Reliable Machine Learning for Omics Data: Evaluation Protocols, Hybrid Models, and Applications in Foodomics

PhD Thesis

Dario Ruggeri
Supervisor: László Vidács, PhD

Doctoral School of Computer Science
Department of Software Engineering
Faculty of Science and Informatics
University of Szeged

Szeged
2026

Contents

List of Figures

List of Tables

Abbreviations

CBM Constraint-Based Modeling 39, 41

CNN Convolutional Neural Network 20, 22, 26, 28, 32, 33, 62, 87, 88

CV Cross-Validation 1–5, 11, 13–16, 18–20, 22–38, 47, 49, 50, 56, 57, 66–71, 74, 76, 78–81, 88, 91–97, 99–101, 125, 126

DL Deep Learning 9–13, 16, 20, 25, 56, 59–65, 68, 73, 82–85, 87, 99, 101, 103, 125–127

ES Early Stopping 1, 2, 4, 5, 12–20, 22, 25–38, 56, 57, 66–68, 96, 125, 126

FBA Flux Balance Analysis 14, 17, 39–44, 46, 50–52, 56

GEM Genome-Scale Metabolic Model 5, 14, 17, 39–42, 44–46

HPO Hyper-parameter Optimization 1–5, 14, 15, 18, 19, 24, 27, 28, 30–32, 34, 36–38, 47, 60, 61, 64–68, 70–73, 81, 83, 85–88, 91–93, 95, 96, 98, 99, 101, 102

LOO Leave-One-Out 1, 19, 23, 24, 28, 38

MAE Mean Absolute Error 49

MINN Metabolic-Informed Neural Network 2, 14, 39, 40, 42–47, 49–52, 55–57

ML Machine Learning 5, 9–14, 16–18, 25, 27, 28, 38–40, 42–44, 50, 51, 55–57, 59, 60, 62–64, 73, 76, 81–85, 88, 99–104, 125–127

MLOps Machine Learning Operations 3, 5, 12, 60, 62–64, 70, 82–85, 87, 88, 90, 91, 98, 100–104, 126, 127

MLP Multi-Layer Perceptron 1, 20–22, 26–30, 32, 38, 61, 68, 69, 87, 88, 91

MSE Mean Squared Error 2, 46, 50, 51, 72, 73

NN Neural Network 2, 5, 10, 13–20, 22, 25, 26, 28, 30, 31, 33, 34, 37–39, 42, 45, 46, 49, 50, 55–57, 60–62, 64–66, 68, 69, 71–74, 76, 77, 81, 82, 88, 89, 92, 94, 95, 103, 125–127

pFBA parsimonious Flux Balance Analysis 39, 41, 50

PINN Physics-Informed Neural Network 15, 16, 42

RMSE Root Mean Squared Error 2, 5, 49, 52, 54, 69, 72–78

RNN Recurrent Neural Network 22, 26

SHAP SHapley Additive exPlanations 3, 5, 66, 69, 70, 76, 78–82

SNP Single-Nucleotide Polymorphism 5, 12, 59, 60, 62, 64–66, 68, 70, 73, 76, 78, 81, 82, 103, 126, 127

XAI Explainable Artificial Intelligence 61, 63, 65, 103, 126, 127

Chapter 1

Introduction

Machine Learning (ML) and Deep Learning (DL) methods are increasingly applied to omics data to model complex relationships between molecular features and phenotypic or functional outcomes. These approaches offer substantial flexibility and predictive power, but they also introduce methodological and practical challenges, particularly in settings characterized by high dimensionality, limited sample sizes, and heterogeneous data sources. As a result, the reliability and interpretability of ML results in omics depend not only on model choice, but also on evaluation protocols, training strategies, a nd experimental organization.

This PhD thesis investigates how ML methods can be applied more reliably and transparently to omics data analysis. The work focuses on two complementary aspects. The first concerns methodological choices that affect model generalization, evaluation, and interpretability, with emphasis on DL models trained in data-scarce regimes. The second addresses applied and technical considerations arising in real-world omics studies, including robust model design, explainability, and the organization of complex experimental workflows.

The contributions of the thesis are developed through a combination of methodological analyses and applied case studies in foodomics and agronomy. Together, these studies aim to clarify how sound experimental practices and appropriate tooling can strengthen applied ML research in omics, supporting results that are both scientifically meaningful and practically reusable.

1.1 Background and Motivation

1.1.1 Machine Learning and Deep Learning

ML focuses on the development of computational models that learn patterns and relationships directly from data. Instead of relying on explicitly programmed rules, ML methods infer these relationships by optimizing model parameters with respect to a given objective function. ML approaches are widely used for tasks such as classification, regression, and clustering, and they have become central tools in data-intensive research.

DL is a subfield of ML based on artificial Neural Networks (NNs) with multiple layers. These models learn hierarchical representations of the input data, where successive layers capture increasingly abstract features. Deep NNs have demonstrated strong performance in domains characterized by complex, non-linear relationships, such as computer vision and natural language processing.

From a modeling perspective, the strength of ML and DL methods lies in their flexibility and capacity to model complex relationships. At the same time, this flexibility introduces challenges related to model generalization, evaluation reliability, interpretability, and reproducibility. These challenges are particularly pronounced when ML and DL models are applied to high-dimensional datasets with limited sample sizes, a setting that is common in omics data analysis.

