameras, from phones to Thorvald robot mounted cameras (see Figures 3.3 and 3.1). The robot mounted cameras gave us much more consistent and repeatable image data sets that were collected in regular intervals, at the same height, and of the rows traveresed over time.

3.4.3 Data Aggregation

Towards using this data we need to create data aggregation pipelines to allow us to access and process the data, we use our previously published Nemesyst framework to orchestrate this as depicted in Figure 3.1(Onoufriou, Bickerton et al., 2019). We used a ROS layer which operated across Thorvald robots for autonomous control. This allowed us to repeatably and reliably coordinate data collection between sites, robots, and over time intervals. We use Nemesyst to manage local data sources and synchronising them to the broader MongoDB instances to a central MongoDB storage for this high volume data. This steaming of data, gave us a near-real-time view of the data as it was being collected. This has the advantage of being scaleable, distributed, and authenticated so that not only this PhD but others can benefit from the data. This also affords us database-side computations through aggregate pipelines so that minimal amounts of data are necessary to transfer. One issue that we did encounter

was network connectivity, so we also maintained a local MongoDB store at the edge, so that data could be stored while it was waiting for connectivity to be transfered off robot. This is also why we specify near-real-time as if the robot is in a patch where it has no such connectivity the data is delayed until it is reasonable to transfer the data. This now aggregated data also makes it readily available for offline back-end deep learning that has no coupling to robots or ROS, or any such packages that may limit our ability to use the latest and presumably most optimal deep learning frameworks and GPU drivers. These sites are responsible for training, and model evaluation of NN models, along with packaging them back into the database for unpacking and use at a local level. These NNs can then be selected based on their performance and suitability to the application, such as the most performant yield prediction of strawberries versus other berries. The selected NNs used locally can then be used in future to inform decision making processes of the robot, such as attention mechanisms. Attention mechanisms can be used with our databases as a message passing interface to alert and request the attention of specialist agronomists to identify uncertain cases and help with learning along with any immediate control needs.

We did not use the image data ourselves, but these pipelines applied to images were useful for other PhDs. Some of which further informed our own work, including works dealing with berry counting.

3.5 Results

Table 3.1 shows how much data we collected, and from what data sources. It also shows the changes in both data sources and to some extent techniques encoded in the size of the data relative to number of samples.

2019: the PhD had begin being conceptualised and we were collecting data that we believe we would need. We found a plethora of difficulties collecting data as we were learning as we were collecting. In particular since we had only recently begun, we did not know what was, and what was not significant towards the PhDs ends. There was however a clear need for a better data collection pipeline, as none were readily

Year	Source	Size	Count
2019	Yield (Punnets)	96.0 B	24
2019	Weather Vane Sensors	5.95 MB	35000
2019	Stationary Imaging	34.7 GB	14000
2019	Robotic Imaging	6.74 GB	1500
2019	Pi Environment Sensors	1.22 MB	26000
2020	Yield (Kg)	41.0 kB	320
2020	Weather Vane Sensors	6.05 MB	35000
2020	Irrigation Sensors	23.1 MB	188500
2020	Stationary Imaging	34.7 GB	14000
2020	Robotic Imaging	222 GB	111000
2020	Pi Environment Sensors	1.22 MB	26000
2021	Yield (Kg)	45.1 kB	363
2021	Weather Vane Sensors	6.05 MB	35000
2021	Irrigation Sensors	23.1 MB	188500
2021	Stationary Imaging	34.9 GB	13000
2021	Robotic Imaging	77.8 GB	15000
2022	Yield (Kg)	*	*
2022	Weather Vane Sensors	6.05 MB	35000
2022	Irrigation Sensors	23.1 MB	188500
2022	Robotic Imaging	191 GB	766000

Table 3.1: Seasonal data collection outcomes on the Riseholme strawberry tabletop. This table shows how the data collection varied between seasons as better techniques were found. 2022 data, due to changes in management, is only partially available to us and our use.

available, we had to create our own. This took significant up-front time to create these initially. The lack of such a pipelines at this early stage hindered some of the data collection efforts towards yields, along with the natural early PhD uncertainties.

2020: the data collection was now fully underway with all the pipelines in place to collect, transform, and transfer all the data. This was especially important since access to the site was heavily restricted thanks to the Coronavirus disease 2019 (COVID-19) pandemic which meant any adjustments to data collection became significantly harder. Thankfully only minor adjustments were necessary, most importantly yield data was now collected both by weight, and by row. This gave us a more accurate representation of the mass of strawberries than punnets, and now being more granular allowed us to grow our yield observations 10-fold. We also added irrigation data from the central management system of the strawberry irrigation system which is very dense data that needs significant transformation and handling.

2021: the data collection continued with little manual intervention other than basic site tasks, observation, and of course picking and (trans)planting of the strawberries. We found that the raspberry pi environment sensors were not needed and were not of sufficient quality to warrant re-use and effort to overcome COVID-19 related barriers.

2022: the site manager changed this year, and with this change came more difficulties attaining and accessing data. There was also a transition to a different yield aggregation system, which to this date has blocked us from accessing yield data for this year.

It should be noted that due to constraints in human time, we could only collect yield values twice a week. Industrial sites will collect yields 2-3 times in a week, so that means we are roughly in line with industrial schedules.

Table 3.2 thus shows some very early experimental results that demonstrate the ability of various recurrent networks to learn with this limited labelling. Due to the size of the data and how early on in the process we are our results (3.2) are split plainly 80% training, 20% testing, with around 10-13 epochs for saturation taking less than a few minutes to train using only environmental and yield data to evaluate its usefulness.

Technique	Mean Absolute Error (Test set)
Vanilla Recurrent Neural Networks	0.210
Long Short-Term Memory	0.381
Gated Recurrent Units	0.155

Table 3.2: Time series forecasting of yield by number of punnets from the original 2019 dataset.

3.6 Discussion

Firstly we found some conflicting information from our industry partners that they currently favour the Zara variety despite its smaller yield outputs (see Figure 3.8). This has implications that absolute yield is not the only factor being selected for. We know this to be the case that they want the largest most flavoursome berries, but where the thresholds are drawn is still unclear or has not been codified. To make things clearer due to this ambiguity we will for now ignore the flavour of the berries, since after all, the food waste, yield outputs, and costs / purchases are done according to weight.

We also see that the environmental temperature is highly fluctuating in Figure 3.9 especially when compared to the soil temperatures that the roots experience in Figure 3.7. In these graphs it is clear to see how related the previous years data is to the new years data, along with how the temperature can have a significant affect on the yield outputs.

The early year 2019, and the latest year 2020 have been difficult. 2019 being so early in the PhD had difficulties with starting since nothing was available. This meant that the yield data was not satisfactorily collected. We trained some very early models which attained results as laid out in Table 3.2. While these preliminary results were satisfactory at the time, performance in later chapters will be attained and be a more critical aspect of discussion.

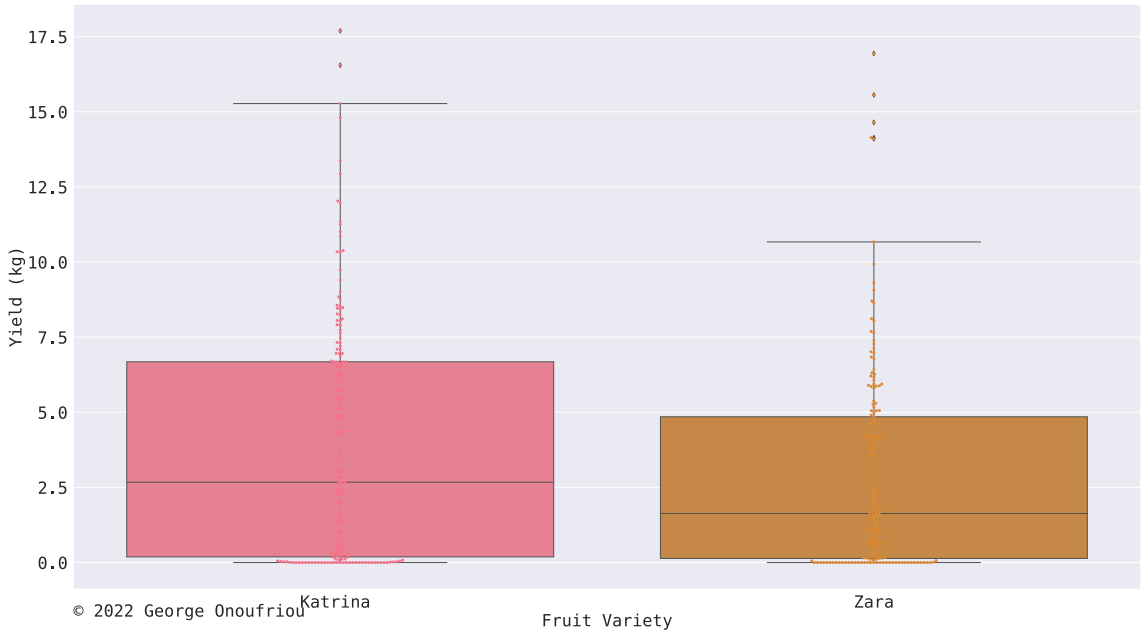

Figure 3.8: 2021 strawberry yields by strawberry variety; Katerina and Zara. Both of which are industrial varieties. Interestingly we are informed that Zara is a very popular variety in spite of its smaller yield outputs in our Riseholme site. This may indicate that either in industrial scenarios a proper analysis is not done when selecting varieties, or that our findings here are not properly representative of these larger industrial sites.

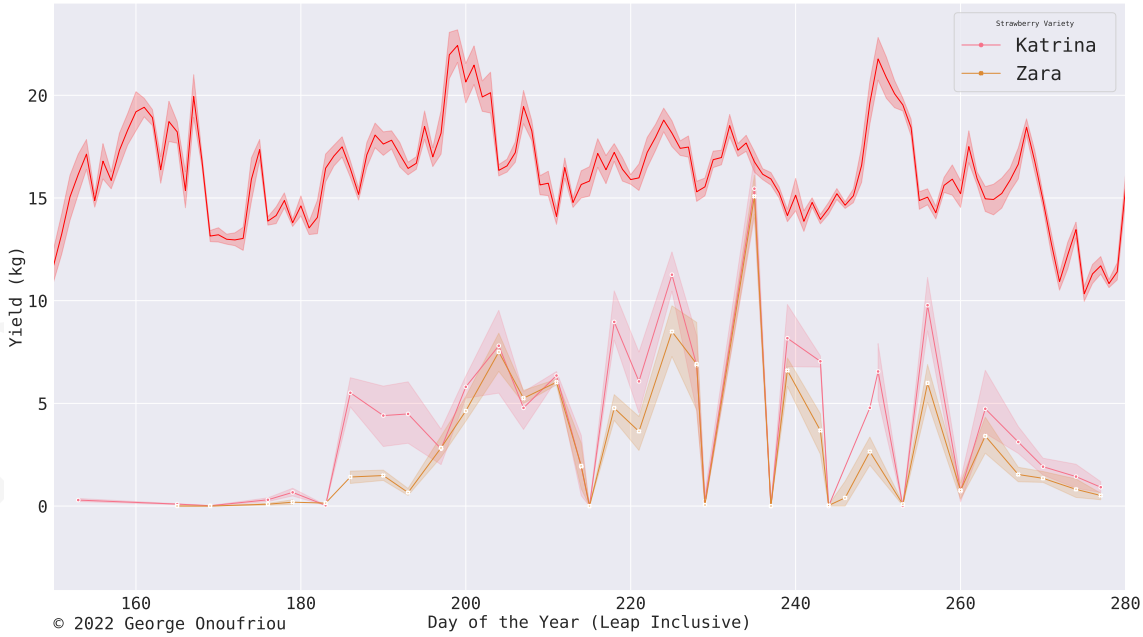

Figure 3.9: 2021 and 2020 strawberry yields and temperatures per ISO day, showing the relation between variety, temperature, and yields. This also shows how erratic conditions can be and how much other factors may also contribute.

3.7 Conclusions

Data necessary for yield forecasting is incredibly difficult to attain. We have resorted to self-collecting such data to ensure we can collect and adapt our collection methods according to directions indicated by the data itself.

A need has been identified for more autonomous data collection to collect more data along with more consistency to feed to NNs to learn more complex representations. To this end we have used our distributed Nemesyst database pipelines for data aggregation and modelling as well as distribution in more complex scenarios such as autonomous agricultural data collection. However good planning, and data collection of yield is still a key component to being able to create good data towards good performant NNs.

We have seen how this can augment the ability of growers and inform them of interesting correlations in its own right. This then allows us to collect data and predict outcomes such as crop yields along with the associated baseline choices such as varieties of strawberries to use.

Lastly our pipeline can also be used as message passing interfaces for agronomists to monitor, be alerted of any unusual cases, label difficult examples, and potentially control the robots to support their efforts. Our next step is to create more performant deep learning models are of the result in such that this can be used for more effective decision making.

Chapter 4

Privacy Preservation and Fully Homomorphic Encryption

George Onoufriou, Marc Hanheide and Georgios Leontidis (2022a). 'EDLaaS: Fully Homomorphic Encryption over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting'. In: *Sensors* 22.21, p. 8124

4.1 Introduction

Privacy is slowly becoming of greater interest (Figure: 4.1) to the broader public, especially during and after particular scandals, such as Cambridge Analytica (corporate actors), Edward Snowden on the five eyes (state actors), (Snowden, 2019) and more recently the Pegasus project on the cyberarms NSO group (both corporate and state). This increased concern for privacy has over time manifested itself in many forms; one of the most notable example being in legislation such as the general data protection Act (GDPR) (Parliament, 2018).

A less thought-of field where privacy is of concern is the agri-food sector. Stakeholders often are incredibly reluctant to share data, due to real, or perceived sensitivity. We believe that this data sharing reluctance originates from two factors. Data is not being collected due to the unawareness of the value-for-cost it can offer, and data is not shared due to concerns over loss of competitiveness if their techniques were leaked. This means it is incredibly difficult for new and possibly disruptive approaches to be used toward forecasting and thus later optimising some compon-

ent in the agri-food chain. One such disruptive approach is the application of deep learning which has become state-of-the-art in almost all areas where sufficient data is present with which to train it. There are many reasons why such new approaches are necessary but the key area we gear our work towards is tackling food waste at production, by forecasting accurate yields. Here in the UK we have dual problems of

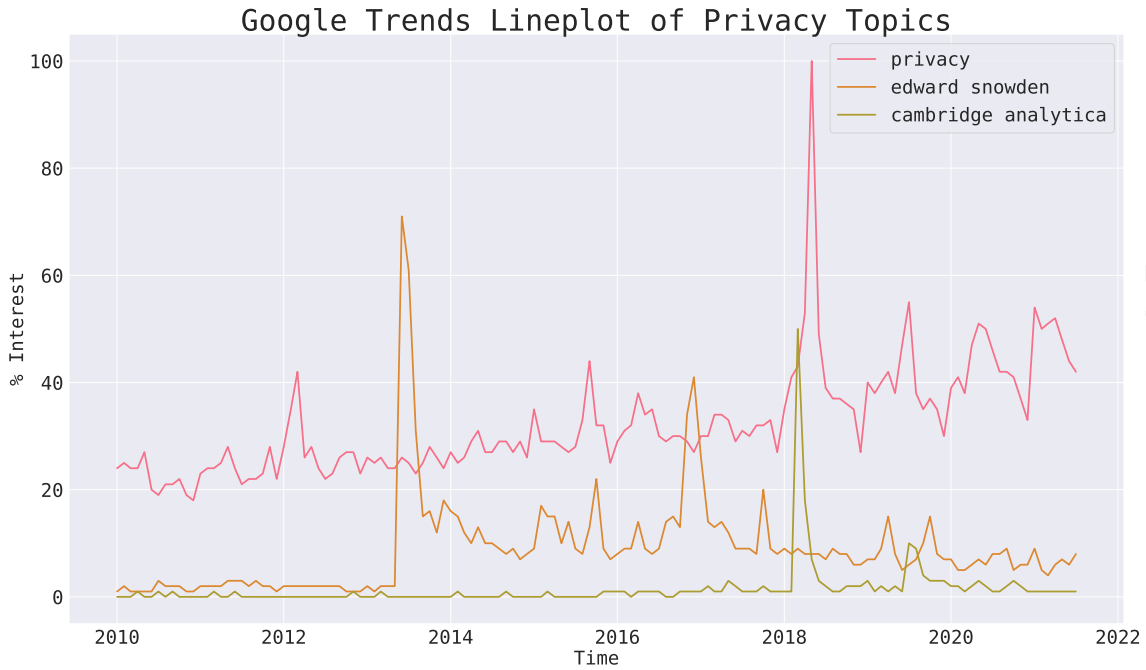

Figure 4.1: Trends of privacy (red), Edward Snowden (orange), and Cambridge Analytica topics (green) on Google trends since 2010 showing a slow but steady increase in the interest of privacy, and particular peaks around events such as the Cambridge Analytica scandal and smaller peaks roughly correlated to Julian Assange. (Google, 2021)

food insecurity and high food waste. It is estimated that the annual combined surplus and food-waste in primary production sis 3.6 million tonnes (Mt) or 6-7% of total harvest. A further 9.5Mt is wasted post production / farm. 7.7Mt is wasted in house holds and 1.8Mt is wasted in manufacturing and retail. The total food purchased for consumption in the UK is 43Mt (Environment Food and Affairs, 2021). Specifically in the soft-and-stone fruit industry a large consortium of growers in 2018 over estimated by 17.7% for half of the growing season, while the remainder of the season they under-estimated by 10%. Underestimation leads to surpluses which create extra cost in fruit disposal along with de-valuing expected produce. Overestimation leads to fix-purchasing which entails importing fruit to cover the shortfall in the expected produce. This costs the consortium 8 Million GBP a year in losses, while the rest of

the industry is estimated to have incurred 18 Million GBP losses a year at the time. The effect of climate change has been exasperating the difficulties in yield forecasting due to the more erratic environmental conditions. Considering that freely available agri-food data are hard to find, given they are highly sensitive, progress in adopting AI technologies are hindered.

As far as using machine learning (ML) is concerned, It is extremely difficult to build and deploy neural network (NN) models to forecast agricultural yields due to the aforementioned privacy/ sensitivity concerns that mean data for training and using these neural networks is scarce. However the impact of using ML technologies in agri-food / fresh produce (FP) supply chains has been shown to be substantial (Kollias et al., 2022; Onoufriou, Bickerton et al., 2019; Thota and Leontidis, 2021). A solution that involved distributed learning was recently proposed with an application on soy bean yield forecasting (Durrant, Markovic, Matthews, May, Enright et al., 2022), which assumes that distributed training is possible. Towards providing an alternative solution to this, we propose to new techniques and formulations using fully homomorphic encryption (FHE) and demonstrate how it works and performs in a bespoke strawberry dataset (Katerina and Zara varieties) that was collected in our strawberry research facility in Riseholme Campus at the University of Lincoln, UK.

FHE affords us the ability to compute cyphertexts without the ability to detect or discern its contents, acting as a truly blind data processor in encrypted deep learning as a service (EDLaaS) applications (Onoufriou, Mayfield and Leontidis, 2021a). In particular EDLaaS is especially useful in highly sensitive/ highly regulated industries such as medicine/ patient data (especially due to GDPR), trade secrets, and military applications. Though FHE is not a panacea. Special care must be taken to ensure/ maximise the security of cyphertexts and the biggest problem with this is it is not immediately apparent if this is not ensured often requiring a deep understanding of the underlying cryptography such that the parameterisation can be understood, analysed, and balanced against. However a standard metric used throughout as a commonality is the number of bits used for the private keys. It is commonly considered that a private key with 128 bits is considered secure (*Microsoft SEAL*

(release 3.4.5) 2020; Dathathri et al., 2020). We maintain this minimum level of security throughout all our experimentation and implementations.

4.2 Contributions

Our contributions towards encrypted deep learning (EDL) given the current state of the field and related works (Section: 4.3.2) are:

(i) We propose a new block-level automatic cyphertext parameterisation algorithm, which we call autoFHE. We also seek to showcase autoFHE in both regression and classification networks, which still appears to be a misunderstood and ongoing problem (Falcetta and Roveri, 2022).

(ii) We provide and showcase open-source encrypted deep learning with a reproducible step-by-step example on an open dataset, in this case Fashion-MNIST, achieved through a dockerised Jupyter-lab container, such that others can readily and easily explore FHE with deep learning (DL) and verify our results.

(iii) We show a new application for encrypted deep learning to a confidential real-world dataset. This can be used in conjunction with our open example dataset to evaluate the performance of EDL when applied to various tasks in classification and regression.

(iv) We demonstrate how neuronal firing in multi-directed graphs can be achieved in our different approach. This neuronal firing algorithm is very different to standard NN approaches since it has to account for computational depth experienced by cyphertexts allowing us to go deeper, faster, and with more certainty in the integrity of the cyphertexts.

(v) We show and detail precisely the computational graph of how a convolutional neural network (CNN) can be constructed using FHE in particular how handling of the sum-of-products can occur. This along with our easily reproduced example, should help clarify many otherwise omitted details from previous works that hinder their application by new researchers to this new field.

(vi) We show recent advancements in FHE compatibility like acrrelu approximations in greater detail along with problems/ considerations as part of a whole computational graph. We also backpropogate the dynamically approximate range of rectified linear unit (ReLU). With ReLU we are much more able to approximate current research results which also use this same, extremely popular, activation function.

4.3 Related Work

4.3.1 FHE Background

FHE is a structure-preserving encryption transformation (Gilad-Bachrach et al., 2016), proposed by Craig Gentry in 2009 (Gentry, 2009), allowing computation on cyphertexts ($\varepsilon(x)$) directly (addition and multiplication) without the need for decryption. This is what could be considered the first generation of FHE as implemented by Gentry in 2011 (Gentry and Halevi, 2010) and the Smart-Vercauteren implementation (Smart and Vercauteren, 2010). Gentry's implementation for any given bootstrapping operating took anywhere from 30 seconds, for the smallest most "toy" example, to 30 minutes for the largest most secure example, with the former having a public-key of 70 Megabytes, and the latter a public-key of 2.4 Gigabytes in size (Gentry and Halevi, 2010). Clearly this would be far too lengthy to be practically viable, however there have been several generations of FHE since building on these initial works and improving computational and spacial complexity; second generation: BV (Brakerski and Vaikuntanathan, 2011), BGV (Brakerski, Gentry and Vaikuntanathan, 2011), LTV (Lopez-Alt, Tromer and Vaikuntanathan, 2013), BFV (Fan and Vercauteren, 2012), BLLN (Bos et al., 2013); third generation: GSW (Gentry, Sahai and Waters, 2013); fourth generation: CKKS (Cheon, A. Kim et al., 2017a). Here we focus on the Cheon, Kim, Kim, and Song (CKKS) scheme, for a plethora of reasons:

(i) CKKS operates with fixed point precision unlike all other schemes, which are

necessary for computation of neural networks with activations and inputs usually falling in the range $0, \pm 1$ (Cheon, K. Han et al., 2018).

(ii) CKKS has multiple available implementations (PALISADE (Al Badawi et al., 2022), HEAAN (Cheon, A. Kim et al., 2017b), Microsoft simple encrypted arithmetic library (MS-SEAL) (*Microsoft SEAL (release 3.4.5)* 2020), HElib (Halevi and Shoup, 2020), etc). Only PALISADE (Al Badawi et al., 2022) and Lattigo (Mouchet et al., 2020) are known to implement CKKS with bootstrapping, although many others have these features road-mapped.

Our implementation uses MS-SEAL, a popular FHE library. Many of our techniques proposed here stretch to almost all other implementations since they follow the same basic rules, albeit with slightly different implications on things like parameters. In this paper we focus on using FHE without bootstrapping, or more precisely levelled-fully-homomorphic-encryption (LFHE), meaning we calculate specific sized although generalised (implementation) neural network circuits. Despite CKKS being the best candidate for forms of encrypted deep learning, it has certain shortcomings. Fundamentally, CKKS cyphertexts are the most atomic form of the data. This is a consequence from the optimisation used in many FHE schemes where a sequence of values (the "message" or plaintext data) are encoded into a single polynomial, and then this polynomial is what is then encrypted (Figure: 4.2). This means there is less overhead since we are encrypting multiple values together, but it means we cannot operate on this value alone, we must always be homomorphic, i.e maintain the same structure and operate on all values. Thus if we encrypt a polynomial of length 10, that shall be the smallest form of the data until it is either bootstrapped or re-encrypted. Therefore, we are only able to operate on the 10 elements as a single whole, i.e. we cannot operate on the 3rd element in the array alone to produce a single number answer. In addition, CKKS cyphertexts computational depth (pre-bootstrapping) is directly related to the length of the polynomial slots, which means we must choose our parameters carefully to ensure we do not have unnecessarily large cyphertexts, and thus slow operations. Lastly, CKKS as with many schemes requires that two cyphertexts operating with each other, must share the same parameters and be from the same private key. This means when for instance we have

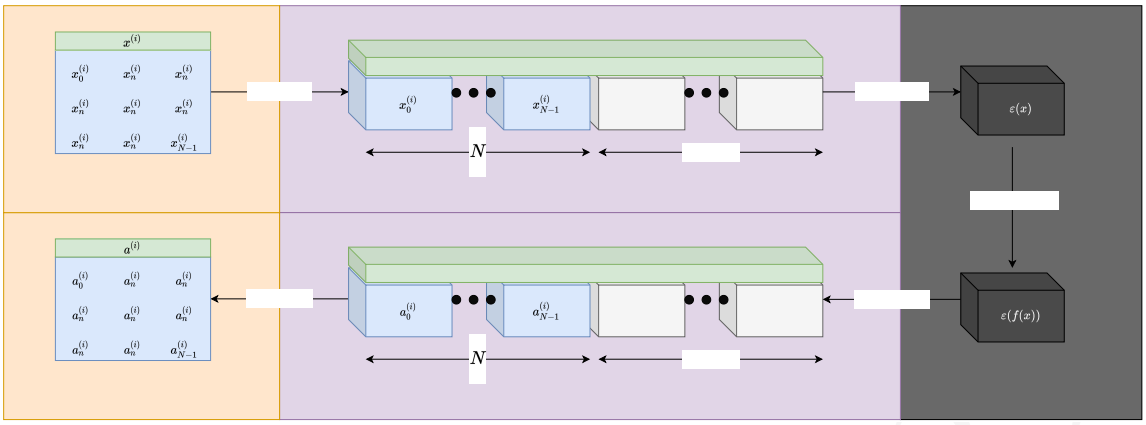

Figure 4.2: Overview of distinct FHE cyphertext stages in computation and properties (Onoufriou, 2021).

multiple inputs into a neural network, all directly interacting cyphertexts must be of the same key. This complicates some automatic parameterisation logic which we will discuss later.

4.3.2 Related Works

Encrypted Deep Learning

There have been many other works that use FHE (bootstrappable) or Levelled-FHE to compute some form of neural network. A few notable examples for FHE and CNNs are by Lee, (J.-W. Lee et al., 2021), Meftah, (Meftah et al., 2021), Juvekar (Juvekar, Vaikuntanathan and Chandrakasan, 2018), and Marcano, (Marcano et al., 2019). Lee uses a modified version of the MS-SEAL library to add bootstrapping as MS-SEAL does not currently support it. Lee shows FHE and DL used on the CIFAR-10 (Krizhevsky, G. Hinton et al., 2009) dataset to mimic the ResNet-20 model achieving a classification accuracy of 90.67%. Juvekar uses the PALISADE library implementation of the BFV scheme with their own (LFHE) packed additive (PAHE) neural network framework to compute both MNIST and CIFAR-10. Meftah uses Homomorphic Encryption Library (HELib) (Halevi and Shoup, 2020) similarly to Lee is particularly focused on improving the practicality of FHE as a means to compute DL circuits. Meftah seeks to do this towards computing ImageNet (Deng et al., 2009) with the second generation BGV scheme (Brakerski, Gentry and Vaikuntanathan, 2011) (on integers) as opposed to Lee using the fourth generation CKKS

scheme (Cheon, A. Kim et al., 2017a) (on floating points). Lastly Marcano similarly to the previous is also concerned with the computational, and spatial complexity of using FHE as a means to compute convolutional circuits. Marcano appears to use a custom FHE implementation on fixed point number format, taking 36 hours to train on the MNIST dataset. It is unclear in all of these papers however, how exactly the gradient descent or backward pass of the neural networks are implemented, which is necessary for neural network training. They also lack detail in key stages of the forward pass such as how they dealt with calculating the sum-of-products of the CNN since a homomorphic cyphertext cannot be folded on itself to form a single number sum, or if they used point-wise encryption to be able to sum between cyphertexts how they dealt with the sheer size of this plethora of cyphertexts. Lastly the above papers do describe in some detail how some of their parameters are decided in particular with regards to security, but they do not cover much on the computational depth or precision effects these parameters have on the cyphertext such as the modulus-switching chain.

FHE Graph Parameterisation

Here FHE graph parameterisation means deriving the FHE parameters from a graph, such as the computational depth and thus the parameters like the modulus size. There have been a few works that define FHE graph parameterisation, the most notable and similar of which is Microsoft encrypted vector arithmetic (MS-EVA) (Dathathri et al., 2020; Falcetta and Roveri, 2022). MS-EVA uses directed acyclic graphs (DAGs) to represent simple operations applied to some input constant. Since MS-EVA also uses MS-SEAL this means it also uses RNS-CKKS the purportedly most efficient CKKS implementation (Dathathri et al., 2020). MS-EVA has been applied to encrypted deep learning inference, specifically LeNet-5 towards MNIST. Dathathri particularly emphasises the non-trivial nature and how parameterisation can be a large barrier to the adoption of FHE. However there are no examples currently available to help lower this barrier. Subsequently their nodes representing single atomic operations means there is overhead when compared to block operations which could be an area of improvement.

4.3.3 Threat Model

Just like similar works in FHE we assume a semi-honest/ honest-but-curious threat model (Dathathri et al., 2020). Where parties follow the specified protocol but attempt to garner as much possible information from their received messages as possible. Or indeed one party has malicious intrusion which can read the data shared, but not necessarily write/ change the protocol.

4.4 Basic Concepts

As a necessary pre-requisite there is some prior understanding about FHE that is necessary but not broadly well known in particular when applied to deep neural network graphs that are often seen in the field of deep learning. We would like to highlight those here to make it clear in other sections how we overcome these limitations and highlight the advancements we make here. We would also like to note that FHE as a concept is distinct from any specific implementation scheme as we have previously eluded to. In our case the scheme we use is the CKKS scheme as previously stated and described, however following is some further information that applies to this scheme:

(i) Two cyphertexts that operate together must be identical containers; Same scheme, the same size, the number of primes into their swapping chain, and they must originate from the same private key.

(ii) Additions double the noise of a cyphertext whereas a multiplication exponentially increases the noise, which means to reduce the noise we must consume an element in our swapping chain to reduce the noise again. Since multiplication is much noisier than addition we tend to only swap after multiplication.

(iii) Abelian compatible operations are the only operations that can occur on an FHE cyphertext. This means addition and multiplication. There are methods to model division and subtraction but these operations are impossible under FHE. Thus the need to create new methods and algorithms.

(iv) Cyphertexts size and number of primes in the swapping chain are related. The bigger the cyphertext the more primes it contains for swapping. However the bigger the cyphertext the longer the computation takes. Thus we want the smallest possible cyphertext that has enough primes to complete the set amount of computations.

(v) Cyphertexts of a larger size also contain more slots, these slots are what are use to store our message or input/ plaintext data. Thus we must also consider that to store a certain number of features we must have a certain sized cyphertext. The CKKS scheme has half the number of slots compared to other schemes for the same size since it models pre and post point fixed precision.

(vi) Once the swapping chain has been consumed a very expensive operation called bootstrapping is necessary to refresh the cyphertext and regenerate the swapping chain to continue to do noise-expensive operations.

(vii) If the cyphertext is too noisy at the point of decryption it will lose precision or if even more noise is present the decrypted message/ data will become garbled and incorrect.

All of these points must be considered in the implementation of FHE compatible neural networks, and this is the primary reason why most existing work in the deep learning field is unfit for use under FHE including existing deep learning libraries.

We would also like to highlight as a consequence that there is little work in the domain of FHE deep learning with which to compare to and draw techniques from.

4.5 Material and Methods

To enable this research it was necessary to create our own python-based FHE compatible deep learning library because there was still a significant lack of compatibility between existing deep learning libraries and existing FHE libraries. While it may be possible to create some form of interface or bridge this left much to be desired in terms of usability and flexibility to explore different research avenues like various FHE backends. As a consequence we created a NumPy API focused library, where

the inputs to the neural networks need only conform to the basic NumPy custom containers specification, allowing the objects passed in to handle their own nature. This means any NumPy conforming object can be used in our networks, this includes NumPy itself (for pure plaintexts) or in this case arbitrary FHE objects. Our research here focuses on CPU computations as compatibility with existing CUDA implementations is currently infeasible due to compatibility which means conducting FHE over GPUs would be extremely difficult at this time. Encrypted deep learning accelerated by graphics processing units (GPUs) is an area we seek to explore in the future, for the rest of this chapter however all operations are conducted on CPUs. Our entire source code for our library Python-FHEz is available online along with the respective documentation (Onoufriou, 2021). We use the MS-SEAL C++ library bound to python using community pybind11 bindings to provide us with the necessary FHE primitives which we then wrap in the NumPy custom container specification for the aforementioned reasons (Zhigang Chen, 2021).

Furthermore in this section we outline our specific implementation, techniques, equations, and methods used to exemplify EDLaaS in practice using both an open dataset, and a preview of more real-world/ complicated but proprietary data scenario. We do this to enable some comparisons to be drawn and to introduce an new way of solving problems encountered in the agri-food industry:

(i) We chose to use Fashion-MNIST, consisting of a training set of 60,000 examples and a test set of 10,000 examples as our classification example as it is a drop in replacement for the MNIST dataset while being more complex, but still familiar to most.

(ii) We also chose to use an agri-food but proprietary dataset to exemplify a different kind of regression network and how FHE might play a role in this sensitive industry where data sharing/ availability is scarce due to a barrier in concerns over competition, of which FHE might help reduce (Pearson et al., 2019). Agri-food is also a key industry which has had a troubled few years due to climate change bringing hotter/ record-breaking summers, while also being effected by both coronavirus and Brexit shortages in staffing and thus supplying. In addition, it has been established that data sharing is a hindering factor that

prevents machine learning technologies from being adopted at scale (Durrant, Markovic, Matthews, May, Leontidis et al., 2021) but some work has already been done around using federated learning to alleviate some of these issues (Durrant, Markovic, Matthews, May, Enright et al., 2022).

For our neural networks we used a node-centric, multi-directed graph approach where:

(i) Each node represents some computation object usually a neuron.

(ii) Each edge represents the movement of data between neurons/ computation objects.

(iii) Each node can accept many inputs that are stacked on top of each other in the same order as the edges, unless there is a single input edge where it is instead mapped to the input of the neuron.

(iv) Each edge can only connect two nodes directed from the first to the second node, parallel edges are possible and are treated as completely separate edges with no special handling.

(v) A node can only be activated/ computed once all predecessor edges carry some data.

(vi) All nodes can have several receptors, that is to say different functions that can be pointed to by the edges, in particular forward and backward receptors for calculating the forward neural network pass, and gradients using the chain rule in the backward pass.

(vii) Nodes return either an iterable to be equally broadcast to all successor edges or a generator to generate independent results for each successor edge.

(viii) The weight of each edge corresponds to the computational depth of the directed-to node. These weights are not used to optimise the path since the majority of nodes must be activated to achieve some desired output, but instead these weights are used to find the longest path between key-rotations to determine

the minimum required encryption parameters to traverse from one rotation to the next.

(ix) Self-loop edges are not treated differently, instead relying on the configuration of the node itself to consider termination of the loop.

(x) A single activation pass of the graph may have multiple input and multiple output nodes/ neurons, like in the two blue regions in the sphira graph (Figure: 4.5).

We do this from the node perspective as we find this to be more conceptually clear and follows our own mental abstractions of how neural networks operate. This makes it easier for us to conceptualise, implement, and communicate our neural networks, in particular visually.

To activate our neural network graph we used our own neuronal-firing algorithm (Algorithm: 1), since we could not find better existing algorithms that would be suitable for firing of encrypted neuron graphs, while offering us the flexibility to adapt to changing our research.

Algorithm 1 Neuronal-Firing, our exhaustive neuron stimulating, depth-first, blocking, node-centric, graph/ neuron stimulation function.

Require: g: Neural network multi-directed computational graph
Require: n: Vector of neurons/ computational nodes for sequential stimulation
Require: s: Vector of signals to be induced in the corresponding neuron
Require: r: Vector of receptors to call on respective node
Ensure: g': Stimulated NN/ modified computational graph
  for $i \leftarrow 0$ to $\textsc{length}(n)$ **do**
    $\textsc{signal\_carrier}(g, n[i], r[i], s[i])$

4.5.1 FHE parameterisation

Our automatic FHE parameterisation approach is similar to that of MS-EVA (Dathathri et al., 2020) where we use (in our case our existing neural network) graphs to represent the computation the cyphertexts will experience. This allows us to automatically generate the smallest secure cyphertext possible that meets the requirements of the proceeding computational circuit. How we differ however is that since we are using neural network neurons instead of atomic (addition, multiplication, etc) operations,

Algorithm 2 Neuronal-firing-signal-carrier; Propagate a single signal thought all possible nodes in the neural network graph recursively based on its position.

```
function SIGNAL_CARRIER(g, n, r, bootstrap)
    s ←GET_INBOUND_SIGNAL(g, n, r, bootstrap)
    if s = None then
        return None
    s ←APPLY_SIGNAL(g, n, r, s)
    if s = None then
        return None
    SET_OUTBOUND_SIGNALS(g, n, r, s)
for all successors in g.node(n).successors() do
    SIGNAL_CARRIER(g, n, r, None)
```

Algorithm 3 Calculate accumulated inbound signal from edges.

```
function GET_INBOUND_SIGNAL(g, n, r, bootstrap)
    if bootstrap ≠ None then
        return bootstrap
    s ← [ ]
    for all edges in g.in_edges(n) do
        s.append(edge.signal(r))
    if length(s) = 1 then
        return s[0]
    return s
```

there are fewer nodes and edges, and thus less overhead necessary of both the graph, and any intermediate storage along edges. This is because we can block-optimise at a higher level that would be possible if purely considering individual atomic operations. Also our neural network graphs are Multi Directed Graphs (MDGs) as opposed to Directed Acryclic Graphs (DAGs) which means we can model more complex operations involving more than two inputs. This affords us the ability to model the complex relationships in neural networks much like standard deep learning libraries.