1.1.2 Omics Data as a Machine Learning Problem

Omics technologies, including genomics, transcriptomics, proteomics, and metabolomics, enable the large-scale measurement of molecular features in biological systems. These datasets are typically characterized by very high dimensionality, with thousands to millions of features, while the number of available samples is often limited due to experimental cost and practical constraints.

From a ML perspective, omics data present a challenging setting commonly described as the "large p, small n" problem, where the number of features far exceeds the number of observations. In addition, omics datasets are often affected by noise, batch effects, missing values, and heterogeneous measurement conditions. These properties complicate both model training and evaluation, and they limit the applicability of many standard statistical assumptions.

Despite these challenges, omics data provide a rich source of information for predictive modeling and pattern discovery. ML and DL methods are therefore increasingly adopted to model complex relationships between molecular features and phenotypic or functional outcomes. However, the successful application of these methods requires careful consideration of model design and evaluation strategies that are adapted to the specific characteristics of omics data.

1.1.3 Open Challenges in Omics Machine Learning

The application of ML and DL to omics data raises several open challenges that motivate this thesis. One major issue concerns model generalization. In high-dimensional, small-sample-size settings, models are prone to overfitting, achieving good performance on training data while failing to generalize to unseen samples. This risk is amplified for DL models, which often involve a large number of trainable parameters.

A second challenge relates to model evaluation. Performance estimates obtained from limited data can be highly sensitive to data splitting strategies and experimental design choices. Inadequate evaluation protocols may lead to optimistic or biased results, making it difficult to assess the true predictive value of a model or to compare results across studies.

Interpretability represents another important concern. While DL models can capture complex relationships in omics data, their predictions are often difficult to interpret. In many applications, understanding which features drive model predictions is essential for gaining insight into the underlying system and for building trust in the model outputs.

Finally, the increasing complexity of ML experiments introduces practical challenges related to experiment management and workflow organization. Variations in data preprocessing, model configurations, and training procedures can influence results and make systematic comparison across experiments difficult. As studies grow in scale and complexity, managing these aspects using ad hoc scripts becomes increasingly challenging.

While these challenges manifest in different ways, they are closely interconnected. Issues related to generalization, evaluation, and interpretability are often compounded by the increasing complexity of modern ML workflows. Addressing these challenges requires both sound methodological choices and well-organized experimental practices. Together, these considerations motivate the methodological and applied contributions developed in the following chapters of this thesis.

1.2 Contributions

The ideas, methods, figures, and results presented in this thesis have been published in peer-reviewed scientific papers, which are listed at the end of the manuscript (in Section 3.6). In summary, the main contributions of the thesis are organized as follows.

Chapter 2. This chapter presents methodological contributions to ML for omics data. It investigates evaluation and training practices for DL models in high-dimensional, low-sample-size settings, with particular focus on the interaction between CV and

ES. In addition, it analyzes hybrid modeling approaches that combine data-driven learning with mechanistic constraints, framing these models from a multitask learning perspective and examining how objective formulation and task balancing affect training stability model behavior.

Chapter 3. This chapter focuses on applied ML and experimental practice in omics research. It introduces an explainable DL framework for wheat Single-Nucleotide Polymorphism (SNP) data analysis that achieves statistically significant improvements over existing benchmarks. The chapter also investigates the use of Machine Learning Operations (MLOps) frameworks to support research-stage ML workflows, demonstrating how structured experiment management can improve reproducibility, transparency, and organization in applied agronomic studies.

Chapter 2

Evaluation Strategies and Hybrid Models for Omics Machine Learning

This chapter focuses on methodological aspects that affect the reliability of DL models applied to omics data. It addresses two related challenges: the design of sound evaluation protocols for NNs trained on small and high-dimensional datasets, and the optimization of hybrid models that combine data-driven objectives with mechanistic constraints. Through two complementary studies, the chapter highlights how methodological choices in training, evaluation, and loss formulation influence model performance, stability, and interpretability, and provides practical guidance for their application in omics research.

2.1 Introduction

ML models applied to omics data operate in challenging regimes characterized by high dimensionality and limited sample sizes. In these settings, methodological choices related to model evaluation, training procedures, and objective formulation can have a strong impact on the reliability and interpretability of results. As a consequence, advances in methodology are often as important as improvements in model architectures.

This chapter focuses on two methodological aspects that are particularly relevant for DL in omics applications. The first concerns the design of reliable evaluation and training strategies, with emphasis on the integration of CV and ES in NN models. Although these techniques are widely used in practice, their combination is often handled inconsistently, leading to biased performance estimates or unintended data leakage.

The second aspect addresses learning under multiple objectives in hybrid modeling settings. In several scientific domains, data-driven models are increasingly combined with biological and mechanistic knowledge to improve generalization and

enforce consistency with known system constraints. In systems biology, Genome-Scale Metabolic Models (GEMs) and Flux Balance Analysis (FBA) provide a structured description of metabolic behavior, while NNs offer flexibility in learning from data. Hybrid approaches that combine these components introduce multi-objective optimization problems, where data-driven and mechanistic objectives must be satisfied simultaneously.

From a methodological perspective, hybrid models raise questions that go beyond architectural design and concern how multiple optimization objectives should be formulated, balanced, and evaluated during training. In this chapter, these questions are examined through the lens of multitask learning, focusing on how competing objectives influence optimization dynamics, generalization behavior, and interpretability. The chapter addresses these issues through two complementary studies: a systematic analysis of evaluation practices for NNs in omics data, and an investigation of multitask learning behavior in a Metabolic-Informed Neural Network (MINN).