In our abstraction, automatic FHE parameterisation becomes a variation of the travelling-salesman problem, but instead of finding the shortest path we need to find the longest possible path or more specifically the highest computational depth experienced by the cyphertext, between sources and sinks. However, even in our abstraction, we must still conform to the constraints of CKKS, i.e. interacting cyphertexts must match, in cyphertext scales, and must be originating of the same private key which means other adjoining paths must be considered where they intersect. A key

Algorithm 4 Activate current node using the accumulated signal and get outbound signal.

```
function APPLY_SIGNAL(g, n, r, s)
    if s = None then
        return None
    s ← g.nodes(n).receptor(r, s)
    return s
```

Algorithm 5 Set outbound edges with activation signal.

```
function SET_OUTBOUND_SIGNALS(g, n, r, s)
    if s = None then
        return None
    for all edges in g.out_edges(n) do
        if isinstance(s, generator) then
            edge.signal(r) ← next(s)
        else
            edge.signal(r) ← s
```

distinction compared to MS-EVA's approach is that our graphs are interpreted instead of being compiled down to some intermediate representation. Our cyphertext objects are also not raw, and are instead part of a larger NumPy-API compatible objects that interpret invocations. These meta objects are also responsible for the decision making of both relinearisation and re-scaling, taking that complexity away from the implementation of encrypted deep learning. An example of this rescaling interpretation is when two cyphertexts are multiplied, the meta-object is responsible for ensuring both cyphertexts match, i.e. swapping down the modulus chain to equal scales depending on which of the two cyphertexts is higher up the modulus switching chain. Similarly an example of relinearisation is when two of our meta-objects are multiplied the computing member (usually the first meta-object in sequence) automatically relinearises the new meta-object, before passing the new meta-result back. This means we offload re-scaling and relinearisation, and it is not necessary to plan for these two operations, instead we need only calculate the longest paths, and the "groups" of cyphertexts. Here, groups of cyphertexts means cyphertexts that interact, and must then share encryption parameters.

In short the minimum necessary information we need to derive from the graph using our algorithms (Algorithms: 6, 7) is:

Algorithm 6 Automatic FHE-parameterisation by source and cost discovery, over multi-directed graphs.

```
function autoHE(g, n, concern)
    for i in n do                                   ▷ Label graph sources and costs
        autoHE_discover(g, i, i, concern, 0)
    r ← tuple(dictionary(), list())                 ▷ Group representation
    for i in n do                                   ▷ Assign + merge groups from labels
        if r[0].get(i) is None then
            r[0][i] ← len(r[1])
            r[1].append(0)
        for j in g.nodes() do
            src ← j[1]["sources"]
            if i in src then
                for k in src do
                    r[0][k] ← r[0][i]
                    if src[k] > r[1][r[0][i]] then
                        r[1][r[0][i]] = src[k]
    return r
```

(i) Which cyphertexts interact at which nodes

(ii) Thus which nodes belong to which group

(iii) What is the maximum computational depth of each group necessary to go from one (type-of-concern) source to another (type-of-concern) sink/ rotation

Each of our nodes must be labelled with its computational depth, so that the highest-cost traversal can take place. This may need to occur multiple times in a single graph, depending on the number of sources and sinks in said graph. Take for instance x_0, and x_1 in the dummy network depicted in Figure: 4.3. The cyphertexts x_0 and x_1 passed in must be able to reach the end of both paths leading to r_0 the very next sink/rotation. To do this they must be inter-operable with each-other at the point at which they meet. This means they must have matching scales, encryption parameters, and must originate from the same private key. However consider that x_1 experiences computations c_0 and c_1 whereas x_0 only experiences c_1. Each computation changes the scale, and thus necessarily their remaining primes in the modulus switching chain which would make them inoperable if not for our specialised logic in the meta-object to match them automatically. For instance, spatial and temporal data in the case of multi-modal datasets (of which Fashion-

Algorithm 7 Recursive FHE-parameterisation source, and cost discovery over multi-directed graphs.

```
function autoHE_discover(g, n, s, concern, c)
    d ← g.nodes().get(n)
    if d.get("sources") = None then
        d["sources"] ← dict()
    if s ≠ n then
        if d["sources"].get(s) = None then
            d["sources"][s] ← c
        else if d["sources"].get(s) < c then
            d["sources"][s] ← c
    if isinstance(d["node"], concern) then
        autoHE_discover(g, n, n, concern, 0)
    else
        for i in g.successors(n) do
            nxt ← c + g.nodes()[i]["node"].cost()
            autoHE_discover(g, i, s, concern, nxt)
```

MNIST is not) would have multiple inputs that require matching. Since decisions on relinearisation and rescaling are left to the meta-object the only information we need to ordain from the graph is the computational depth, and co-dependency of parameters. This can then be used to associate parameters together and select the minimum viable polynomial modulus degree.

In our node centric view of the graphs we say an edge from node A to B has the cost associated with B. This algorithm should be able to handle multiple cyphertext ingress nodes (x, y, etc), multiple cyphertext egress nodes ($\hat{y}$, and any others), and key-rotation stages in-between that will also need to be parameterised along the way. Our proposed algorithm can be seen in Algorithm: 6. The output of this algorithm is a tuple representation of the graphs parameterisation-groups. We will know which nodes need to share parameters, and what the highest cost of that parameter-group is. If we combine this graph parameter representation and some basic logic, we can tune/ parameterise automatically. This will of course vary for each implementation of FHE, from CKKS to BFV for example, requiring different parameters. The difference in parameterisation is why we separate out this final step, so that custom functions can be injected.

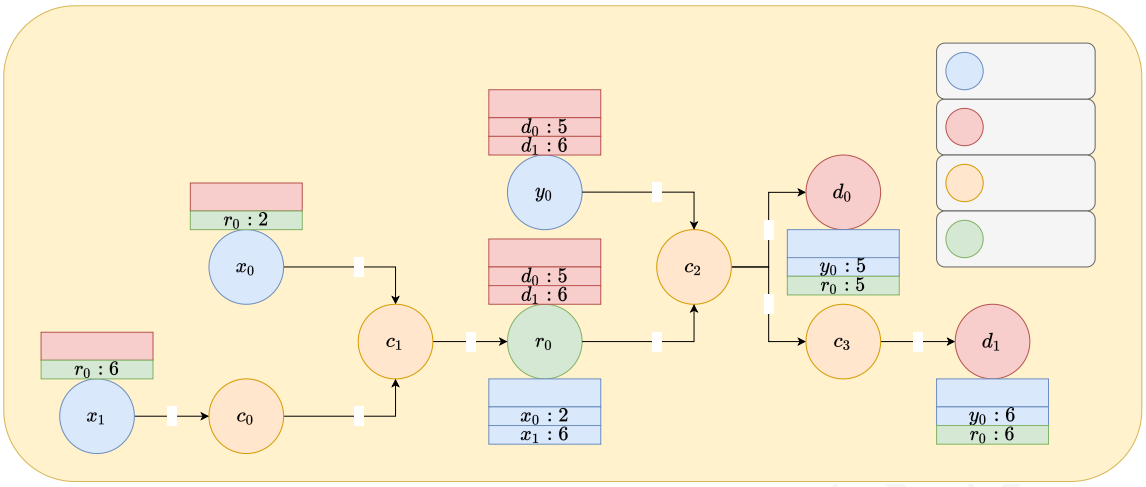

Figure 4.3: Example automatic FHE parameterisation problem, over a multi-directed graph. Sources are where data becomes a cyphertext. Sinks are where cyphertexts become plaintexts. Computation nodes are generic nodes that represent some operation that can be applied to both cyphertexts and plaintexts. Explicit rotation nodes are where a cyphertexts keys are rotated, either to refresh them, or to change the form of the cyphertext, potentially into multiple smaller cyphertexts. Please note this does not necessarily follow the colour coding of our other automatically-generated graphs (Onoufriou, 2021).

The FHE parameters we deal with here primarily geared toward the MS-SEAL CKKS backend are:

(i) Scale; computational scale/ fixed point precision

(ii) Polynomial modulus degree; polynomial degree with which to encode the plaintext message, this dictates the number of available slots, and the available number total bits which the coefficient modulus chain can contain.

(iii) Coefficient modulus chain; a list of byte sizes with which to switch down the modulus chain, this dictates the computational depth available before bootstrapping or key-rotation is necessary.

However the information we derive from the graph is generic and can be broadly adapted to generate parameters for other schemes also.

We use the default 128-bit security level of MS-SEAL just as MS-EVA (Dathathri et al., 2020), being the most similar existing framework. This is security level is broadly considered reasonably secure (J.-W. Lee et al., 2021; Meftah et al., 2021; Dathathri et al., 2020), and matches our threat model of honest-but-curious.

Lastly now that we have calculated the groups, the cost of the groups, and the asso-

Algorithm 8 Heuristically parameterise RNS-CKKS scheme using expected cost of computation.

Require: c: Integer maximal-cost of this cyphertext group.
Require: s: Integer scale-power, the scale of the cyphertext. Default: $s = 40$. We advise not to go below 30 due to noise accumulation and lack of prime availability.
Require: p: Float special-prime-multiplier, the multiplier that dictates the scale-stabilised special-primes in the coefficient-modulus chain. Default: $p = 1.5$
Ensure: *parms*: MS-SEAL RNS-CKKS parameter dictionary/ map.

```
function PARAMETERISE(c, s, p)
    parms ← dict()
    parms["scheme"] ← 2                          ▷ 2 is CKKS in MS-SEAL
    parms["scale"] ← pow(2, s)                   ▷ scale power
    m ← [s for i in range(c + 2)]
    m[0] ← int(m[0] * p)                         ▷ Mult first special prime
    m[−1] ← int(m[−1] * p)                       ▷ Mult last special prime
    b ← 27
    while b < sum(m) do
        b ← b * 2
    parms["poly_modulus_degree"] ← int(1024 * (b/27))
    return parms
```

Figure 4.4: Fashion-MNIST sample showing examples of data such as: boots, bags, jumpers, and trousers (Xiao, Rasul and Vollgraf, 2017).

ciated nodes that belong to which groups, we can use a rough heuristic (Figure: 8) to estimate the necessary FHE parameters to accompany these groups. This heuristic can be tuned, and overridden for other FHE schemes to more tightly parameterise if necessary.

4.5.2 Open Data Fashion-MNIST

In this section we describe our openly available Jupyter implementation (Onoufriou, 2021) of an FHE-compatible CNN operating on the open dataset called Fashion-

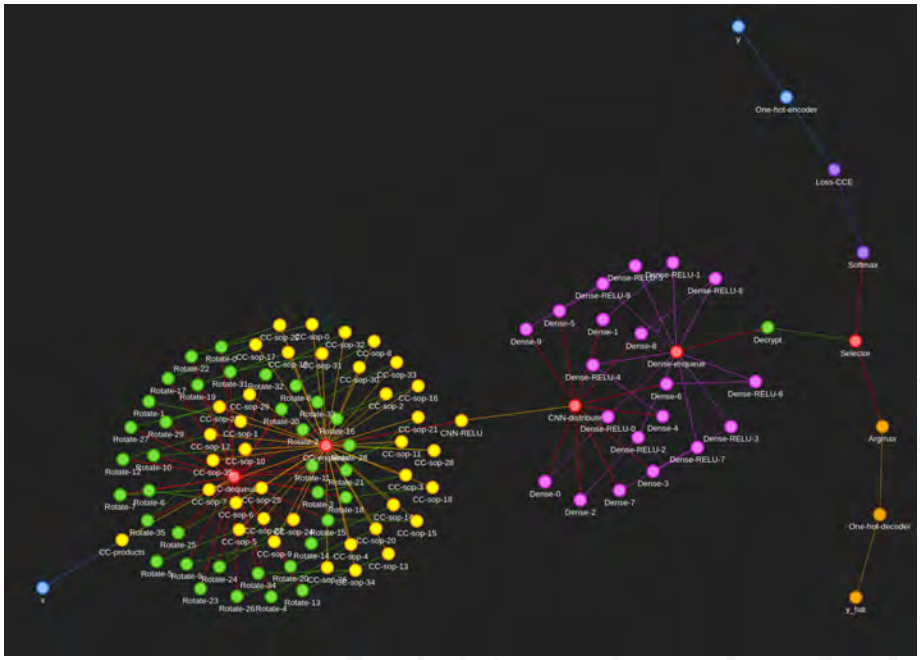

Figure 4.5: Fashion-MNIST computational graph we call "sphira", showing the colour coded graph and the respective nodes used to train/ compute Fashion-MNIST using our neuronal-firing algorithm. Blue represents the input and input transformation circuit that deals with passing the signals into the neural network in a way it is expecting them. Yellow represents the convolutional neural network components where one filter neuron passes multiple output cyphertexts to a plethora of summing nodes. Pink represents the fully connected dense layer for each class. Purple represents the loss calculation circuit necessary for backpropagation. Orange represents the output/ prediction circuit. Red represents the generic glue operations necessary to bind components together. Green represents the encryption specific nodes like decryption, rotation, encryption. An interactive version of this graph is available in our source code documentation so that clusters of nodes can be peeled apart for investigating individual nodes and connections. (Onoufriou, 2021)

MNIST as can be seen in Figure: 4.4. This dataset contains in total 70,000 images, 60,000 for training and 10,000 for testing. This dataset contains images of certain items of clothing, constituting 10 classes. Each image is a mere 28x28x1 pixels. The full implementation can be found in the examples of our source code repository. (Onoufriou, 2021)

We chose Fashion-MNIST as it is a drop in replacement for MNIST while also being a somewhat more difficult problem than standard MNIST. Coincidentally being that MNIST and thus Fashion-MNIST are both classification rather than regression they represent an even more difficult scenario for encrypted deep learning since they do not provide one continuous/ regressed output so the computational circuit becomes more complex/ deeper as far as necessary to process these classifications, i.e. the extra dense nodes for each class, and the whole addition of both softmax and categorical

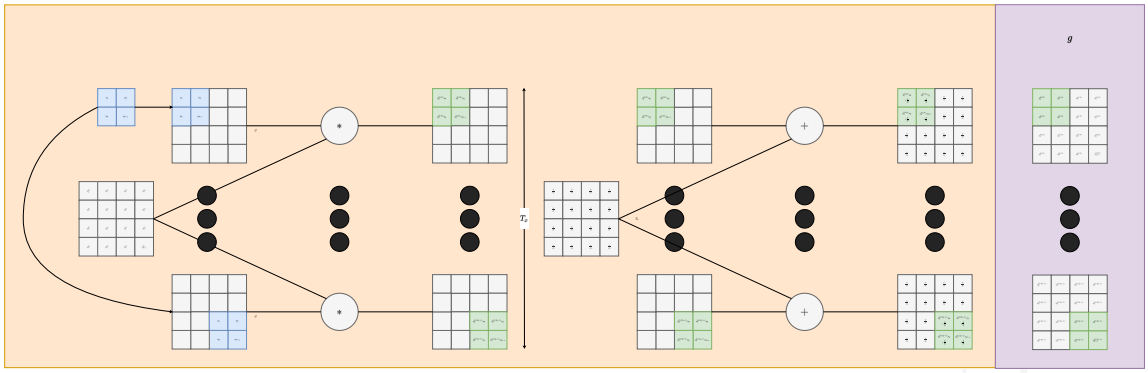

Figure 4.6: Encrypted CNN, this is a particular unusual implementation since there can be no summing of the filters, and instead this sum is commuted in the case where the filter operates on an input that is a single cyphertext (i.e. not a composite of multiple cyphertexts). Please see our documentation for closer detail (Onoufriou, 2021).

cross-entropy (CCE) to replace the mean squared error (MSE) loss function in the case of would be regression networks. This also poses a problem as methods usually used towards classification like softmax (Equation: 4.2) are not compatible with FHE since they include division although some alternative approximations do exist such as those used by Lee (J.-W. Lee et al., 2021).

Data Wrangling and Inputs

Fashion-MNIST is largely pre-wrangled especially if you use one of many forks of the data which present each figure-classification, and image as a one-dimensional feature vector between 0-255 stacked in a comma-seperated values (CSV) file. This means the only two necessary steps toward this data are to normalise between 0-1, and reshape the individual feature vectors back into their original shape of 28x28x1. The feature vector is encrypted and the cyphertext passed in as a signal to node "x" in the sphira network (Figure 4.5), and the figure-classification is passed in to node "y" as a separate signal. Whereby our neuronal-firing algorithm (Algorithm 1) will propagate these signals thereafter.

CNN

$$a^{(i)<t>} = g(\sum_{t=0}^{T_x-1}(k^{<t>}x^{(i)} + b/N)) \tag{4.1}$$

As our CNN (yellow in Figure: 4.4) we use a biased cross correlation layer (CC)

to calculate the product of a given filter against the input cyphertext. We use a SIMO scheme we call kernel masquerading. Here kernel-masquerading shall mean the merging of weights and a respective zeros mask into a sparse n-dimensional array such that they become a single operation conducted on the input cyphertext (Figure: 4.7), reducing the computational depth experienced by the input cyphertext to 1 (multiplication) and allowing for subset operations to be conducted on the cyphertext to selectively pick regions of interest. This is only possible in the plaintext-weights strategy, since this allows the weights to be operated on arbitrarily and selectively to reform them into the shape of the input cyphertext and sparsity of the filter/ kernel. This is a simple operation of which a two and three dimensional variant can be seen in Juvekar's, and Meftah's work (Juvekar, Vaikuntanathan and Chandrakasan, 2018; Meftah et al., 2021). The main drawback of the kernel-masquerade is that if we were to apply a convolutional-kernel-mask on some cyphertext $\varepsilon(x^{(i)})$ we would end up with separate modified cyphertexts $\varepsilon(x^{(i)<t>})$ that correspond to different portions of the data, however we are unable to sum them without a key rotation such that we are summing between different cyphertexts since we cannot fold a cyphertext in on itself. This means we have a choice at this stage, we can either rotate the keys now to reduce complexity or try to save computation time by doing as much processing while the values are encoded in one larger cyphertext which is significantly more efficient from findings in Juvekars work on SISO cyphertexts (Juvekar, Vaikuntanathan and Chandrakasan, 2018).

Since key rotation would make the outputs normally-processable for operations like summation we wont address that variant here instead we choose to see how far we can instead commute this sum to get the maximum performance as far fewer cyphertexts. One thing we can and did do in our CNN implementation is to commute the bias forward to be before summation. so instead of $z = \sum_{i=0}^{N}(x_i w_i) + b$ we decompose b into the product calculation before summation as $z = \sum_{i=0}^{N}(x_i w_i + \frac{b}{N})$ since this is equivalent over the full computation of the cyphertext. We could have simplified to just $z = \sum_{i=0}^{N}(x_i w_i + b)$ if we calculate the gradient with respect to the bias $\frac{df}{db}$ as $\frac{df}{db} = Nx$ instead of $\frac{df}{db} = x$ such that the neural network is effectively aware of this

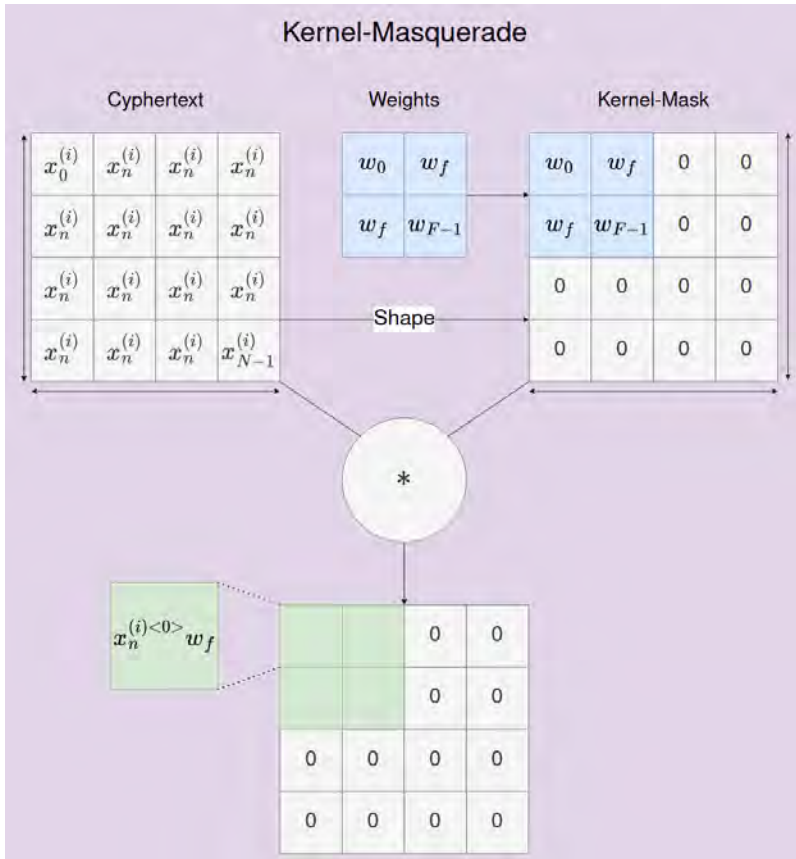

Figure 4.7: Merged mask and kernel together to create a single sparse kernel which zeros undesired components in the cyphertexts polynomial of values using Hadmard products. Please see our documentation for closer detail (Onoufriou, 2021).

higher contribution of the bias, and it would be naturally accommodated through the gradient descent process.

From here forward special logic/ considerations need to be made to ensure the output cyphertexts of the biased-cross-correlation are treated as a singular un-summed-value. We tried to push this cyphertext through the neural network further but we had to ensure all further operations were both linear and abelian compatible. Take for instance an encoded non-summed sequence as a cyphertext x, $x = (1 + 2 + 3 + 4) = 10_{\text{plntxt}}$, then lets try a multiplication $4x = 4(1 + 2 + 3 + 4) = (4 * 1 + 4 * 2 + 4 * 3 + 4 * 4) = (4 + 8 + 12 + 16) = 4 * 10_{\text{plntxt}} = 40_{\text{plntxt}}$, but now lets try a multiplication against itself or another non-summed sequence so for instance a nonlinear $x^2 = (1+2+3+4)(1+2+3+4) = (1*1+1*2+1*3+1*4+2*1+2*2+2*3+2*4+3*1+3*2+3*3+3*4+4*1+4*2+4*3+4*4) = 10^2_{\text{plntxt}} = 100_{\text{plntxt}}$.

This is a problem since we cannot cross-multiply cyphertexts since we cannot select

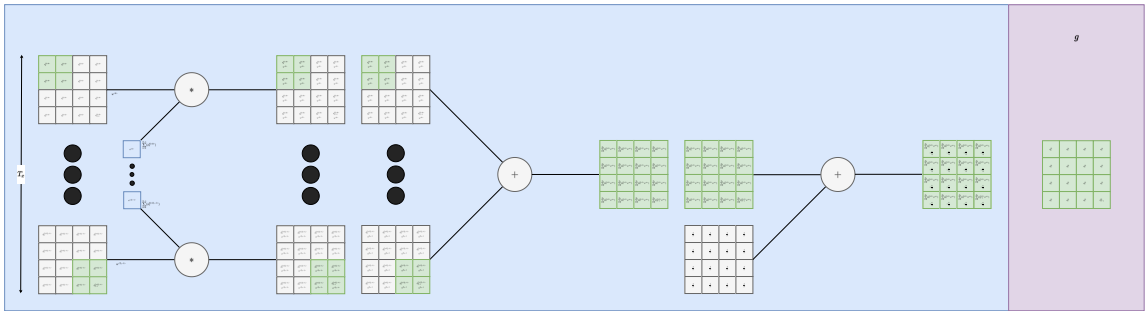

Figure 4.8: Encrypted variant of an artificial neural network (ANN)/ dense neural network, usually used in our case to merge divergent times/ branches/ filters back together into a single output. Please see our documentation for closer detail (Onoufriou, 2021).

elements from either cyphertext, if we were to attempt to multiply this cyphertext with itself it would calculate the element-wise product of the two. The best we could do if we did want to compute this would be to conduct a key rotation to expose the elements we desired as separate cyphertext but if we are going to do that we would be just as well served by just rotating to sum then passing it through the element-wise product as normal. It is possible to commute the sum further if we use linear approximations of our activation functions like if we take Sigmoid (Equation: 4.3) and its approximation Equation: 4.5 then if we ensure our products will always be between 0-1 through a modified version of batch norm ((Meftah et al., 2021)) we could then safely use only the linear component of the Sigmoid approximation (Equation: 4.5) $\sigma(x) \approx \sigma_a(x) = 0.5 + 0.197x$ since it would still closely follow in the -1 to 1 range and loses approximation beyond this range instead of the usual -5 to 5 range the full approximation affords and would also cut down the computational cost. For ourselves we choose to key rotate to encrypt the elements to be summed until such a time as we have fully fleshed out a fully-commuted-sum alternative.

For our cross-correlation activation function we use the more recent ReLU (Equation: 4.4) approximation (Equation: 4.6) and the derivative of this approximation for backward propagation (Equation: 4.10) as its own separate node to allow them to be decoupled and easily swapped out with new or improved variants and so interjection with batch norm is readily variable without having to rewrite existing nodes.

Dense/ ANN

For each of our classes we have a dense fully-connected neuron (Figure: 4.8, red in Figure: 4.4) to interpret the activation vector of the biased-cross-correlation and activation combination / CNN. Thus our dense layer is comprised one 10 ANN nodes. There is nothing of any note in this layer other than it must accept multiple cyphertexts that are added together / summed across the first axis, otherwise it behaves almost the same as a standard neuron as depicted in figure. However careful attention should be paid to broadcasting such that the gradient is still correct and we do not attain an exploded result that could fall outside of approximation golden zones like sigmoids -5 to 5.

We accompany each neuron with its own ReLU approximation node before passing the activations on to the different forward evaluation circuits for loss calculation and prediction output.

Prediction

Argmax is an effective and quick computation of the highest value in a vector. Since the ANN layer outputs a vector of 10 values one for each class the Argmax function serves to take the highest activation and turn it into a 1-hot-encoded representation of the predicted class. This can be passed into a 1-hot-decoder to attain the predicted class $\hat{y}$. However since argmax relies on the context to find the max, it is necessary to conduct this operation in plaintext on the client side, to effectively pick from this 10 element vector. There is no backpropagation from this branch, it is purely an output branch for providing predictions to the data owner. These stages are pink in Figure: 4.5.

Loss

The loss calculation stage is represented by purple in Figure: 4.5. Argmax is not an effective function for the purposes of backpropagation of the loss since only one of the ten input ANN neurons would receive all of the gradient multiplied by 1, which does not give the majority of the network much information to update the weights

from any single given example. Thus as-per-norm we used a Softmax layer instead which better distributes the gradient between not only what neuron was responsible for positive activation but also the others that should not be activating.

The softmax ensures that all output values summed together equal 1, and that they are effective predicted probabilities of the network that a certain class is what was given in the input. We use a standard CCE function to calculate the loss and subsequently the derivative with respect to each of the 10 classes to pass back to the softmax and hence the ANN layer.

The CCE function also receives input/ stimulation from a 1-hot-encoder that encodes the ground-truth y value or the actual class that the input x corresponds to for the purposes of loss calculation.

$$\sigma(\overrightarrow{a})_i = \frac{e^{a_i}}{\sum_{j=1}^{K} e^{a_j}} \tag{4.2}$$

It should also be noted that the loss circuit (pink) requires decryption since both softmax and CCE are not FHE compatible operations. There have been proposed ways to allow for softmax to be computed with cyphertexts by Lee (J.-W. Lee et al., 2021) however we were unable to create a working fhe-compatible softmax from what information was available and would require bootstrapping 22 times, and it would still need to be unencrypted for the CCE calculation. Given this data using the sphira (4.5) network we garnered the following results:

4.5.3 Strawberry Yield Data

Unfortunately we do not have permission to publicise the specific data used in this section. As such we shall preview this application and seek to further elaborate and develop the techniques used here in-depth in future papers. We will only touch briefly here of this data as a means to show how FHE can be used in real world problems effectively, and for a different kind of problem; regression instead of classification.

This data is geared towards yield prediction often weeks in advance. The prediction horizon is typically between one to three weeks ahead of the strawberry fruit becom-

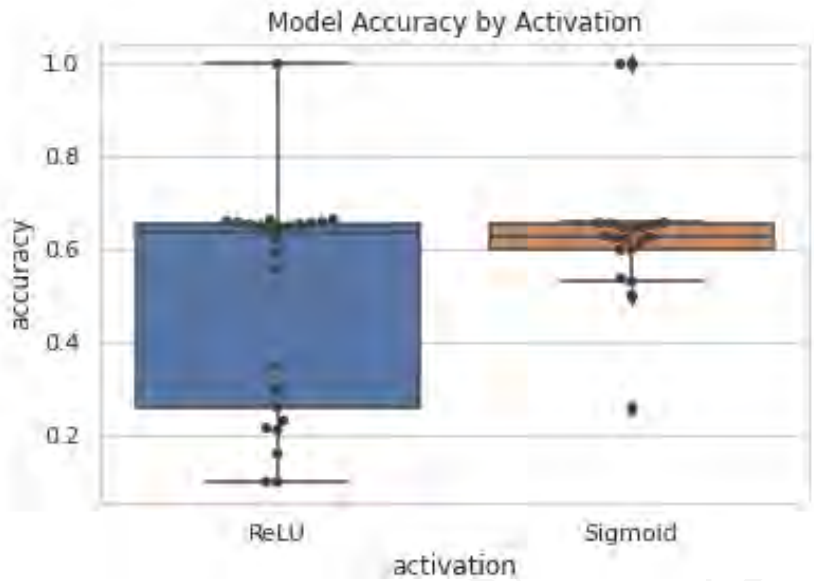

Figure 4.9: Model performance using different activation functions in the sphira network on the fashion-MNIST dataset. All activations here are their FHE compatible approximations unless otherwise specified. Each dot is a different network, or the same network with a different data type(cyphertext, plaintext) (Onoufriou, 2021).

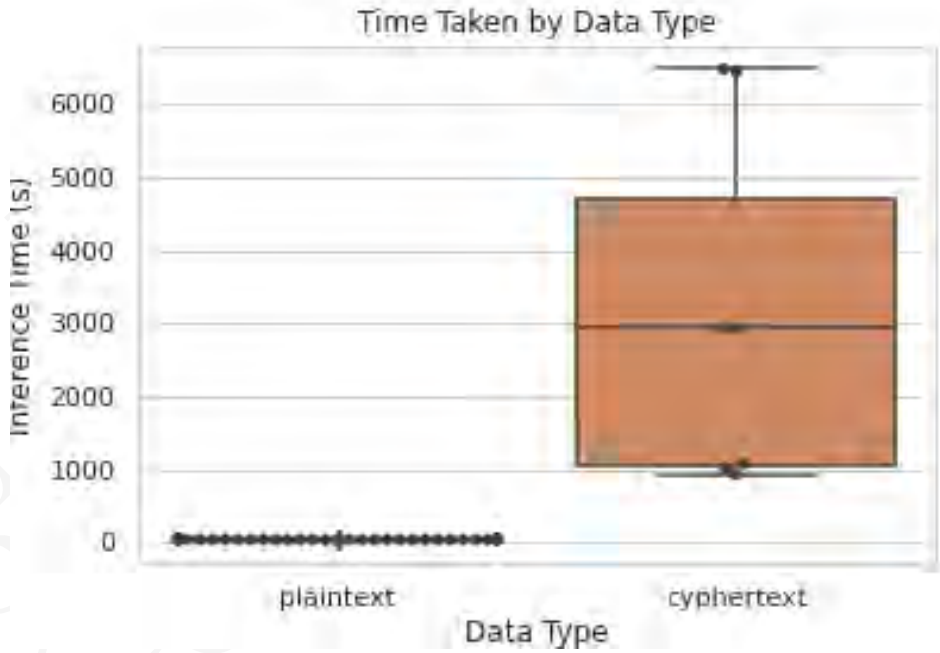

Figure 4.10: Model inference time, by different types. Plaintext types mean where the graph is run using plaintext data. Cyphertext types mean where the graph is run using cyphertext data. Both plaintext and cyphertext data conforms to the same NumPy API, meaning they can be used interchangeably. Each dot is a different network (i.e. differently initialised weights but the same structure), or the same network with a different data type (cyphertext, plaintext) (Onoufriou, 2021).

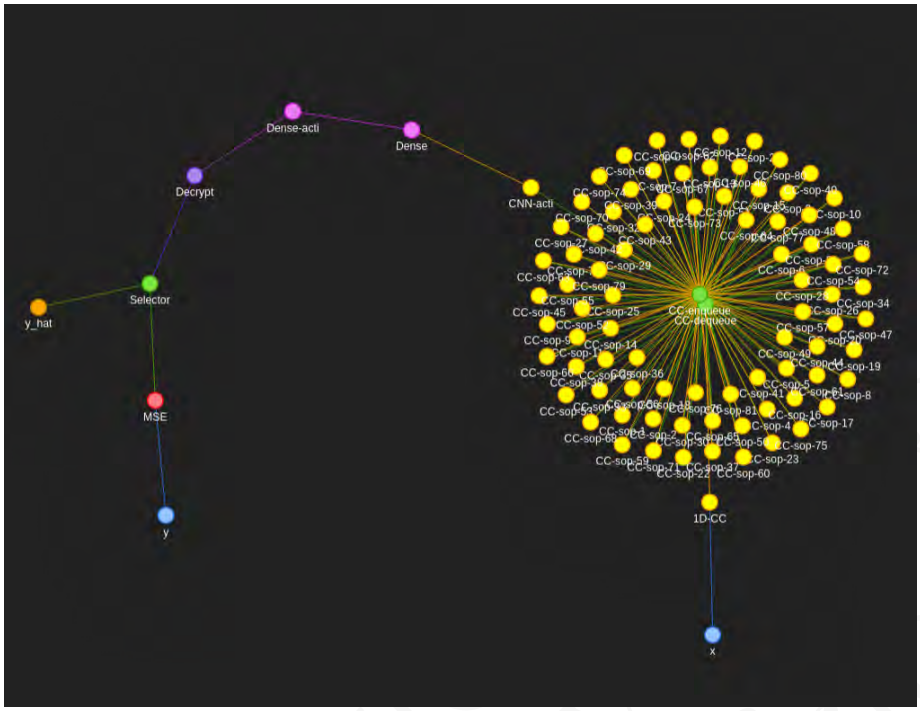

Figure 4.11: Strawberry yield/ regression computational graph we call 'constellation', showing the colour coded graph representation and nodes used to train on strawberry yield, based on environmental factors. Blue are input/ encryption nodes. Yellow are convolutional-related nodes. Green are operational nodes necessary to "glue" the network together. Pink are dense/ ANN nodes. Orange is the output prediction node. Red is the loss calculation node. Purple is an FHE specific node used for decryption of the input data. Please see our documentation for closer detail (Onoufriou, 2021).

ing ripe. This allows time for logistical constraints such as price negotiation, and picker/ staff scheduling. Thus performance of these predictive models is critically important to ensure all the fruit is being accounted for in negotiations with retailers (and thus can be sold), and that there is sufficient manpower at the point of need to gather this produce. Over predicting can result in insufficient harvests to meet contractual obligations, most likely meaning the yield must be covered by buying other producers yields. However if there is a shortfall of yield from one producer, the factors that lead to that shortfall such as adverse environmental conditions are felt by most other producers in the geographic region. This means that usually (in the UK) the yields must be imported from abroad increasing the price substantially. Conversely if there is under-prediction of yield this results in unsold fruit, which is either sold at a significant discount if possible or destroyed. There is also a clear and significant lack of agriculture data available due to perceived data sensitivity. This affects many forms of agriculture for various reasons. In the soft-fruit industry this

Days Ahead	Mean Absolute Percentage Error (MAPE)
7	8.001
14	14.669
21	22.326

Table 4.1: Table of predictive results of the constellation network (Figure: 4.11) predicting strawberry yield.

tends to be proprietary genetic varieties, and operational specifics such as irrigation nutrition mixtures. However there is a tendency to distrust and a perceived lack of benefit to data sharing, due to no obvious performant outcomes.

Due to a lack of available data, we use historic yield data we gathered in our Riseholme campus polytunnel/ tabletop over two years, and combine this with environmental data experienced by these strawberries leading up to the point-of-prediction. The environmental data includes: wind speed, wind direction, temperature, light-intensity, humidity, precipitation, positions-of-strawberries, yield-per-row-of-strawberries, and many more less significant features. This data also includes irrigation data such as nutrition, soil-moisture, soil-temperature, irrigation-status. We normalised, one-hot encoded categorical variables and split the data (80-20) randomly into training and test sets. We then further subdivided the training set into validation sets for model selection purposes.

We applied a 1D/ time-series CNN (Equations: 4.1, 4.5 as depicted in Figure: 4.6) followed by a dense ANN, and Sigmoid again as depicted in Figure: 4.8) to summarise the feature vector and to predict the output/ yield of the strawberries. This prediction will then be based on the environment the strawberries experienced leading up to the point of prediction. Given our data, and our feature engineering, we were able to obtain the following outcomes in Table: 4.1.