2.2 Methodological Background

Reliable modeling in omics data analysis depends on how evaluation strategies, training procedures, and objective formulations are defined and applied in practice. The following sections outline methodological considerations related to these aspects, with particular emphasis on settings characterized by high dimensionality and limited sample sizes.

2.2.1 Model Evaluation in High-Dimensional Settings

Evaluating the performance of ML models in high-dimensional and small-sample-size settings is inherently challenging. In such scenarios, performance estimates can exhibit high variance and strong sensitivity to experimental design choices, including data splitting strategies and hyper-parameter selection procedures.

CV is widely used to obtain more robust estimates of model performance by averaging results across multiple data partitions. However, when applied to complex models such as deep NNs, CV requires careful integration with other training components, including model selection and ES. Improper handling of validation data or reuse of test data during model tuning can lead to data leakage and overly optimistic performance estimates.

In practice, these evaluation challenges manifest through subtle interactions between data partitioning, model selection, and training control mechanisms. As a result, evaluation protocols must be specified with precision, particularly when CV is combined with ES or Hyper-parameter Optimization (HPO). These interactions form the basis for the empirical analysis developed in the first study of this chapter.

2.2.2 Training Strategies and Regularization

Training deep NNs in high-dimensional settings requires strategies that balance model flexibility with the risk of overfitting. Regularization techniques are commonly employed to constrain model complexity and improve generalization. These include architectural choices, parameter penalization, and training procedures that limit excessive adaptation to the training data.

ES is one such strategy, where training is halted based on performance on a validation set to prevent degradation of generalization performance. While effective in practice, ES introduces additional methodological considerations when combined with CV or HPO, as it implicitly relies on data partitioning choices.

Beyond ES, training strategies such as controlled model capacity, parameter sharing, and task-related constraints can further influence generalization behavior. Selecting appropriate training and regularization mechanisms is therefore a key methodological decision, particularly in small-sample omics settings.

2.2.3 Hybrid and Mechanism-Informed Neural Networks

In several scientific domains, purely data-driven models are increasingly combined with domain knowledge to improve generalization, robustness, and plausibility of predictions. This has led to the development of hybrid or informed NNs, where prior knowledge is embedded into the learning process through architectural constraints, custom loss functions, or optimization procedures. A prominent example is represented by Physics-Informed Neural Networks (PINNs), which incorporate physical laws directly into NNs training.

In the context of omics data analysis, similar challenges arise. Data-driven models can capture complex relationships between molecular features and system-level outputs, but they often require large datasets and may produce solutions that are inconsistent with known biological mechanisms. Conversely, mechanistic models provide structured constraints but are limited by incomplete knowledge and restricted flexibility. Hybrid approaches aim to combine these complementary strengths by guiding learning with mechanistic principles while retaining the flexibility of NNs.

From a methodological standpoint, hybrid and informed NNs introduce optimization problems involving multiple loss components that reflect distinct modeling objectives. These objectives typically correspond to data fidelity terms and constraint-enforcing terms derived from mechanistic knowledge. Training such models therefore requires explicit design choices regarding loss formulation, scaling, and interaction.

This setting can be formalized as a form of multitask learning in which a shared representation is optimized with respect to heterogeneous objectives rather than multiple prediction targets. Framing hybrid models in this way enables the analysis of task interference, objective dominance, and training stability using established

concepts from multitask learning theory. These considerations provide the conceptual basis for the second study presented in this chapter.

2.3 Related Works

Prior work has explored a range of strategies for evaluating NNs in data-scarce, high-dimensional settings, as well as for integrating mechanistic knowledge into data-driven models. The following review focuses on how these approaches have been implemented in practice and on the methodological limitations that remain unresolved.

2.3.1 Model Evaluation and Training Practices in Omics Machine Learning

A substantial body of work has examined evaluation strategies for ML models in data-scarce and high-dimensional settings, with CV emerging as a dominant approach for performance estimation. In parallel, ES has been widely adopted as a practical regularization mechanism in NNs training.

Despite their widespread adoption, the interaction between CV and ES has received limited systematic analysis in the context of DL [19, 70, 154, 156]. In many applied studies, when combined with CV, ES is integrated in an ad hoc manner, often without clear separation between training, validation, and test data. This can lead to unintended data leakage or optimistic bias in reported results. Prior work has highlighted that improper evaluation protocols can substantially affect performance estimates and undermine the comparability of models across studies [23, 40, 83, 144].

In omics-focused applications, these issues are further exacerbated by limited sample sizes and complex data preprocessing pipelines [78, 107, 126]. As a result, methodological clarity regarding evaluation design is critical. However, in much of the existing literature, the interaction between CV and ES in NNs training is handled implicitly, without explicit discussion of its methodological implications.

2.3.2 Hybrid and Mechanism-Informed Neural Network Models

Beyond purely data-driven approaches, there is growing interest in NN models that incorporate prior knowledge or mechanistic constraints. In scientific computing and engineering, PINNs have been proposed as a way to embed physical laws directly into the training process through constrained loss functions. These approaches aim to improve generalization and ensure that learned solutions remain consistent with known system behavior.