4.5.4 Equations

Sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{4.3}$$

ReLU

$$R(x) = \max(0, x) \tag{4.4}$$

Sigmoid-Approximate

$$\sigma(x) \approx \sigma_a(x) = 0.5 + 0.197x + -0.004x^3 \tag{4.5}$$

ReLU-Approximate

$$R(x) \approx R_a(x) = \frac{4}{3\pi q}x^2 + \frac{1}{2}x + \frac{q}{3\pi} \tag{4.6}$$

Sigmoid-Derivative

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1+e^{-x})^2} = (\frac{1+e^{-x}-1}{1+e^{-x}})(\frac{1}{1+e^{-x}}) = (1-\sigma(x))\sigma(x) \tag{4.7}$$

ReLU-Derivative

$$\frac{dR(x)}{dx} = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

Sigmoid-Approximate-Derivative

$$\frac{d\sigma(x)}{dx} \approx \frac{d\sigma_a(x)}{dx} = 0.197 + -0.012x^2 \tag{4.9}$$

ReLU-Approximate-Derivative

$$\frac{dR(x)}{dx} \approx \frac{dR_a(x)}{dx} = \frac{8}{3\pi q}x + \frac{1}{2} \tag{4.10}$$

4.6 Results

As can be seen in Figure: 4.9 using the same network sphira (Figure: 4.5) with different approximated activation functions Sigmoid (Equation: 4.5) and ReLU (Equation: 4.6) dramatically effects the precision of the neural network over multiple training attempts with randomised weights. However the accuracy of both on average is

roughly equal within a few percent. This shows that in our implementation at least that this ReLU approximation being backpropagated may indeed cause some instability, more frequently. Sigmoid in contrast is a static approximation which may be part of the reason for its greater stability, and thus consistency provided randomised weights to the rest of the network. We can see that both the sphira (Figure: 4.5) and constellation (Figure 4.11) networks can produce acceptable results on the testing set while computing over cyphertexts and plaintexts. Our networks can be seen working in both classification and regression problems, Fashion-MNIST, and strawberry yield prediction respectively.

We find, however, that in our strawberry yield prediction one of the weaknesses of our approach was to completely randomise the sequences, as some sequences could possibly overlap, meaning that the network may have at least some prior experience of the gap between point-of-prediction and point-predicted. This should be an area of possible future expansion is to split the data differently by time, and use only environmental data in the future that is distinct. Another area where improvement could be garnered is by using smaller but bootstrappable cyphertexts, this may reduce predictive performance of the networks since each bootstrapping operation would incur a noise penalty, but this would significantly improve the speed of computation since we could use smaller cyphertexts that take less time to transverse, transmit, and compute. We can see from Figure: 4.10 that the time taken for computing plaintexts is relatively small, producing results rapidly. The same network however provided cyphertexts computes near equivalent results, but significantly slower. Not only are cyphertexts more time intensive, but they are also significantly more space intensive. We could see during cyphertext inference anywhere from 72-80GB of RAM usage, meaning this is certainly not plausible on low-specification machines. We could see significant gains in computational performance if we added more rotation nodes to refresh the cyphertext more frequently, to limit the number of levels it would contain, and thus the size of the cyphertext. However in our case we wanted to reduce the number of rotations as in practical applications this would result in more transmissions from client-server which itself can be an expensive operation. This is

a good example of why bootstrapping while incurring a high cost itself, could save computational time and space in the future, when it is more widely available.

The absolute performance of the two models in Figure 4.9 and Table 4.1 is acceptable despite being fairly shallow models compared to those used in many normal deep learning models. Our absolute performance is probably quite limited by the shallowness of these models, in that the model may not be complex enough to properly model some of these problems. In particular there are a plethora of standard models that achieve 90% accuracy or greater, many of which use 2-3 convolutional layers, with batch-normalisation, and max-pooling. Clearly we cannot ordain context from a cyphertext making max-pooling impossible however we can and do use strides as a way to reduce the dimensionality in a similar way that max-pooling does. There have also been proposals for batch-normalisation that involve multiplying by small fractions that are occasionally recalculated, however this is quite complex and not something we have been able to implement ourselves as of yet. This would however stabilise the activations between nodes, and reduce the likelihood of escaping the dynamically predicted range in the ReLU approximation causing the in-precision in Figure 4.9.

Given that we can get acceptable performance, in different scenarios like agricultural yield regression and image classification, this opens avenues for data sharing. There are two avenues in particular, through encryption, and through trust. In our case we assume a semi-honest threat model, yet we have outlined a way of computation that does not need to reveal any data to the third party. This means if we can provide sufficient predictive performance then there are few barriers preventing sharing of encrypted data for inference. There is of course the notable exception of any yet unknown vulnerabilities in the underlying FHE scheme with default parameters provided by MS-SEAL. The other avenue of data sharing that FHE fosters is that of trust. Given a track record of reliable data processing in the encrypted form, that this could lead to an increased awareness of the gains of deep learning applied to various fields. With this greater awareness, and track record, it could be surmised that it is more likely that over time the data owners might choose to share data in the perceived-sensitivity scenario.

4.7 Discussion

Considering the importance around ascertaining privacy when developing new machine learning methodologies, it is paramount that we start scaling up research on privacy-enabled machine learning. This should take place in tandem with showing how real-life problems, e.g. strawberry yield forecasting, can be tackled with such methodologies, which is what this paper has contributed to, results-wise as well (Table 4.1). Nevertheless, our work on encrypted deep learning has certain limitations we would like to highlight:

FHE Training; In this paper we laboriously implement, describe, and show how encrypted deep learning inference can be conducted. However there is little reference to encrypted learning, that is to say where a neural network is trained on cyphertexts. This is due to multiple limitations prevalent in the field such as the lack of FHE compatibility with certain functions, such as loss functions. This is an active area of research which we and the broader research community are actively working on, to complete the encrypted learning-to-inference chain. Another issue with cyphertext training for example is when do we decide to stop training? As we cannot see the results, and thus cannot gauge whether the network has under or over-fit. This is a particularly interesting and challenging problem which we seek to also tackle in future. Here however it can be thought that the models would be pre-trained or transferred from a similar problem to avoid privacy leaks involved in training.

LFHE; As previously mentioned our work here is over Levelled-FHE, where we create optimised circuits for cyphertexts with discrete scales and primes, which we swap down for each multiplication, a "level". LFHE is FHE without bootstrapping. Bootstrapping is an expensive operation that refreshes the cyphertexts levels allowing for an effective limitless depth to computations (albeit with noise), while also helping to keep cyphertexts smaller than their LFHE counterparts. Smaller cyphertexts can be operated on faster, but bootstrapping in small circuits can often outweigh the benefit of using a smaller but bootstrapped cyphertext, due to how expensive of an operation it is. This limitation comes from a lack of bootstrapping support in MS-SEAL. Once bootstrapping is supported however existing networks we propose

here will still be compatible assuming appropriate NumPy-container abstractions of FHE will be passed in.

State-of-the-Art Neural Networks; While this work particularly focuses on ANN and CNN neural networks, these are not current State-of-the-Art networks for many tasks. In particular in future we intend to continue to work applying FHE to existing networks such as transformers which are SotA in sequence tasks. However much work remains in mimicking certain functions of transformers in an FHE compatible manner (Falcetta and Roveri, 2022). We also believe while we could compute privately, we can significantly improve the performance of the predictions themselves with more performant network architectures like transformers. We could then draw more comprehensive comparisons between encrypted and unencrypted deep learning for yield forecasting and other applications.

4.8 Conclusions

In this paper, we have shown how FHE can be automatically parameterised directly from multi-directed graphs for neural networks, using groups and a variation of the travelling salesman problem for costs. It was demonstrated how multi-directed graphs can be used in an FHE compatible manner with FHE compatible nodes to facilitate encrypted deep learning. We have also evaluated a recent ReLU approximation (with additionally backpropagated approximation range), against the Sigmoid activation function, finding it slightly less accurate but much less precise due to instabilities in the weight initialisation. The proposed encrypted deep learning procedures were utilised in both classification and regression problems. For the former, we used an open dataset Fashion-MNIST with open-source reproducible code examples to aid reproduction and experimentation. For the latter, We demonstrated how our methods can be used in an real world (sensitive) problem predicting strawberry yield, paving the way to introduce such a technology at scale in the agri-food sector. We believe that our implementation is the most comprehensive encrypted deep learning library currently available, now with automatic FHE parameterisation, traversal, cross-compatible/ interoperable NumPy custom-containers, documentation and ex-

pandability for future distributed or GPU accelerated computations with FHE, using the state-of-the-art RNS-CKKS FHE scheme provided by the MS-SEAL back end.

However, there is still much research that needs to be conducted, in particular with FHE and training. Encrypted deep learning is not a solution currently to any problem that relies on very specific data that is very dissimilar to other problems, meaning we cannot transfer some understanding in a private manner. We are still limited by multi/parallel-processing, however in the case of Python-FHEz we leave the back end open-ended following the NumPy custom container specification such that this gap can be easily retrofitted later, just like Dask and CuPy have for standard NumPy.

Finally with encrypted deep learning we can open avenues for data sharing that have previously been untenable in the face of their rightful privacy concerns. The more of the pitfalls of FHE that are solved, and the more usable encrypted deep learning becomes, the more likely we are to see it provide some critical predictive service to improve fields like agriculture, and medicine.

Chapter 5

Yield Forecasting and Premonition

George Onoufriou, Marc Hanheide and Georgios Leontidis (2022b). 'Premonition Net, A Multi-Timeline Transformer Network Architecture Towards Strawberry Tabletop Yield Forecasting'. In: *arXiv preprint arXiv:2211.08177*

5.1 Introduction

Precise and accurate yield forecasting is a key component in fresh produce (FP) supply chain management (FSCM), since it plays a critical role in price negotiations, logistics, and scheduling. In particular accurate yield estimates are required a minimum of 3 weeks ahead (in the strawberry domain) which we call the horizon (Figure 5.1), so that adequate time can be given to bidding, labour timetabling, logistics, and procurement. However, forecasting FP is incredibly difficult especially over a 3-week horizon where any number of variabilities can exist such as environmental fluctuations. Often the quantities of fresh produce we seek to deal with make it impractical to expect climate-controlled greenhouse conditions, meaning there is an element of weather forecasting that is required however we do not expressly aim to forecast weather in this work as this is a separate and highly complex problem of its own. Instead, we show how good yield forecasting can be and improve upon current practices while allowing for future works to delve specifically into weather forecasting.

Yield forecasting is difficult in particular due to the in-availability of data with which to forecast, this data being mostly non-existent, or incredibly difficult to attain. We

believe the reasons why the data is unavailable is because of the difficulty of data collection, the perceived sensitivity with which this data is held, and the lack of clear benefits to the digital collection of such data. We also see resistance to the positive dynamic impetus of modernisation requiring a departure from growers' previous fixed practices.

FP optimisation is of global strategic importance since horticulture and agriculture are some of the biggest producers of greenhouse gasses, such that there can be a significant benefit to optimising production or minimising waste. In the UK our government has committed to reducing greenhouse gasses to net 0 by 2050, and agriculture has been expressly named as a key contributor of greenhouse gasses in the United Nations Climate Change Conference 2021 (COP21). Inaccurate forecasting or more specifically under/ over estimation leads to food waste and destruction costs or importing of FP from abroad. Assuming the cause of this discrepancy/ variability is adverse weather conditions, then those same weather conditions will have affected geographically approximate growing sites. In the UK climate discrepancies usually mean fruit must be imported from abroad, given our size, to meet any given procurement contract, as all the neighbouring growing sites will have suffered the same adverse environmental conditions and thus under-production.

Other works (outlined with more detail in Section 5.3) have sought to solve the lack of data availability in agriculture using satellite/ remote-sensing data, using various machine learning, statistical, and some deep learning techniques. In this paper we show how we can collect data at some scale but with local/ high granularity, including fruit images, weather conditions, and irrigation data locally. Here we shall focus specifically on strawberry yields of strawberry tabletop and how we can predict them. We exemplify this approach at our Riseholme strawberry tabletop/ polytunnel growing site and employ this data to create accurate forecasts with this 3-week horizon/ window to meet the needs of the bidding and procurement process. We do all this in collaboration with Berry Gardens Growers (BGG), one of the UK's largest soft, and stone fruit producers, and with their direction on industry standards to keep as close to the typical expectations as reasonably possible. We

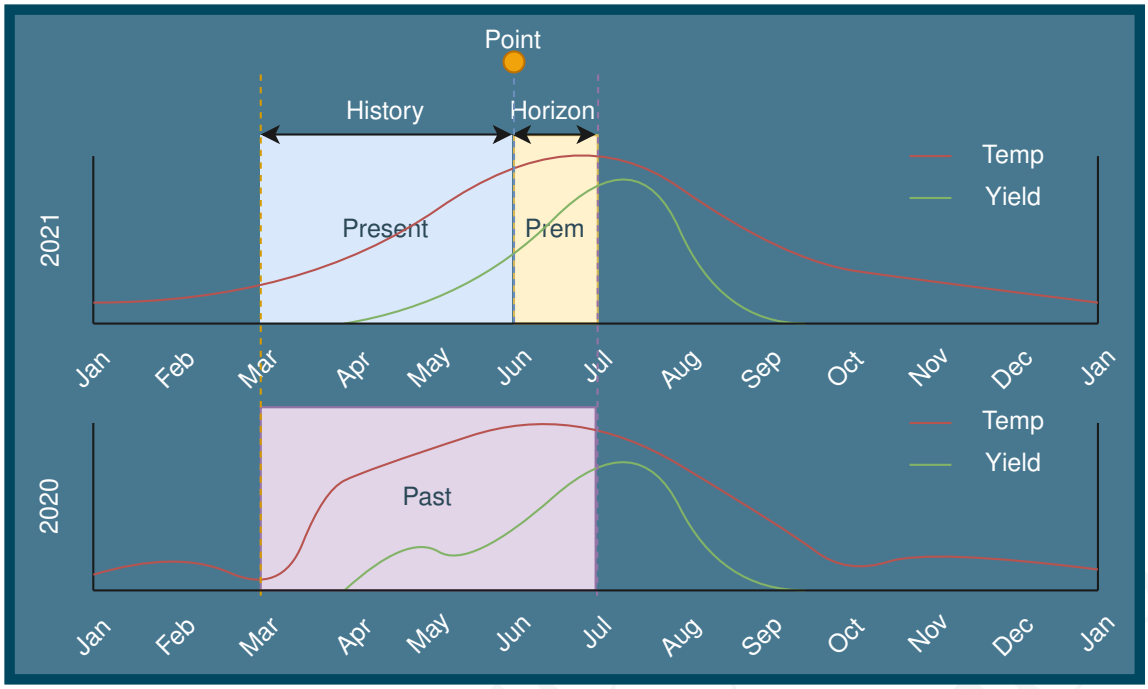

Figure 5.1: Past (purple-pink), present (blue) and premonition (yellow) timelines/ windows overlaid on a depiction / rough reference of strawberry yields through the years of 2020 and 2021 along with temperature. Depicting the point of prediction relative to (at the seam of) horizon and history.

also have fortnightly visits by agronomists to ensure we are growing the strawberries satisfactorily.

We use this data in various neural network architectures in Section 5.4 and evaluate their performance in Section 5.6, since the literature would suggest that deep learning approaches are the most performant even for FP. Of these new architectures, we showcase our Premonition Network which seeks to improve upon current tabular/ sequence prediction approaches using all three forms of context, the past, the present, and the premonition of the future. We use the past to learn the overarching distribution, we use the present to set some scale and granularity, and we use the premonition for variability from the standard distribution.

5.2 Contributions

(i) We propose a new multi-timeline transformer neural network (NN) architecture, towards forecasting over multiple growing seasons with varying contexts for the past, present, and the premonition of the future. Our method allows a

transformer to model the relationship between what we have seen before, what we have seen in the current season, and what we expect to see in the future given our current understanding. This reduces the start-of-season forecasting issues and improves season-wide performance significantly when compared to previously published techniques.

(ii) We provide several solutions and techniques necessary to overcome real world problems in out data. This includes skipping windows, resampling for synchronisation, and detailed training and architectural decision making. Our techniques allow complex transformers such as our multi-timeline transformers to ingest data regardless of inevitably varying picking schedule and quality, between seasons where data may not align. This expands the repertoire of applicable data, to allow for deeper training of more complex networks.

(iii) We apply our methods to a real functioning strawberry tabletop site, which suffers from various issues such as pests, labour shortages. This provides a real world baseline for future comparison, albeit on a smaller site, more intensive site.

5.3 Related Work

There are relatively few works in strawberry yield prediction using deep learning, instead the majority focus on statistical machine learning, and almost none that refer to privacy considerations (Hopf et al., 2022; van der Velde and Nisini, 2019; Bouras et al., 2021; Paudel et al., 2021; Zhu et al., 2022; Bali and Singla, 2022; Jafari et al., 2020; Gastli, Nassar and Karray, 2021; Maskey, Pathak and Dara, 2019). However, several papers have stressed that a lack of data availability ((Pearson et al., 2019; Durrant, Markovic, Matthews, May, Leontidis et al., 2021; Durrant, Markovic, Matthews, May, Enright et al., 2022)), or more specifically a high expense of acquisition which significantly hinders the smooth application of state-fo-the-art neural networks towards the creation of powerful forecasting models (Nassar et al., 2020; Jafari et al., 2020; Gastli, Nassar and Karray, 2021; Y. Chen et al., 2019; Maskey, Pathak and Dara, 2019). Many of the aforementioned papers largely choose

to tackle this lack of data by using satellite imagery although in some cases they use the California strawberry commission data paired with the California strawberry commission irrigation management information system (CIMIS). Unfortunately the data mentioned in these papers is behind multiple walls, and the CIMIS data is currently unavailable from the original source, so while we were able to find an excerpt of the CIMIS data elsewhere we were unable to find the full dataset making it very difficult to compare to.

Many different proposals for methods of predicting / forecasting yield (generically) exist, some using classical machine learning (Paudel et al., 2021) others such as those by Nassar (Nassar et al., 2020) use neural networks in their specific case a mixture of CNN, LSTMs, GRUs and some attention heads. However all emphasise the need for better forecasting systems as demand increases and supply decreases due to global factors such as (but not limited to) COVID-19 and the Russia-Ukraine war. Current yield forecasting methods are highly archaic, often times they can be as simple as forecasting the average of the last few years' yields, or simple linear models based on heat hours (Paudel et al., 2021). One such example is the European Commission's MARS crop forecasting system (MCYFS) which has purportedly seen no improvement in its forecasting performance since 2006 and uses no machine learning (Paudel et al., 2021). Lastly the work by Paudel (Paudel et al., 2021) shows that machine learning can already at the very least match (at the start of the season) or beat existing large-scale traditional crop yield forecasting systems such as the aforementioned MCYFS system.

The MCYFS system from 2006 to 2015 has a median MAE of 0.379, 0.368, 0.570 in soft wheat durum wheat and grain maize (van der Velde and Nisini, 2019). The most performant forecasts for this system appear to be sunflower yields at 0.162 MAE. However the assessment carried out by van der Velde does not state over what period these yield predictions are made specifically whether that be a few weeks, days or months ahead making this also a difficult comparison to make. It is also apparent that forecasting is becoming increasingly difficult with the higher degree of variability in climate conditions as the performance of this largely static forecasting system seems to be in slow decline (van der Velde and Nisini, 2019).

As more modern dynamic techniques are still only just beginning to be used in literature towards strawberry tabletop forecasting we look towards the application of these much more modern techniques, in particular deep learning / neural networks. However, as previously stated data is incredibly difficult to attain in this domain. Nassar (Nassar et al., 2020) appears to show how the compound deep learning models outperform standalone deep learning models and traditional machine learning models. Nevertheless, as with much work in this space, it is difficult to garner any concrete comparable statistics. From one of their diagrams (14) we believe we can see their most performant model to produce an MAE loss of roughly 0.14 or 14% MAPE. They call this model Attention-ConvLSTM2D. While we do not have access to the same data as they have, we have seen even simple GRU models attain similar performance in our strawberry tabletop. However, we believe we can improve this performance on our own data by means of attention as their paper would also suggest, but instead of standalone attention heads we intend to use a much more complex and performant transformer model.

Transformers as proposed by Vaswani et al (Vaswani et al., 2017) are state-of-the-art neural network components for sequence-to-sequence problems. Strawberry yield prediction is such a problem thus we are keen to implement and use them in this scenario, having used other methods to varying degrees of success in the past (Onoufriou, Hanheide and Leontidis, 2020b; Onoufriou, Hanheide and Leontidis, 2021). We also note that in contrast to our previous techniques transformers and their attention heads can help focus the neural network into parts of the data that are most important thus reducing the need for quite as much data compared to equivalently complex neural networks.

In short yield forecasting is essential for improving on food security, and sustainable development (Zhu et al., 2022). Yield estimation is difficult due to a lack of data availability and thus a lack of research using modern data-hungry techniques in this domain (Nassar et al., 2020; Jafari et al., 2020; Gastli, Nassar and Karray, 2021; Y. Chen et al., 2019; Maskey, Pathak and Dara, 2019). Most attempt to solve this data shortfall by using remote sensing, or by using a select few difficult-to-attain datasets like the california commissions data (Zhu et al., 2022; Jafari et al., 2020;

Nassar et al., 2020). Few works have applied modern deep learning / neural networks successfully to agriculture, and especially strawberries, the majority use either old neural network forms or don't use neural networks at all.

5.4 Material and Methods

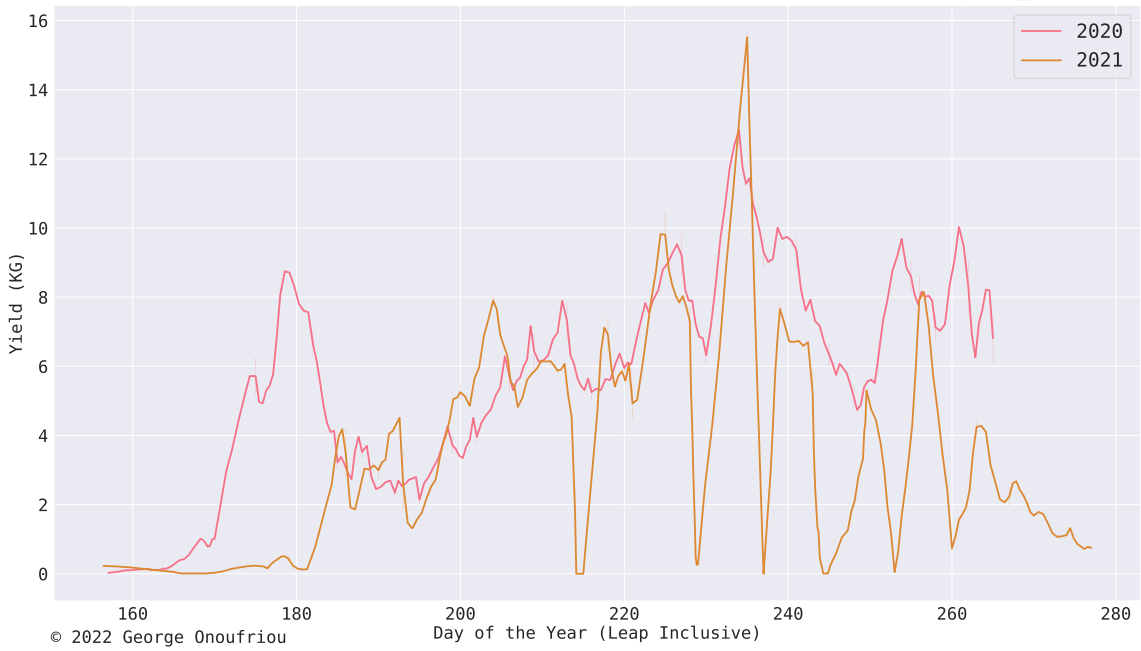

Figure 5.2: Seven day rolling average line-plot of the strawberry yields of both the 2020 and 2021 seasons.

We have collected 3 years of strawberry tabletop data at our Riseholme campus. This data comprises 2 polytunnels, each with 5 rows of strawberry tabletop, each tabletop being 20 meters long. Thus in total, we had 200 meters of strawberry tabletop over any single season. Over these rows we had two different June bearing varieties at any one time from Driscoll's Zara, Katerina, and Malling Centenary. Figure 5.3 shows the two varieties chosen for the 2021 growing season from the aforementioned three, as can be seen, their performance while similar, differ in that Katrina is expected to output more total yields in any given picking session on average. The data capture devices we employed for this strawberry tabletop was:

(i) Irrigation data from the tabletop irrigation system. This includes features describing the nutrients, moisture levels, soil temperature, input irrigation, and irrigation runoff. With a sample rate of 1 sample per 2 minutes.

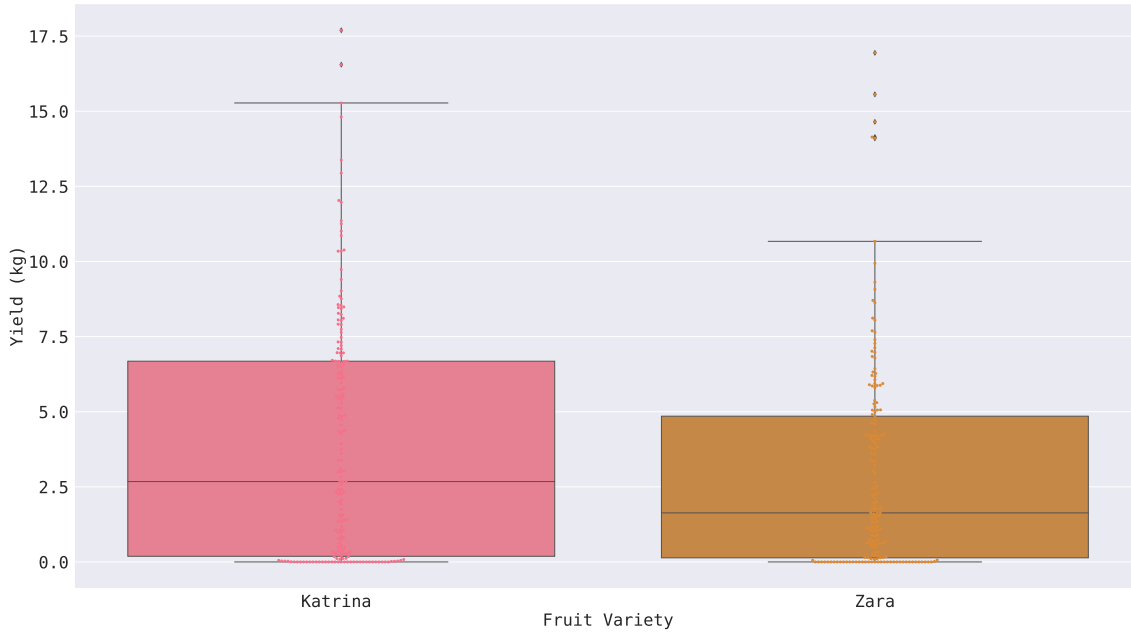

Figure 5.3: Yield performance of the Katerina and Zara strawberry varieties over the 2021 growing season.

(ii) Environmental data from a central weathervane which collected information about: Temperature, humidity, wind direction, wind speed, solar radiance, and precipitation. With a sample rate of 1 sample per 15 minutes.

(iii) Yield weight and quality data from our strawberry picking team. With a sample rate of 2 full picks per row per week.

5.4.1 Data Wrangling

One of the biggest challenges when working with any time-series dataset is to ensure synchronicity. Since all 3 data sources are sampled at different sometimes overlapping intervals it was necessary to re-sample the datasets to achieve synchronisation. We opted to synchronise over the 15 minutes intervals to match the weathervane data. We later downsampled the synchronised data to a much more manageable 4-hour interval when fed into our MTT.

One of the other challenges when working with any data is missing or unrepresentative samples. Unfortunately in real-world scenarios we always expect to capture some missing or inaccurate data, especially when humans are necessarily involved in the process. We chose to use a forward-fill strategy whereby any missing values are

filled with the last known values. The only features not forward-filled are ones that are sampled too infrequently to be able to reasonably forward-fill them. This means any missing values in yields for instance (which are collected bi-weekly) are removed as we cannot reasonably infer them from neighbouring values.

Now that we have a regular dataset with no missing values we can begin example extraction as per Figure 5.1. We create hopping windows that end on/ are aligned to observed yield outcomes in the current/ predicted-for year. The window lengths we chose are 21 days for the premonition, 12 weeks for the present and the cumulative period for both combined in the previous year as the past. This way we have information on adverse weather forecasts, current strawberry performance and performance of strawberries at the same site last year. We then create time sequences using expected date ranges. the historic data and when we have specific outcomes for fruit yields. This meant we roughly formed 2 examples for every week in the growing season. We then further split this data by row into training (2,3,4,6,7,8,10), and testing (1,5,9) sets, while further subdividing the training set into training and validation using k-fold cross validation where $k = B_t$ with a batch size of $B_s = 32$ which resulted in $B_t = 10$ batches. We held out the two final shuffled batches as a per-epoch validation set. We split in this manner to ensure there is no overlap between training and testing sequences, and it enables us to have a full multi-year view since there are not enough years of data with which to hold out.

Finally we normalised our dataset feature-wise using a basic linear transformation Equation 5.1.

$$x_i'^{<t>} = (b - a)\frac{x_i^{<t>} - min(x_i)}{max(x_i) - min(x_i)} + a \tag{5.1}$$

Where the desired normalised feature value for x_i at timestep t post normalisation $x_i'^{<t>}$ is in $[a, b]$. We chose our range to be $[-1, 1]$. We inverted our results to real values using the inversion Equation 5.2.

$$x_i^{<t>} = (x_i'^{<t>} - a)\frac{(max(x_i) - min(x_i))}{b - a} + min(x) \tag{5.2}$$

5.4.2 Architecture

As can be seen in Figure 5.4, our MTT consists of 3 differently parameterised transformers merged together using a dense layer. Thus our architecture is comprised of 3 encoders, 3 decoders and a dense layer.

Encoder and Decoder

As is standard for transformer networks it is necessary to decide upon some form of positional encoding (Vaswani et al., 2017). In our case we use a standard fixed positional encoding where even positions are encoded using Equation 5.3 and odd positions are encoded using Equation 5.4.

$$PE_{pos,2i} = sin(\frac{pos}{10000^{2i/D}}) \tag{5.3}$$

$$PE_{pos,2i+1} = cos(\frac{pos}{10000^{2i/D}}) \tag{5.4}$$

This positional encoding for each odd and even position is then added to the feature vector to allow the neural network some context into the order of inputs. There was no need to form a tokenised input embedding since we already have a distinct feature space described in our feature vector directly from the tabular sequences.

An abbreviated form of the multi head attention depicted in Figure 5.4 (c) is Equation 5.5 along with the weight matrices.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}}) \tag{5.5}$$

$$W_i^Q \in \mathbb{R}^{D \times d_k} \tag{5.6}$$

$$W_i^K \in \mathbb{R}^{D \times d_k} \tag{5.7}$$

$$W_i^V \in \mathbb{R}^{D \times d_v} \tag{5.8}$$

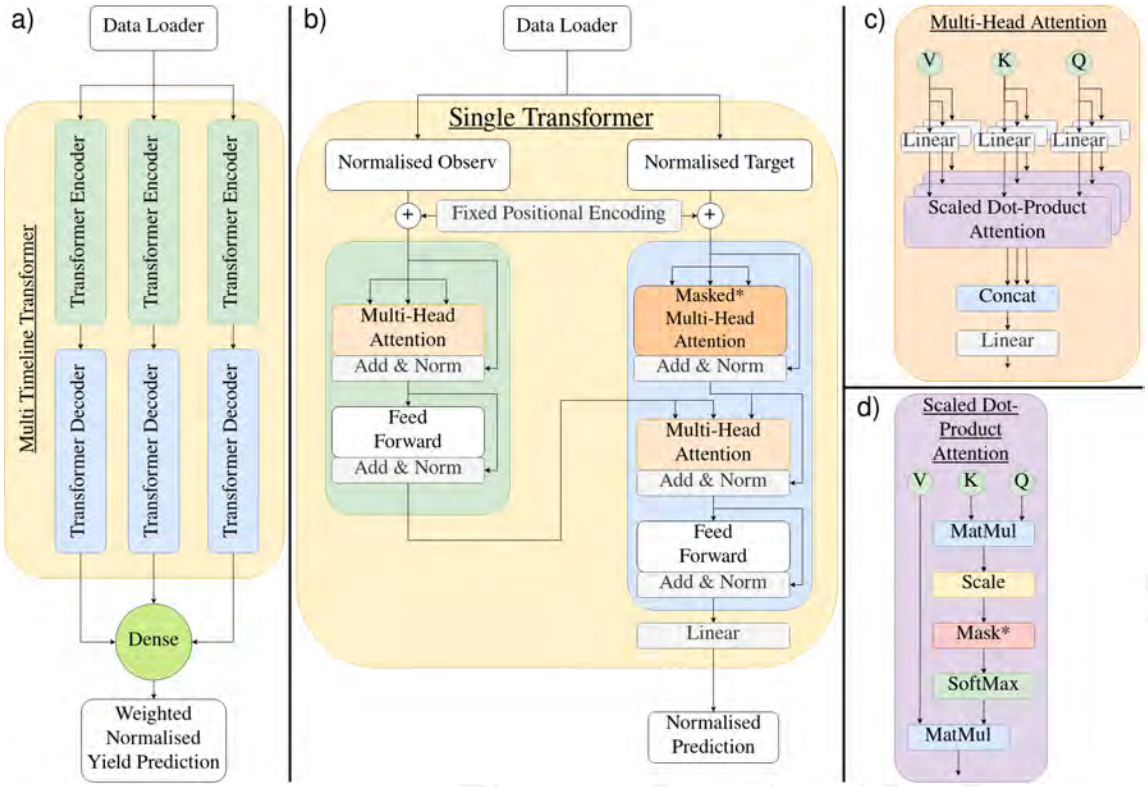

Figure 5.4: a) Mutil Timeline Transformer (MTT) architecture wherebye three single transformers that each process different data streams, are merge by a learned dense layer to weight their significance. b) A full single transformer architecture comprised of fixed positional encoding, encoder, decoder, and linear layers notably missing Softmax. c) Multi-head attention mechanism with query, key, and value matrices. This is a sub-components of transformer encoder and decoders with optional masks to maintain the temporal blindness when processing all the data simultaneously. d) Scaled dot-product attention showing the various matrix operations necessary to compute. this is a sub component of multi-head attention.

Dense

The dense layer is a simple linear layer with enough weights to form the weighted sum of the inputs and concatenate them into a singular value output in Equation 5.9

$$\hat{y} = \sum a_t W_t + \sum a_n W_n + \sum a_f W_f \tag{5.9}$$

Towards gathering data we employed our own data collection pipeline on our Riseholme strawberry tabletop site, the respective yields of this site can be seen in Figure 5.2. All the following data is streamed into MongoDB and accessed using aggregation pipelines to help speed up the transformation process.

Weight Initialisation

For weight initialisation we used the default pytorch Kaiming uniform initialisation as defined in Algorithm 9 for leaky-ReLU ((Nair and G. E. Hinton, 2010; Radford, Metz and Chintala, 2015)).

Algorithm 9 Kaiming uniform weight initialisation using leaky-ReLU with the fan-in method. where a: (default 0 for ReLU, or -0.01 for leaky-ReLU) is the negative slope of the rectifier used after this layer. W: a randomised weight matrix with mean 0 and variance 1 (shape e.g $(64, 32)$) *mode*: is a flag which represents a different value for the *fan* whether the method being used is for feedforward or backpropagation (e.g if $mode =$ fanin then $fan = 64$ else $fan = 32$ given previous example W matrix).

```
function KAIMING_UNIFORM_WEIGHT_INIT(a, W, d)
    if mode = fanin then
        fan = dim(W, 0)
    else
        fan = dim(W, 1)
    std = sqrt(2 / ((1 + a^2) × fan))
    return W ⋆ std
```

Loss Function

We chose to use the Mean Squared Error (MSE) as our loss function where MSE $= \frac{\sum_{i=0}^{N-1}(y-\hat{y})^2}{N}$. This allows us to exponentially penalise large more errors than small errors on our continuous yield forecast. We in particular seek to reduce the networks tolerance for larger single errors as these would mean even if the total error was the same, being particularly peaked in one prediction would result in the growers having to import fruit that particular week. We would much rather be consistently out by a known amount than having almost perfect performance one week and then large errors the next.

As is commonly the case we use adaptive moment estimation (ADAM) (Kingma and Ba, 2014) as our neural network optimiser as it is has been shown to be more performant than just first order or second order moments and is by and large the defacto standard. We calculated our first order moments $m_t = \beta_1 * m_{t-1} + (1-\beta_1) * g_t$ $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ and second order moments.

5.4.3 Models

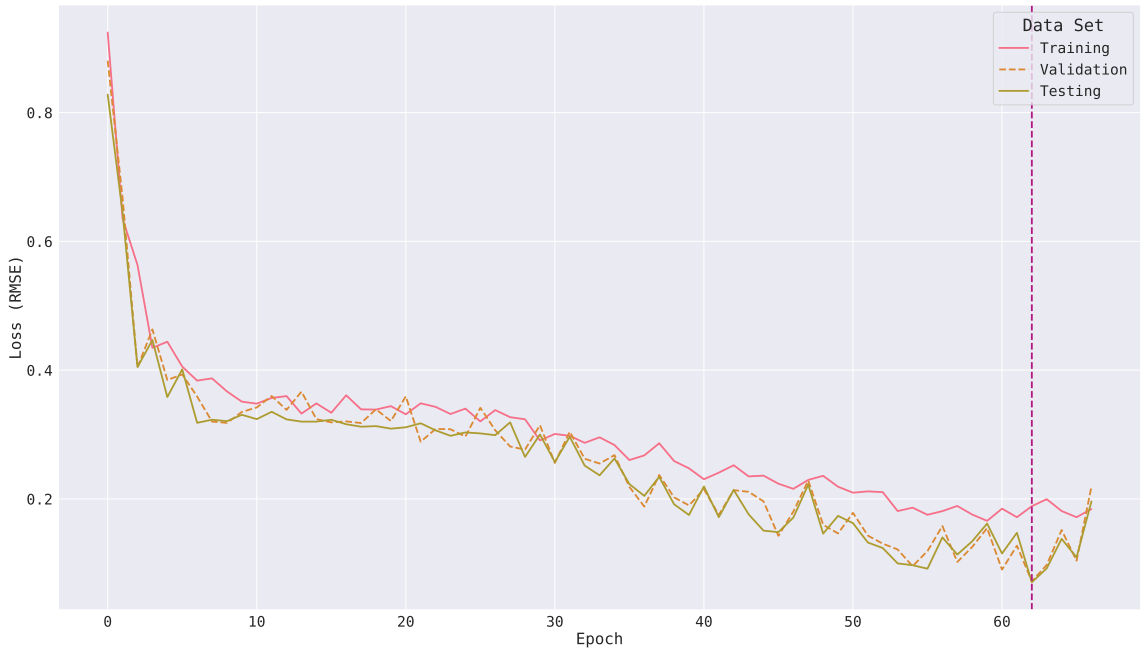

Figure 5.5: Three timeline transformer loss training, validation and testing sets, per epoch of training. Beyond 62 epochs (pink vertical line) validation and testing loss steeply increases again.