In systems biology, mechanistic models such as GEMs provide a structured representation of metabolic processes, often analyzed through FBA [25, 79, 111]. While these models offer interpretability and biological consistency, they are limited in their ability to incorporate heterogeneous omics data. To address this limitation, hybrid modeling approaches have been proposed that combine NNs with mechanistic models, allowing data-driven components to complement structured biological constraints.

From a ML perspective, these hybrid approaches introduce multi-objective optimization problems [51, 92, 157], where different loss components reflect data fidelity and mechanistic consistency. This setting can be interpreted as a form of multitask learning, in which shared representations are optimized with respect to multiple, potentially competing objectives. Prior work has shown that such interactions can lead to optimization conflicts and training instability if not carefully handled [51, 56, 136].

Although hybrid and mechanism-informed NNs have demonstrated promising empirical results, their optimization behavior is still not fully understood. In particular, interactions between heterogeneous loss components, task dominance effects, and evaluation choices are often addressed on a case-by-case basis. These considerations motivate the multitask learning perspective adopted in this chapter.

2.4 Integrating Cross-Validation and Early Stopping: Pitfalls, Bias, and Practical Guidelines for Omics Neural Networks

2.4.1 Introduction

NNs have achieved strong performance across several mainstream machine learning domains, particularly in areas where large, well-curated datasets are available, such as computer vision and natural language processing. This success has encouraged their adoption in a variety of scientific fields, including biomedical research and, more recently, omics-based applications.

In biomedical contexts such as medical microbiology, the increasing availability of data has supported the use of NNs. However, their performance has generally been more modest than in data-rich benchmark domains [71]. This contrast is informative for foodomics, where NNs are still not widely adopted, largely due to the scarcity of publicly available datasets [26, 33, 91].

Although NNs have achieved notable successes, their use in small, high-dimensional datasets remains challenging. In these settings, models are prone to overfitting and may fail to generalize to unseen data. To mitigate these risks, a range of regularization strategies has been proposed, including dropout [133], batch normalization [84], and data augmentation. Among these, ES is widely recognized

as a key component in the training of NNs [44, 68, 85, 130, 142, 156], and is now standard practice in domains where NNs are predominant [55].

Another crucial obstacle when working on small datasets is the performance evaluation of ML models. This is vital to ensure model robustness and reliability on unseen data. This issue is critical for model performance assessment on both unseen test data and validation data, in the latter case, for model selection.

CV techniques are widely used in scenarios with limited data, as they can offer more robust evaluations than the train, validation, and test splitting strategy [20, 78, 113, 159]. k-fold CV is one of the most commonly used methods within the family of CV techniques in the literature. By partitioning the data into several subsets and systematically using different subsets for training and validation, k-fold CV offers a robust framework for model evaluation. This approach is especially pertinent in the biomedical [11, 12, 13, 14, 64, 65, 66, 107] and foodomics [126] fields, where the sample size can be limited, underscoring the need for robust model performance evaluation to guide model selection.

Despite the widespread recognition of k-fold CV and ES as critical techniques in ML, there is a significant gap in the literature regarding guidelines or comprehensive research on effectively combining these methodologies within NN models, especially in foodomics. While both techniques have been extensively explored individually in many methodological papers [19, 70, 154, 156] and practical applications [55], their integration remains largely unexplored. This lack of methodological clarity poses a challenge in the foodomics field, where researchers seem to implicitly propose new methods to combine these techniques that, however, lack theoretical or empirical analysis and support from the previous literature [48, 125, 146].

These issues are not confined to foodomics. Comparable integration problems have been observed in bioinformatics and medical AI, where k-fold CV and ES have sometimes been combined in ways that blur the separation between monitoring and evaluation, or reuse the same data across HPO and testing [23, 40, 83, 144, 153].

Both k-fold CV and ES are particularly relevant in foodomics because datasets are often small, noisy, and high-dimensional, conditions that make NNs especially prone to overfitting. k-fold CV maximizes the use of limited samples for evaluation, while ES prevents models from memorizing the training data by stopping at the optimal epoch. However, their integration raises a central methodological challenge: the risk of overlap between evaluation folds and ES monitoring data. When the same fold is used for both purposes, the resulting estimates can become optimistically biased, and this effect is amplified in small-sample omics contexts.

By reviewing the recent literature, this review does not only summarize how k-fold CV and ES have been applied in foodomics, but also highlights the critical methodological issues that arise when combining them. We analyze recurrent problems such as the overlap between evaluation folds and ES monitoring data, the use of suboptimal

validation strategies, or the reuse of the same data across HPO and evaluation stages. Building on this analysis, we propose a recommended approach that addresses these challenges and reduces risks of bias or leakage.

While our proposal cannot yet replace the need for stronger theoretical and empirical validation, it provides a practical framework to guide future studies and move toward more standardized practices in the field.

2.4.2 Review Methodology

This work should be considered a *methodological scoping review*. We followed some systematic principles (clear query, inclusion and exclusion criteria), but the main goal was not to cover every single study. Instead, we focused on comparing how k-fold CV and ES were applied in Foodomics NNs research.

We searched the literature using Google Scholar[1] as the main database. This choice was made because it allows full-text searches, which was important since in many papers the use of CV or ES is not mentioned in the abstract. We only included studies published from 2000 onward to capture recent trends.

The query used was:
food OR cheese OR wine "neural network" OR NN OR "deep learning" "early stopping" omics OR genomic OR microbial OR bacterial OR strain OR biological "cross-validation" OR "cross validation" OR kfold OR "k-fold"

We also checked the references of review papers on food applications and included some additional studies from there.