We primarily focused on two different types of model. One holistic model that learned from all of the training rows using random subsets for training and validation (Figure 5.5). Then we also attempted to create smaller weaker predictors as an ensemble only trained on a smaller set of the training data to each other as an ensemble to attain simple certainty metrics, which we deem would be invaluable towards building trust in the models and enabling re-investigation of uncertain scenarios. We split the training data used into 3 row sets of tabletop for each ensemble member. Each ensemble member is equivalent to the base MTT, including weight initialisation, loss function, and optimiser. Overall this means there was a one-row overlap between the first-second and second-third MTT. The results of our two current attempted approaches along with our past approaches and expected forecasting performance of growers and agronomists can be seen in Table 5.1.

Forecaster	Expected Error
Grower	25%†
Agronomist	17%†
Recurrent Neural Network (RNN)	21%
Long-Short Term Memory network (LSTM)	38%
Gated Recurrent Network (GRU)	16%
Multi-Timeline Transformer (MTT)	**8%**
Ensemble of MTT (average)	27%
Ensemble of MTT (median)	30%

Table 5.1: Expected errors by forecasting source. All models are from our previous work trialling different methods on the same dataset.
† : These are estimates and may not be representative of any grower or agronomist specifically but are instead ballpark figures for illustration based on our information from our industry partners.

5.5 Results

As can be seen in Table 5.1 our primary MTT that can forecast three weeks ahead within 8% RMSE is a large improvement over current capabilities as forecasts by agronomists tend to not only vary wildly from agronomist to agronomists (14 to 30%), rely on specialist human presence, and are less accurate than our current model. However, a large caveat is that our model was created with intensive/high-quality environmental and yield data, on a small site compared to the typical industrial settings.

The results shown in Table 5.1 and Figure 5.6 are a significant step forward in the prediction of strawberry yields, however, there are some weaknesses to our approach and the yield outcomes. Firstly our ensemble is significantly under-performing especially since a single predictor trained on the whole dataset beats the ensemble significantly. This is likely due to data, with almost three times the parameters, we suspect that we require more training data to learn adequately, yet they receive 1/3 of the total training data each. However, as time progresses and more data becomes available to us over more seasons, we believe this ensemble will outperform the single MTT while enabling ensemble-based certainty estimation. Secondly and most difficult is the data itself. While we are fortunate to have access to our Risehome campus and the strawberry tabletop site, there is still a lack of data available for

Figure 5.6: Ordered forecasts of single MTT compared to ground truth with a horizon of 3 weeks and a history of 12 weeks.

use. This relatively small site means we likely have not learned some of the more complex variances present on larger sites where the sensors' immediate environment might be significantly different to another area on the growing site some distance away meaning the data in such scenarios might be significantly less representative of the conditions experienced by the strawberries.

5.6 Discussion

Our strawberry dataset while covering 200 meters of strawberries is still limited. Commercial sites in comparison have hectares of such crops, meaning our 200 meters is not as representative of larger sites with more intra-crop variability. However, as previously mentioned data availability is scarce making it practically very difficult to collect hectares of data, not least due to actual or perceived data sensitivity by the respective growers. In spite of this, while there may need to be some adjustments to account for more intra-crop variability of these larger sites our neural networks perform well given the data availability. While the sites are smaller and easier to learn, they also have less data to do so, which we believe to be a fair trade-off with no loss in difficulty between their larger sites and our smaller site.

We have a high level of intra-crop variability with our dataset in the similarity between rows. Largely while there is inter-row variance there is still a risk of overfitting since even if the neural network cannot see row 1, for instance, it may be able to relate the yields of row 1 from previously trained/known yields of row 2. We would have ideally liked to have split by time, and claimed one whole season as a completely separate testing set with none of those rows being trained on. However, due to the reality of strawberry seasonality and that there are only so many seasons with which it was possible to collect data, we had to split in such a way as to give the neural networks context for at least two seasons from start to finish. This is only necessary since the current methods of strawberry prediction in industry are largely based on the occurrences of the last season. As such we attempted to base our methods on existing techniques, and intuitively the performance of the strawberries last year will be related to the current season's performance unless some large shift in methods between the seasons occurs.

Figure 5.3 shows a significant number of zero / near-zero values. This is due to the slow start at the beginning of every season as shown in Figure 5.2. In our data collection, we still recorded fruitless strawberry picking sessions to account for some strawberry plant varieties producing for longer periods in the growing season, whereas others started later. This is significant as the total berries one would expect to harvest over the season is affected. In particular, for the 2021 season, we experienced a very slow start to our season with very low yields when compared to the 2020 season. Later in the season, we may also experience zero / near-zero values, these are difficult to distinguish from actual low values and bad picking sessions. One way that we might have made such assumptions is by assuming the harvesting effect causes all temporally adjacent picks of the same row to have diminishing returns.

Our MTT used an interval of 4 hours despite our data being synchronised over 15 minutes intervals. This was a tradeoff between data density (thus model complexity), and data availability. Since we only had a finite number of concrete outcomes that we observed we had to limit the complexity and weights of the model so that it could train its fewer weights with what limited data we had for concrete observations. In contrast, if we had used a data density of 15 minutes intervals we would have had

to have significantly larger weight matrices being backpropagated from the limited number of observed yield values. If, however, we found ourselves with large hectare scale datasets with many more observed outcomes we could tune the model to be more complex to leverage this data, to allow the model to understand much more complex relationships like the aforementioned expected intra-crop variability.

It may also be noted that we use a simple missing data imputation algorithm strategy namely forward-fill which involves filling missing values with the last known value. This was chosen as we mostly only incurred individual or relatively sparsely missing data. In larger sites one might expect to find entire regions that have some data unavailability for some time, meaning more advanced data-filling strategies may be necessary under such conditions. However, in our site, since the missing values were relatively sparse, the forward fill strategy is sufficient to allow us to leverage data in spite of any missing observations or features. The only notable exception is that of yield values. Since yield values were recorded sparsely a single missing value represents a much larger significance. Thus any such missing values are excluded entirely. Thankfully we had very few such missing values.

Due to the data scarcity, we used fixed positional encoding as opposed to learned encoding. This means the gradients would not be shared with the learned positional encoding. This is sufficient since in the original transformer paper (Vaswani et al., 2017) fixed positional encoding and learned positional encoding result in similar performance.

Finally, we chose to use a tri-transformer architecture merged using a dense fully connected layer. We did this to allow the neural network to train separate contextualising units for each potential timeline. This way we can easily conceptualise the timelines as follows. The pasts purpose is to have a broad view of the relationship between the features and the expected outcomes. This is important as we want to ensure the network has context for how yields are expected to outcome given past scenarios. The present serves to contextualise how this current specific season or crop is performing such that it can later be related to what has happened in the past.

The future timeline/transformer is to add mitigations and adverse effects, such that high expected fluctuations can be considered at the merging layer.

5.7 Conclusions

Transformers are more performant, even on small datasets, for forecasting strawberry yields, than many other forms of neural networks (CNNs, RNNs, LSTMs, GRUs etc). Multi-timeline transformers are very capable of learning from the past, the present, and the premonition of the future, even when these use similar approaches to human forecasters who perform far worse. There has been little work in forecasting strawberries using state-of-the-art deep learning methods, and all of the works that do exist struggle with data availability. Data is clearly the principal problem, we need more data, and we need to encourage more data so more impactful research with up-to-date methods can be performed. With more data we can properly test ensemble models and similarly data hungry models which currently its in-availability prohibits due to the poor training we can expect to see.

Towards future works one key area area would involve to implementing certainty metrics that do not require the use of ensembles so that we can keep the neural network parameters down. This would reduce the necessary data to train more complex models. We also seek to make transformers that are abelian compatible such that we can use some of our prior fully homomorphic encryption (FHE)(Gentry and Halevi, 2010) deep learning methods with these currently incompatible but performant transformers (Onoufriou, Mayfield and Leontidis, 2021a; Onoufriou, Hanheide and Leontidis, 2021).

Lastly, we seek to find ways in which to make our data available for wider use, currently that is not possible due to contractual constraints which were necessary to enable us to collect this data with industrial varieties in the first instance. However, we seek to remedy this in future.

Chapter 6

Conclusions

From the outset as outlined in 1.2 our aim was (reaffirmed here for convenience):

To provide automated agronomy support for agronomists at scale using machine/ deep learning techniques for yield prediction, to minimise costs, and maximise specialist human time in areas that require the most attention, from high dimensional spatio-temporal data. Including providing certainty metrics to mitigate, and reaffirm uncertain predictions, with reasonable security to protect both the data owner, and neural networks.

To achieve this aim, we set out milestones that would mark the steppingstones to achieving this goal and how we achieved these goals:

(i) **Create an autonomous data collection system**; We conceived, setup, maintained, and exploited several different data systems that autonomously collected different data. We created in-part a ROS based system in conjunction with SAGA robotics and LCAS as part of a larger Rasberry project. This traversed the strawberry tabletop to collect various images of the strawberry tabletop to inform us of their condition over time. We also created stationary data collection and camera imaging systems to collect information about the environment inside and outside the polytunnel, and also utilised existing systems like the irrigation system to automatically collate data thought the year.

(ii) **Create a data aggregation, and utilisation pipeline**; We created a distributed MongoDB based database layer, and aggregation pipelines to automat-

ically sync, backup, and serve the data from across sites but in particular our Riseholme strawberry tabletop. We also created, maintained, and exploited pytorch dataloaders, datasets, and neural networks (NNs) to automatically stream, batch, clean, filter, this MongoDB data and utilise it directly into our various models.

(iii) **Deploy an agronomy assistive machine learning (ML) model to predict plant yield ahead of time**; We created various models based on CNNs, RNNs, GRUs, LSTMs, and transformers to forecast yield accurately. We found that multi-timeline transformers significantly outperforms most other methods and is a strong candidate for use in industry due to its ability to accurately forecast based on environmental conditions alone which reduces the need for complex robotic systems in practice.

(iv) **Assess viability of privacy-preserving machine learning**; We chose the most under-developed and most in-need form of privacy-preserving machine learning (PPML). fully homomorphic encryption (FHE). We conceived new forms of encrypted deep learning (EDL) and used it to forecast yields completely privately with negligible loss in performance. We were very surprised by the abilities and promise of FHE, it offers a very unique solution to the problem of privacy, and to the future of deep learning (DL).

Along the way we concluded certain key findings:

Agricultural **yield forecasting is extremely difficult**, yet it is very important to forecast accurately to improve upon food security, and reduce waste in the fresh produce (FP) supply chain. Industry current practices are simply insufficient when compared to what we can achieve with DL and NNs when data is available. Thus we must encourage stakeholders to collect more, better quality data, so that NNs can provide them with highly accurate, timely, and scaleable forecasts. We have garnered excellent results on our Riseholme tabletop using transformer-based NNs forecasting 3 weeks ahead, which should allow enough time for negotiations and planning to reduce any waste or expensive "make-up" importing to cover any shortfalls.

Data is scarce, and difficult to authenticate. We have found that this makes it

incredibly difficult to build any model of any note with what data is available, without exhausting countless amounts of labour to create even the simplest models, let alone complex state-of-the-art models. This is in large part due to its initial in-availability but also as a consequence of the link rot phenomenon, affecting what few datasets are available. We solve this issue for ourselves by collecting our own data, however we hope to encourage more data sharing through PPML and showing stakeholders the benefits of DL. This should improve on the awareness and willingness of stakeholders which will hopefully result in greater general availability of such datasets.

DL is ripe for abuse, and we have seen that stakeholders are keenly concerned with such abuses of their data leading to exploitation. We have shown how PPML can help solve these issues by combining DL and FHE into EDL and encrypted deep learning as a service (EDLaaS) both of which are concepts we have coined, and exploit actively. We have found that FHE helps produce private inference given compatible abelian-based NNs. We have found, and created new ways for more NNs to be abelian-based (convolutional neural networks (CNNs), rectified linear unit (ReLU) activation functions) and for these abelian-based networks to be traversed.

6.1 Limitations

The gains we can garner from FHE currently are limited. The primary limitation is computational depth. Not all implementations of FHE are equal, some implementations are complete with bootstrapping. Bootstrapping allows us to refresh the keys without decryption of the cyphertext before the noise of the cyphertext outgrows our ability to correct the errors, which would otherwise turn the decrypted result into a garbled message of no worth. While much of our work over the course of the PhD has been based on Microsoft simple encrypted arithmetic library (MS-SEAL), it does not currently support bootstrapping. However we are well underway swapping to and using Lattigo which is a Go based library that does support boostrapping. With bootstrapping in future we will be able to go to much deeper depths with neural networks, which is necessary if we are to use EDL over more advanced NNs.

Further limitations of FHE are that it is not particularly helpful during model train-

ing. This is because it is impossible to know when to stop training, as the outputs should, given everything is encrypted, be impossible for us to peer into to ascertain if the outputs are improving or have reached the optimal level of representation.

We sought to use various external datasets but we consistently found that there was no readily available data that we could confidently use to provide any reasonable outputs. We had access in our MongoDB database, data from other Berry Gardens Growers (BGG) farms but this data was pure yield outputs and no associated locations, and now environmental data to pair. We also found partial data for California strawberries, but this did not have enough environmental data to pair nor was the yield data in a format that was reasonably processable being encoded as PDFs. This means we cannot verify how representative our models are for large industrial sites since our Riseholme dataset which was supposed to be our fallback dataset is still quite small. This is in part also due to the Coronavirus disease 2019 (COVID-19) pandemic making it much harder to collaborate with our industry partners.

6.2 Implications Insights and Future Perspectives

We would like to split our insights and perspectives into the constituent topics to make it simpler and cleaner to elaborate:

6.2.1 Yield Forecasting

We believe schedules, and weather forecasts are often overlooked and are in-fact key components in yield forecasting. The picking schedules play a key role in yield availability and thus can cause monumental shifts of yields. For future works we believe rote schedules, with picking guarantees are critical, any fruit that remains will only serve to over-ripen and cause disease as well as reduce actual recorded picking values. Remaining fruits will also continue to drain the shared resources uptaken by the root system which bottlenecks the plant in most optimal instances. In our data you can see some of the consequences of this in Figure 5.2 where we have dips to near 0 then peaks. Given there are 5-6 picks before a 3 week forecast horizon, if these picking schedules are not taken into account they could account for

a large portion of the deviance in actual observed yield outputs, even if the model forecast all other factors perfectly.

We believe our work has broader implications for yield forecasting, which is currently very course and rudimentary. Not enough attention is paid to how data is processed and how multiple seasons can be leveraged to inform our understanding of the future. This in particular helps us get consistent forecasts even towards the beggining of a given season. We know now that DL (transformers) paired with good data collection practices can achieve excellent results even in low-data scenarios. We know that with this improvement to forecasting performance we can reduce waste, and improve prices, with relatively little data. The problem remains that data must be collected, we hope that in showing such good performance we are able to convince stakeholders to share more data.

6.2.2 Deep Learning

DL is an ever expanding and bright field, with a plethora of large advances in very short timeframes. In many ways it is advancing faster than a single researcher can hope to keep up broadly speaking unless they sub-specialise. However for DL to continue to flourish it must also become more harmonious with society, as it is advancing faster than the associated social change / acceptance and certainly faster than legislative regulation. This means it is up to us as experts and researchers to responsibly advance the field to make it sustainable. The key way we propose is by at a bare-minimum respecting privacy, DL applications can already be a fairly worrying topic for the public, they aren't even cognoscente of the true depth of the field. ChatGPT (general-purpose transformer (GPT)-3), deep fakes, driverless cars, facial recognition, automation of all kinds, we have yet to have a flash point that focuses the public's attention. All of these applications have very real consequences to the fabric of society, and have very difficult moral quandaries of which we are wholly unfamiliar, and under-prepared to deal with.

Our work here shows yet again how performant and applicable transformers are. DL is riddled with transformers, that are state-of-the-art in a concerning number

of fields. Transformers were conceptualised in December of 2017, it has been just a little over 5 years (at the time of writing), and yet they are markidly overrepresented in the state-of-the-art. As eluded to much earlier (sec 2.4) they are state of the art in time series forecasting (TSF) (Zeng et al., 2022; Zhu et al., 2022; Minhao et al., n.d.; H. Zhou et al., 2021), machine translation and language modeling (Takase and Kiyono, 2021; Ghorbani et al., 2021; Shoeybi et al., 2019), semantic segmentation (Zhe Chen et al., 2022; W. Wang et al., 2022) to name a few. This is all to say, DL advances rapidly, it takes a remarkably short amount of time and papers to see leaps and bounds in improvements. We must be keenly aware of the pace of DL, and this work here shows how transformers for agriculture are yet another performant application.

6.2.3 Fully Homomorphic Encryption

FHE is a very promising field, that can help mitigate many problems, including some that are yet to come to pass. Namely quantum computing and thus decryption, as well as privacy enhancing technologies for DL. FHE promises us the ability to compute completely privately, in a quantum resistant manner. However it also comes at significant cost. FHE is very time, computation, and space intensive, which means it generates more carbon, and costs more to operate. It is thus certainly not a catch-all panacea to all problems. However for already extremely taxing and expensive operations like DL and the sheer risk DL could imply to sensitive domains, could warrant this order of magnitude increase in resource consumption. We do believe that there is still a significant gap that other PPML techniques are required for, and there is still much advancement of FHE schemes themselves to make them more palatable, simple, and efficient. Under this assumption of subsequent improvements we believe this gap will narrow, broadening the appropriate applications for its use based on the specific cost-benefits.

We have shown how FHE can be applied to agriculture using simpler NNs. We show this can achieve respectable results, but the next challenge is to implement state-of-the-art models like transformers which are as yet unimplemented due to the recentness of softmax approximations which were previously incompatible due to

attention and softmax being non-commutative (J.-W. Lee et al., 2022). However given the rapid pace of advancement it will be difficult to keep up with DL without significantly more attention and effort, which will also necessarily require lowering the barrier to entry, which we believe our work here does. We make it simpler, more digestible, and more packageable for more future research to pick up where we leave.

6.3 Funding

This research was supported in part by the Biotechnology and Biological Sciences Research Council (BBSRC) studentship [grant numbers 2155898, BB/S507453/1].

6.4 Future Beyond PhD

This work has inspired the incorporation of Deep Cypher Ltd. (12989167), an open-source Kerckhoffian Fully Homomorphically Encrypted deep learning as a service company. This will allow me to further develop and invest in the ideas of this PhD, and move closer to the goals of private machine learning for all.

References

Al Badawi, Ahmad et al. (2022). 'OpenFHE: Open-Source Fully Homomorphic Encryption Library'. In: *Cryptology ePrint Archive* (cit. on p. 43).

Alvarez, R (2009). 'Predicting average regional yield and production of wheat in the Argentine Pampas by an artificial neural network approach'. In: *European Journal of Agronomy* 30.2, pp. 70–77 (cit. on p. 19).

Anderson, Rachel et al. (2019). 'Evaluating deep learning techniques for dynamic contrast-enhanced MRI in the diagnosis of breast cancer'. In: *Medical Imaging 2019: Computer-Aided Diagnosis*. Vol. 10950. International Society for Optics and Photonics, p. 1095006 (cit. on p. 19).