The criteria were the following:

- Excluded: studies not related to food applications; studies not using genomic, metabolomic, or proteomic data; review papers (these were only used to expand our pool and are discussed in the Introduction Section 2.4.1).
- Included: studies that either used k-fold or Leave-One-Out (LOO) CV or applied ES in NNs training, with the main focus on the ones that integrated both methods.

After applying these rules, we selected 22 studies that we analyzed in detail in this review. Rather than aiming to cover the entire field, we focused on a representative set of works that clearly show how k-fold CV and ES have been applied in Foodomics. This allowed us to compare the different strategies in depth and highlight common issues and open challenges.

[1] `http://scholar.google.com/`

2.4.3 Background

This section outlines the key concepts and techniques relevant to the integration of ES and k-fold CV in NNs training. The focus is on how these methods are applied in the context of foodomics, with particular attention to data characteristics, model architectures, and evaluation strategies. The aim is to provide the necessary background for understanding the methodological choices discussed in the following sections.

Omics Data

In the past two decades, high-throughput omics technologies have significantly advanced various molecular domains, including genomics, transcriptomics, proteomics, and metabolomics. However, the omics datasets generated in these research fields are characterized by high dimensionality and complexity, coupled with the challenge of often limited sample sizes. These characteristics present several challenges for computational analysis, driving the need for innovative approaches.

Foodomics Foodomics is at the intersection of these technological advancements and the field of nutrition science, employing omics technologies to explore the molecular characteristics of food components and their implications for health and disease. This holistic approach goes beyond traditional food science and nutrition by complementing it with omics analysis to enhance our understanding of food quality, safety, and nutritional value [26].

Machine learning and deep learning

While, in recent years advanced DL models have been introduced, such as attention-based transformers [140] and generative adversarial networks [57], because omics datasets, and especially foodomics, are often small in size, simpler NNs architectures like Multi-Layer Perceptrons (MLPs) and Convolutional Neural Networks (CNNs) are still the most commonly used.

MLPs are feedforward networks made up of multiple layers of neurons, as presented in Figure 2.1. Each neuron processes the inputs by applying weights and a non-linear function. This structure makes MLPs especially suitable for tabular data, where the relationships between features are not spatial or sequential [118].

CNNs are designed to capture spatial patterns in the data and are most known for their use in image analysis [75]. They apply filters to the input data to detect local features. Although CNNs were originally built for image processing, they have also been used for one-dimensional data such as spectra or sequences, making them useful in the field of foodomics.

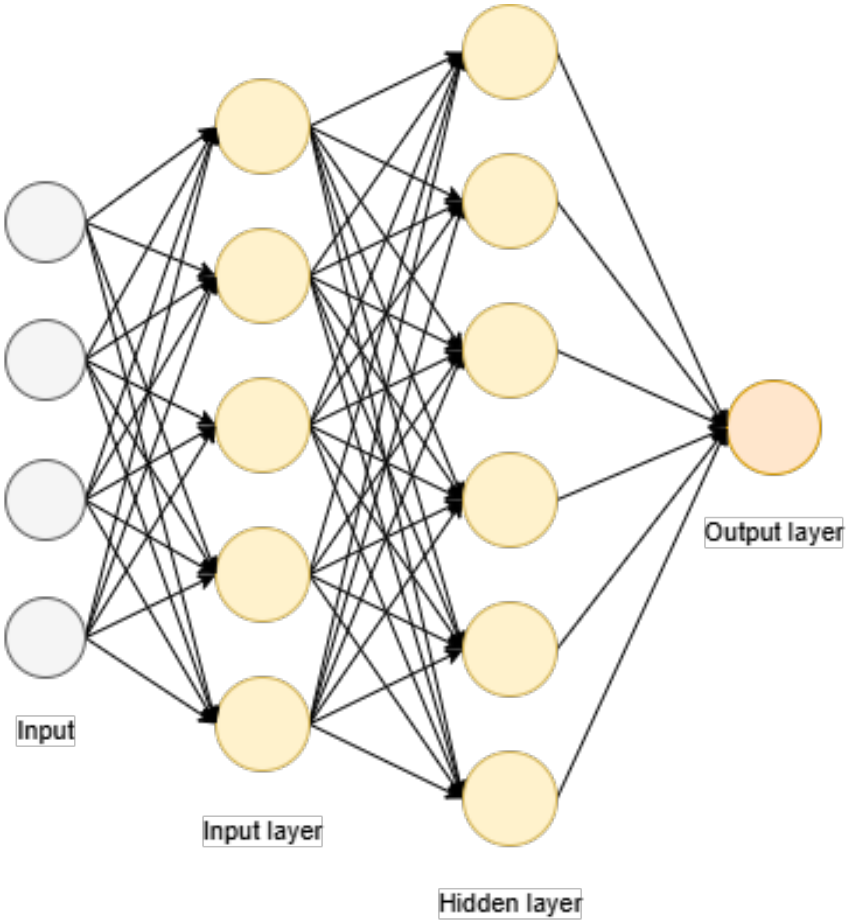


Figure 2.1: *Structure of a MLP with one hidden layer. Each circle represents a neuron, and arrows indicate the direction of data flow in this feedforward architecture.*

Recurrent Neural Networks (RNNs) differ from MLPs and CNNs because they are designed to handle sequential data. They include a memory component that allows the network to consider the order of inputs over time [131]. This makes RNNs a good choice for time series or natural language processing. Few applications are present in the foodomics literature [89, 120], where MLPs and CNNs are more common choices.