Baghdasaryan, Liana et al. (2022). 'Deep density estimation based on multi-spectral remote sensing data for in-field crop yield forecasting'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2014–2023 (cit. on p. 18).

Bali, Nishu and Anshu Singla (2022). 'Emerging trends in machine learning to predict crop yield and study its influential factors: a survey'. In: *Archives of computational methods in engineering* 29.1, pp. 95–112 (cit. on p. 76).

Biswas, M et al. (2019). 'State-of-the-art review on deep learning in medical imaging.' In: *Frontiers in bioscience (Landmark edition)* 24, pp. 392–426 (cit. on p. 19).

Bos, Joppe W. et al. (2013). *Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme*. Cryptology ePrint Archive, Report 2013/075. `https://eprint.iacr.org/2013/075` (cit. on p. 42).

Bouras, El Houssaine et al. (2021). 'Cereal yield forecasting with satellite drought-based indices, weather data and regional climate indices using machine learning in Morocco'. In: *Remote Sensing* 13.16, p. 3101 (cit. on p. 76).

Brakerski, Zvika, Craig Gentry and Vinod Vaikuntanathan (2011). *Fully Homomorphic Encryption without Bootstrapping*. Cryptology ePrint Archive, Report 2011/277. `https://eprint.iacr.org/2011/277` (cit. on pp. 42, 44).

Brakerski, Zvika and Vinod Vaikuntanathan (2011). *Efficient Fully Homomorphic Encryption from (Standard) LWE*. Cryptology ePrint Archive, Report 2011/344. `https://eprint.iacr.org/2011/344` (cit. on p. 42).

Brown, Tom et al. (2020). 'Language models are few-shot learners'. In: *Advances in neural information processing systems* 33, pp. 1877–1901 (cit. on p. 14).

Chen, Qiang et al. (2022). 'Group detr v2: Strong object detector with encoder-decoder pretraining'. In: *arXiv preprint arXiv:2211.03594* (cit. on p. 14).

Chen, Yang et al. (2019). 'Strawberry yield prediction based on a deep neural network using high-resolution aerial orthoimages'. In: *Remote Sensing* 11.13, p. 1584 (cit. on pp. 76, 78).

Chen, Zhe et al. (2022). 'Vision Transformer Adapter for Dense Predictions'. In: *arXiv preprint arXiv:2205.08534* (cit. on p. 96).

Chen, Zhigang (2021). *SEAL-Python Bindings Source Repository*. Wanli university. URL: `https://github.com/Huelse/SEAL-Python` (visited on 10th May 2021) (cit. on p. 48).

Cheon, Jung Hee, Kyoohyung Han et al. (2018). 'Bootstrapping for approximate homomorphic encryption'. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, pp. 360–384 (cit. on p. 43).

Cheon, Jung Hee, Andrey Kim et al. (2017a). 'Homomorphic encryption for arithmetic of approximate numbers'. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, pp. 409–437 (cit. on pp. 42, 45).

– (2017b). 'Homomorphic encryption for arithmetic of approximate numbers'. In: *International conference on the theory and application of cryptology and information security*. Springer, pp. 409–437 (cit. on p. 43).

Chlingaryan, Anna, Salah Sukkarieh and Brett Whelan (2018). 'Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review'. In: *Computers and electronics in agriculture* 151, pp. 61–69 (cit. on pp. 19, 23).

Cobbe, Karl W et al. (2021). 'Phasic policy gradient'. In: *International Conference on Machine Learning*. PMLR, pp. 2020–2027 (cit. on p. 14).

Dathathri, Roshan et al. (June 2020). 'EVA: an encrypted vector arithmetic language and compiler for efficient homomorphic computation'. In: *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*. DOI: `10.1145/3385412.3386023`. URL: `http://dx.doi.org/10.1145/3385412.3386023` (cit. on pp. 16, 41, 45, 46, 50, 55).

Deng, Jia et al. (2009). 'Imagenet: A large-scale hierarchical image database'. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255 (cit. on p. 44).

Do, Hai Ha et al. (2019). 'Deep learning for aspect-based sentiment analysis: a comparative review'. In: *Expert Systems with Applications* 118, pp. 272–299 (cit. on p. 19).

Durrant, Aiden, Milan Markovic, David Matthews, David May, Jessica Enright et al. (2022). 'The role of cross-silo federated learning in facilitating data sharing in the agri-food sector'. In: *Computers and Electronics in Agriculture* 193, p. 106648 (cit. on pp. 40, 49, 76).

Durrant, Aiden, Milan Markovic, David Matthews, David May, Georgios Leontidis et al. (2021). 'How might technology rise to the challenge of data sharing in agri-food?' In: *Global Food Security* 28, p. 100493 (cit. on pp. 49, 76).

Environment Food, Department for and Rural Affairs (Dec. 2021). *United Kingdom Food Security Report 2021: Theme 2: UK Food Supply Sources*. URL: `https://www.gov.uk/government/statistics/united-kingdom-food-security-report-2021/united-kingdom-food-security-report-2021-theme-2-uk-food-supply-sources` (visited on 11th Oct. 2022) (cit. on p. 39).

Ershov, Mikhail (2015). *Survey of Algebra*. URL: `http://people.virginia.edu/~mve2x/3354_Spring2015/` (visited on 10th Nov. 2019) (cit. on pp. 11, 12).

Falcetta, Alessandro and Manuel Roveri (2022). 'Privacy-preserving deep learning with homomorphic encryption: An introduction'. In: *IEEE Computational Intelligence Magazine* 17.3, pp. 14–25 (cit. on pp. 7, 41, 45, 71).

Fan, Junfeng and Frederik Vercauteren (2012). *Somewhat Practical Fully Homomorphic Encryption*. Cryptology ePrint Archive, Report 2012/144. `https://eprint.iacr.org/2012/144` (cit. on p. 42).

Fawaz, Hassan Ismail et al. (2019). 'Deep learning for time series classification: a review'. In: *Data Mining and Knowledge Discovery* 33.4, pp. 917–963 (cit. on p. 19).

Gastli, Mohamed Sadok, Lobna Nassar and Fakhri Karray (2021). 'Deep Learning Models for Strawberry Yield and Price Forecasting Using Satellite Images'. In: *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 1790–1796 (cit. on pp. 76, 78).

Gentry, Craig (2009). 'Fully homomorphic encryption using ideal lattices'. In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169–178 (cit. on p. 42).

Gentry, Craig and Shai Halevi (2010). *Implementing Gentry's Fully-Homomorphic Encryption Scheme*. Cryptology ePrint Archive, Report 2010/520. `https://eprint.iacr.org/2010/520` (cit. on pp. 42, 90).

Gentry, Craig, Amit Sahai and Brent Waters (2013). *Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based*. Cryptology ePrint Archive, Report 2013/340. `https://eprint.iacr.org/2013/340` (cit. on p. 42).

Ghorbani, Behrooz et al. (2021). 'Scaling laws for neural machine translation'. In: *arXiv preprint arXiv:2109.07740* (cit. on p. 96).

Gilad-Bachrach, Ran et al. (2016). 'Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy'. In: *International Conference on Machine Learning*, pp. 201–210 (cit. on p. 42).

Google (2021). *Google trends topics: Privacy, Edward Snowden, Cambridge Analytica*. Online: `https://trends.google.co.uk/trends/explore`. URL: `https://trends.google.co.uk/trends/explore?date=2010-01-01%202021-07-19&geo=GB&q=%2Fm%2F06804,%2Fm%2F0vzv0rq,%2Fg%2F11clljsm74` (visited on 19th July 2021) (cit. on p. 39).

Güera, David and Edward J Delp (2018). 'Deepfake video detection using recurrent neural networks'. In: *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, pp. 1–6 (cit. on p. 19).

Halevi, Shai and Victor Shoup (2020). 'Design and implementation of HElib: a homomorphic encryption library'. In: *Cryptology ePrint Archive* (cit. on pp. 43, 44).

Hopf, Alwin et al. (2022). 'Development and improvement of the CROPGRO-Strawberry model'. In: *Scientia Horticulturae* 291, p. 110538 (cit. on p. 76).

Huval, Brody et al. (2015). 'An empirical evaluation of deep learning on highway driving'. In: *arXiv preprint arXiv:1504.01716* (cit. on p. 19).

Jafari, Fatemeh et al. (2020). 'Yield forecast of California strawberry: Time-series Models vs. ML Tools'. In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 3594–3598 (cit. on pp. 76, 78).

Juvekar, Chiraag, Vinod Vaikuntanathan and Anantha Chandrakasan (2018). 'GAZELLE: A low latency framework for secure neural network inference'. In: *27th USENIX Security Symposium (USENIX Security 18)*, pp. 1651–1669 (cit. on pp. 44, 59).

Kingma, Diederik P and Jimmy Ba (2014). 'Adam: A method for stochastic optimization'. In: *arXiv preprint arXiv:1412.6980* (cit. on p. 84).

Kollias, Stefanos et al. (2022). 'AI-enabled Safe and Efficient Food Supply Chain'. In: (cit. on p. 40).

Krizhevsky, Alex, Geoffrey Hinton et al. (2009). 'Learning multiple layers of features from tiny images'. In: (cit. on p. 44).

Lee, Joon-Woo et al. (2021). 'Privacy-Preserving Machine Learning with Fully Homomorphic Encryption for Deep Neural Network'. In: *arXiv preprint arXiv:2106.07229* (cit. on pp. 44, 55, 58, 63).

– (2022). 'Privacy-preserving machine learning with fully homomorphic encryption for deep neural network'. In: *IEEE Access* 10, pp. 30039–30054 (cit. on pp. 15, 16, 97).

Liakos, Konstantinos G et al. (2018). 'Machine learning in agriculture: A review'. In: *Sensors* 18.8, p. 2674 (cit. on p. 23).

Lopez-Alt, Adriana, Eran Tromer and Vinod Vaikuntanathan (2013). *On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption*. Cryptology ePrint Archive, Report 2013/094. `https://eprint.iacr.org/2013/094` (cit. on p. 42).

Marcano, Néstor J Hernández et al. (2019). 'On fully homomorphic encryption for privacy-preserving deep learning'. In: *2019 IEEE Globecom Workshops (GC Wkshps)*. IEEE, pp. 1–6 (cit. on p. 44).

Maskey, Mahesh L, Tapan B Pathak and Surendra K Dara (2019). 'Weather based strawberry yield forecasts at field scale using statistical and machine learning models'. In: *Atmosphere* 10.7, p. 378 (cit. on pp. 76, 78).

Meftah, Souhail et al. (2021). 'DOReN: Towards Efficient Deep Convolutional Neural Networks with Fully Homomorphic Encryption'. In: *IEEE Transactions on Information Forensics and Security* (cit. on pp. 44, 55, 59, 61).

Minhao, LIU et al. (n.d.). 'SCINet: Time Series Modeling and Forecasting with Sample Convolution and Interaction'. In: *Advances in Neural Information Processing Systems* (cit. on pp. 14, 18, 96).

Mouchet, Christian et al. (2020). 'Multiparty homomorphic encryption from ring-learning-with-errors'. In: *Cryptology ePrint Archive* (cit. on p. 43).

Nair, Vinod and Geoffrey E Hinton (2010). 'Rectified linear units improve restricted boltzmann machines'. In: *Icml* (cit. on p. 84).

Nassar, Lobna et al. (2020). 'Prediction of Strawberry Yield and Farm Price Utilizing Deep Learning'. In: *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. DOI: `10.1109/IJCNN48605.2020.9206998` (cit. on pp. 18, 76–79).

Niedbała, Gniewko (2019). 'Application of Artificial Neural Networks for Multi-Criteria Yield Prediction of Winter Rapeseed'. In: *Sustainability* 11.2, p. 533 (cit. on p. 24).

Onoufriou, George (2019). *Nemesyst; Generalised and highly customisable, hybrid-parallelism, database based, deep learning framework*. URL: `https://github.com/DreamingRaven/nemesyst` (visited on 10th Nov. 2019) (cit. on p. 20).

– (2021). *Python-FHEz Source Repository*. Online: `http://gitlab.com/deepcypher/python-fhez`. URL: `https://gitlab.com/DeepCypher/Python-FHEz` (visited on 10th May 2021) (cit. on pp. 44, 48, 55–58, 60, 61, 64, 65).

Onoufriou, George, Ronald Bickerton et al. (2019). 'Nemesyst: A hybrid parallelism deep learning-based framework applied for internet of things enabled food retailing

refrigeration systems'. In: *Computers in Industry* 113, p. 103133 (cit. on pp. 5, 6, 31, 40).

Onoufriou, George, Marc Hanheide and Georgios Leontidis (2020a). 'The Augmented Agronomist Pipeline and Time Series Forecasting'. In: (cit. on pp. 5, 6, 19).

– (Apr. 2020b). 'The Augmented Agronomist Pipeline and Time Series Forecasting'. In: DOI: 10.31256/Qm1Fu7L (cit. on p. 78).

– (2021). *EDLaaS; Fully Homomorphic Encryption Over Neural Network Graphs*. DOI: 10.48550/ARXIV.2110.13638. URL: https://arxiv.org/abs/2110.13638 (cit. on pp. 78, 90).

– (2022a). 'EDLaaS: Fully Homomorphic Encryption over Neural Network Graphs for Vision and Private Strawberry Yield Forecasting'. In: *Sensors* 22.21, p. 8124 (cit. on pp. 5, 7, 38).

– (2022b). 'Premonition Net, A Multi-Timeline Transformer Network Architecture Towards Strawberry Tabletop Yield Forecasting'. In: *arXiv preprint arXiv:2211.08177* (cit. on pp. 6, 8, 73).

Onoufriou, George, Paul Mayfield and Georgios Leontidis (2021a). 'Fully Homomorphically Encrypted Deep Learning as a Service'. In: *Machine Learning and Knowledge Extraction* 3.4, pp. 819–834 (cit. on pp. 40, 90).

– (2021b). 'Fully homomorphically encrypted deep learning as a service'. In: *Machine Learning and Knowledge Extraction* 3.4, pp. 819–834 (cit. on pp. 5, 7).

Parliament, United Kingdom (2018). *Data Protection Act 2018*. URL: https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted (visited on 4th Jan. 2021) (cit. on p. 38).

Paudel, Dilli et al. (2021). 'Machine learning for large-scale crop yield forecasting'. In: *Agricultural Systems* 187, p. 103016 (cit. on pp. 18, 76, 77).

Pearson, Simon et al. (2019). 'Are Distributed Ledger Technologies the panacea for food traceability?' In: *Global Food Security* 20, pp. 145–149 (cit. on pp. 48, 76).

Prasad, Anup K et al. (2006). 'Crop yield estimation model for Iowa using remote sensing and surface parameters'. In: *International Journal of Applied Earth Observation and Geoinformation* 8.1, pp. 26–33 (cit. on pp. 19, 23).

Radford, Alec, Luke Metz and Soumith Chintala (2015). 'Unsupervised representation learning with deep convolutional generative adversarial networks'. In: *arXiv preprint arXiv:1511.06434* (cit. on p. 84).

Raffel, Colin et al. (2020). 'Exploring the limits of transfer learning with a unified text-to-text transformer.' In: *J. Mach. Learn. Res.* 21.140, pp. 1–67 (cit. on p. 14).

Rahnemoonfar, Maryam and Clay Sheppard (2017). 'Real-time yield estimation based on deep learning'. In: *Autonomous Air and Ground Sensing Systems for*

Agricultural Optimization and Phenotyping II. Vol. 10218. International Society for Optics and Photonics, p. 1021809 (cit. on p. 23).

Sartore, Luca et al. (2022). 'Assessing machine leaning algorithms on crop yield forecasts using functional covariates derived from remotely sensed data'. In: *Computers and Electronics in Agriculture* 194, p. 106704 (cit. on p. 18).

Microsoft SEAL (release 3.4.5) (Nov. 2020). `https://github.com/Microsoft/SEAL`. Microsoft Research, Redmond, WA. (cit. on pp. 40, 43).

Shafiq, Iram et al. (2021). 'Crop photosynthetic response to light quality and light intensity'. In: *Journal of Integrative Agriculture* 20.1, pp. 4–23 (cit. on p. 24).

Shoeybi, Mohammad et al. (2019). 'Megatron-lm: Training multi-billion parameter language models using model parallelism'. In: *arXiv preprint arXiv:1909.08053* (cit. on p. 96).

Shokri, Reza and Vitaly Shmatikov (2015). 'Privacy-preserving deep learning'. In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1310–1321 (cit. on p. 14).

Smart, Nigel P and Frederik Vercauteren (2010). 'Fully homomorphic encryption with relatively small key and ciphertext sizes'. In: *International Workshop on Public Key Cryptography*. Springer, pp. 420–443 (cit. on p. 42).

Snowden, Edward (2019). *Permanent Record*. Metropolitan Books. ISBN: 9781250237231 (cit. on p. 38).

Takase, Sho and Shun Kiyono (2021). 'Lessons on parameter sharing across layers in transformers'. In: *arXiv preprint arXiv:2104.06022* (cit. on pp. 14, 96).

Thota, Mamatha and Georgios Leontidis (2021). 'Contrastive domain adaptation'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2209–2218 (cit. on p. 40).

van der Velde, M. and L. Nisini (2019). 'Performance of the MARS-crop yield forecasting system for the European Union: Assessing accuracy, in-season, and year-to-year improvements from 1993 to 2015'. In: *Agricultural Systems* 168, pp. 203–212. ISSN: 0308-521X. DOI: `https://doi.org/10.1016/j.agsy.2018.06.009`. URL: `https://www.sciencedirect.com/science/article/pii/S0308521X18300179` (cit. on pp. 76, 77).

Vaswani, Ashish et al. (2017). 'Attention is all you need'. In: *Advances in neural information processing systems* 30 (cit. on pp. 78, 82, 89).

Wang, Anna X et al. (2018). 'Deep transfer learning for crop yield prediction with remote sensing data'. In: *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*. ACM, p. 50 (cit. on p. 23).

Wang, Wenhui et al. (2022). 'Image as a foreign language: Beit pretraining for all vision and vision-language tasks'. In: *arXiv preprint arXiv:2208.10442* (cit. on p. 96).

Xiao, Han, Kashif Rasul and Roland Vollgraf (2017). 'Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms'. In: *CoRR* abs/1708.07747. arXiv: `1708.07747`. URL: `http://arxiv.org/abs/1708.07747` (cit. on p. 56).

You, Jiaxuan et al. (2017). 'Deep gaussian process for crop yield prediction based on remote sensing data'. In: *Thirty-First AAAI Conference on Artificial Intelligence* (cit. on p. 23).

Yu, Jiahui et al. (2022). 'Coca: Contrastive captioners are image-text foundation models'. In: *arXiv preprint arXiv:2205.01917* (cit. on p. 14).

Zeng, Ailing et al. (2022). 'Are Transformers Effective for Time Series Forecasting?' In: *arXiv preprint arXiv:2205.13504* (cit. on pp. 14, 18, 96).

Zhou, Haoyi et al. (2021). 'Informer: Beyond efficient transformer for long sequence time-series forecasting'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 12, pp. 11106–11115 (cit. on pp. 14, 18, 96).

Zhu, Yilin et al. (2022). 'A deep learning crop model for adaptive yield estimation in large areas'. In: *International Journal of Applied Earth Observation and Geoinformation* 110, p. 102828 (cit. on pp. 14, 18, 76, 78, 96).