Cross-Validation Techniques

These techniques are used to divide the datasets into partitions with different scopes of training the model and evaluating it. Often, two partitions, one to train the model and one to evaluate its generalizability to new data, are not enough, as during the design of the model, multiple parameters and models have to be evaluated to estimate their performance on unseen data [138]. For this reason, generally the dataset is partitioned into 3 disjoint sets: one for training the model, one for evaluating its performance during model selection or hyper-parameter tuning, and a final one for evaluating the model on unseen data.

Several techniques have been proposed in the literature for generating these partitions. Iterative techniques, which we will present further in this section, have been proposed for generating validation partitions from the training data; so after an initial split of the dataset into training and testing partitions. But, as we will see in this review, these splitting methods have also been used to generate testing splits from the whole dataset.

Train, Validation and Test Splits In this setup the dataset is first split into training and testing splits, and then the training split is split further into training and validation splits (Figure 2.2). The training split is used for model training, the validation split to assess the performance of the model for hyper parameter tuning, model selection and in the case of the use of an ES mechanism is used to monitor NN model's performance and halt the training process before reaching the stage of overfitting, finally, the test split is devoted to the final model evaluation on unseen data [154].

This data splitting strategy is often used when training NNs on large datasets, as on the one hand, the size of the dataset guarantees a robust evaluation, and on the other hand, it presents a lower computation cost when compared to the iterative CV techniques that will be introduced in the following subsections [54].

However, when dealing with a limited sample size, the assessment of model performance becomes unstable due to the small size of the test and validation splits [154] and as the training phase generally takes much shorter time, CV techniques are more indicated to improve the stability of evaluations [152].

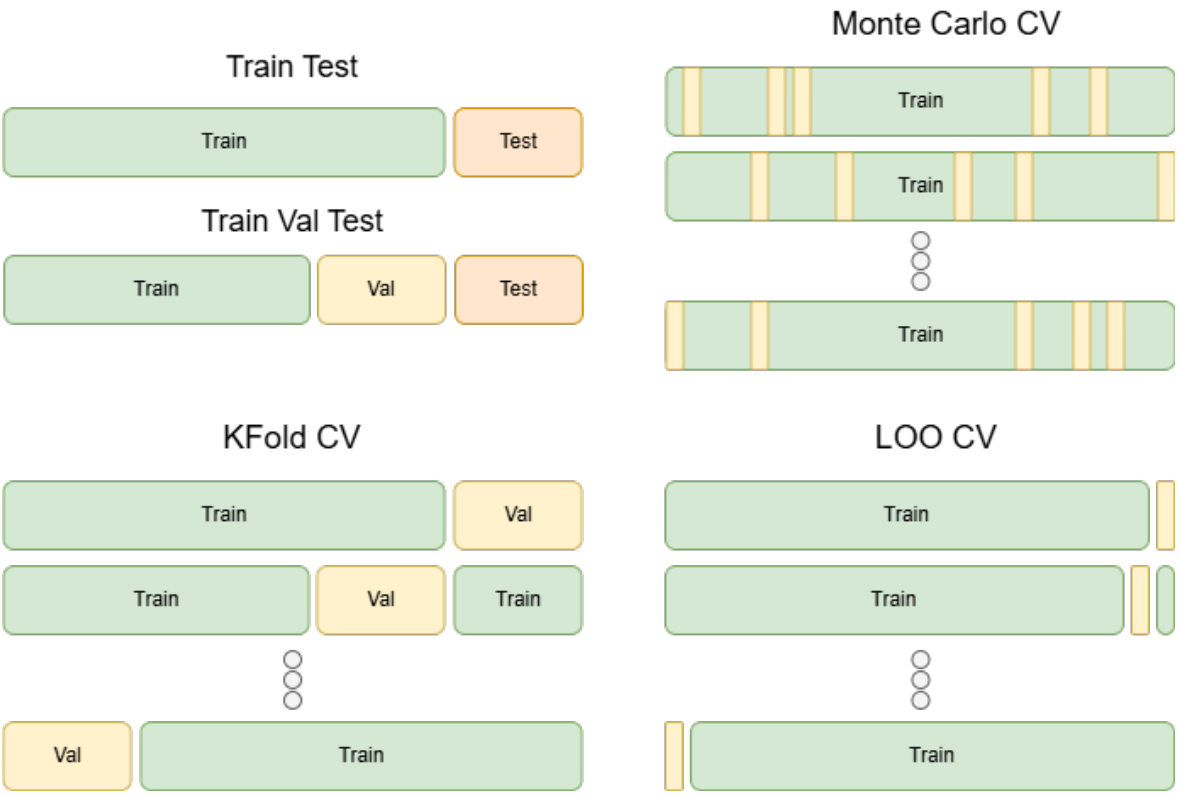


Figure 2.2: *Overview of common CV strategies. The figure shows: the train–validation–test split, Monte Carlo CV with random resampling, k-fold CV in which each fold is used once as a test set, and LOO CV, where k equals the sample size.*

Monte Carlo Cross-Validation The simplest improvement to the train, validation, and test splitting method is the Monte Carlo CV, which consists of sampling the evaluation partition repeatedly, which can be used to generate test splits if used on the whole dataset, or to generate validation splits if applied on the training partition after separating the testing partition from the dataset. In this setup, the model is trained and evaluated multiple times on different train and evaluation splits (Figure 2.2). So, it becomes possible to evaluate the robustness of the model across the different evaluation splits.

k-fold Cross-Validation K-fold CV is an improvement of the Monte Carlo CV method as it guarantees that any sample of the dataset takes part to the model assessment once. This is generally used for HPO and model selection, so applied on the training data after excluding the testing partition upstream. During k-fold CV, the dataset is divided into k subsets; the model training and evaluation are then repeated for k iterations, with each of the k subsets serving as the validation set once and the remaining subsets forming the training set (Figure 2.2). The evaluation metrics are then averaged across the k independent evaluations, and the standard deviation quantifies the robustness of the model [154]. In some setups, this evaluation process is used to evaluate the model [48, 144]; in these cases, the validation folds serve as test splits, and no train-test split is done upstream. This setup improves the robustness of model evaluation on unseen data, as k-fold CV reduces variability by averaging results across multiple, systematically varied test partitions.

Leave-One-Out Cross-Validation LOO CV represents an exhaustive variation of k-fold CV, where the number of folds, k, is set to equal the number of samples. In this approach, each observation is isolated as the evaluation set, with the rest of the dataset serving as the training set, and the training and evaluation are repeated for each observation of the dataset (Figure 2.2). Despite its thoroughness, LOO CV is rarely applied in practice, primarily due to its intensive computational cost.

Nested K-Fold Cross-Validation Nested CV extends k-fold by introducing two CV loops that decouple model evaluation from HPO. In this setup, the dataset is split into k_{outer} folds; at each outer iteration, one fold serves as the test split for final evaluation, while the remaining folds form the outer training split. Within this outer training split, an inner k_{inner}-fold CV is performed for HPO. The selected configuration is then retrained on the full outer training split and evaluated once on the outer test fold [28]. It should be noted that the computational cost of this procedure is very high, as the model goes through HPO k_{inner} times and is then trained and evaluated k_{outer} times. Indeed, it can become easily impractical for large datasets or models with intensive training.

Early Stopping

As NNs are highly flexible models, a crucial issue is to avoid overfitting to preserve the model's generalizability to new data. This issue is fundamental in the context of small datasets, where these models can easily go to a stage of overfitting where they learn irrelevant noise patterns from the train data split, with a consequent dramatic degradation of generalizability [39].
In this context, among many proposed methods, validation-based ES, generally referred to as ES, is widely considered a crucial technique to avoid overfitting when training NNs [44, 55, 68, 70, 85, 98, 130, 142].
In this setup, a portion of the dataset, disjoint from the train and test splits, is monitored during the training process. The training is halted once the model's performance on the validation set ceases to improve, preventing the model from learning noise or irrelevant patterns in the training data. This technique is crucial for NNs, as it anticipates, with the validation set performance, when the model's performance would start to degrade on unseen data.
Regarding the ES mechanism, an active branch of research proposes advanced methods to trigger the stopping mechanism [130, 142].

2.4.4 Applications in Food Omics Research

Researchers in the foodomics literature have proposed various ways to combine ES and k-fold CV, probably due to the lack of clear guidelines. However, some aspects of these methods diverge from the best practices in the broader literature on ML and DL [10, 70, 154].

Moreover, Figure 2.3 shows that, despite more than two decades of published research, there is no clear convergence or stratification of practices, with heterogeneous strategies continuing to appear across years.

This underscores the urgent need for guidelines for integrating these techniques.

In this section, we explore these diverse techniques and give a brief analysis in light of the available literature. First, we will present the approaches that best align with the literature on ES and CV with: Section 2.4.4 with approaches that used k-fold CV but did not use ES, then Section 2.4.4 with studies that used ES but did not use CV techniques, and then Section 2.4.4 with approaches that tried to merge the two techniques. Then, in Section 2.4.4 more unconventional strategies to integrate the ES and k-fold CV techniques are presented.

For clarity, Table 2.1 provides an overview of all the reviewed studies, summarizing their CV methods, ES strategies, integration choices, and main issues.

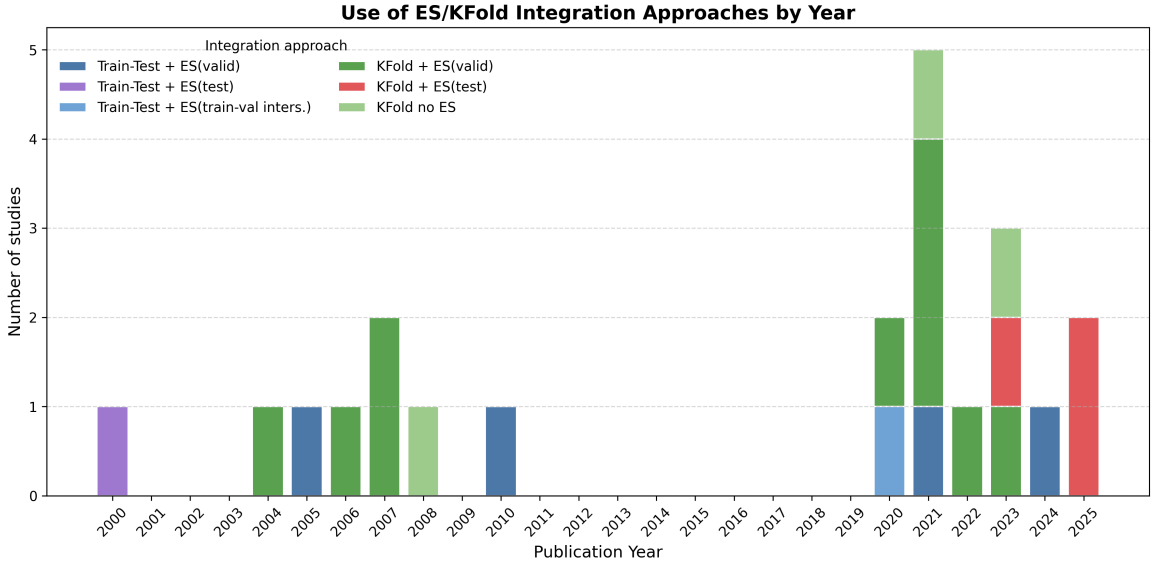


Figure 2.3: *Timeline of approaches combining ES and CV in foodomics studies. Each bar shows the number of studies per year adopting a given ES–CV integration strategy.*

K-Fold Cross-Validation without Early Stopping

A portion of the literature has not employed the ES mechanism based on validation data, but has proposed alternative strategies to halt the training process. These approaches have the advantage of being often more straightforward to implement; however, they are generally considered suboptimal in the literature as they expose the model to the risk of overfitting the training data and are generally considered suboptimal techniques compared to the ES based on a separated validation set [70, 98].

In the case of [120], where RNNs and MLPs were trained to identify umami peptides, k-fold CV was used to find optimal hyper-parameters, and the number of training epochs was included as a parameter in this search. Then, the optimal hyper-parameters were used to define the model to be evaluated on the test set.

In the study [89], CNN was used for bacterial identification from genome sequencing data. The dataset was initially partitioned into train, validation, and test splits. However, the authors did not use the ES technique. Instead, the validation learning curve was manually analyzed to determine the optimal number of epochs. Then this number of epochs was fixed as a stopping criterion for the k-fold CV evaluation on the entire dataset to measure the generalizability of the model on unseen data. It is worth considering that the data used to determine the optimal number of epochs has also been included in the test sets generated during the k-fold iterations, which could bias model evaluation. Lastly, [95] utilized a shallow NN, comprising only input and output layers, trained over a fixed number of epochs to regress the trichothecene

content in wheat.

Although alternative techniques to online ES based on validation data, such as setting a fixed number of epochs, analyzing the learning curve across the entire dataset, or defining the stopping point as a hyper parameter are generally easier to implement, they are considered suboptimal in the literature [98]. Fixing the number of epochs in advance carries a high risk of overfitting, as the optimal stopping point can vary significantly across different data splits, even when informed by prior model runs. Furthermore, treating the number of epochs as a hyper-parameter can affect the apparent performance of certain configurations during HPO, as differences in training duration may confound the evaluation of the actual model architecture or parameter choices. This may introduce noise into the optimization process and hinder the identification of genuinely optimal settings.

Train, Validation, and Test Split with ES

Another branch of the literature opted for the train, validation, and test splitting strategy instead of the k-fold CV. Performance evaluation begins with the validation set to estimate the model's generalization capability for the purpose of HPO and model selection, then the best models are evaluated on the test set to assess their performance on unseen data. In this setup, the ES mechanism is applied to the validation set, which is also utilized for performance measurement. Generally, this approach is favored in scenarios where data is plentiful, such as in computer vision and natural language processing domains; on the other hand, it is considered unstable in scenarios where data is limited [154].

Among the studies implementing the train, validation and test splitting, [135] presented a data-driven approach for classifying food safety alerts related to chemical and microbial contaminants in dairy products. This work utilized k-fold CV for the HPO for the classical ML models (e.g., k-nearest neighbors and SVM) but did not implement k-fold CV and HPO for the MLP model. Instead, the training data was partitioned in train and validation splits, and the latter was used for the ES mechanism.

Similarly, [160] explored the fermentation processes of penicillin and l-lysine using the MLP model. The dataset was divided into train, test, and validation splits instead of the k-fold CV. Also in this case, the validation set was used for ES monitoring. This approach was also replicated in [4], where the dataset was segmented into three splits, with the validation segment serving, this time, the dual purpose of ES and model selection.

Additionally, [112] employed this methodology to train an MLP model with a single hidden layer for predicting various food products' cooking stages. The dataset was divided into training, validation, and test sets for employing the ES mechanism. Notably, the authors reported that the model trained without ES yielded comparable results; this could be partially due to the significant portion of the training samples

(50%) designated for ES.

The use of the train, validation, and test splitting strategy in conjunction with the dynamic ES mechanism effectively maintains the benefits of the ES technique to avoid overfitting, but, on the other hand, gives up the robustness of the k-fold CV evaluation. A robust evaluation such as the one with k-fold CV becomes crucial when dealing with small datasets, which is often the case in the foodomics literature, as model evaluations can be unstable across different data splits. Moreover, the small sample size attenuates the computational cost introduced with the k-fold CV.

Early Stopping on the Test Data split Notably, in the literature a study employed the ES technique on the test split. In the study [116], an MLP model with one hidden layer was used to model the pH curves of cheese products. In this setting, the dataset is divided into only train and test splits, with the ES mechanism applied directly to the test data. Monitoring test data to halt the training of the NN model introduces a considerable risk of data leakage [85], as the model is stopped when it achieves the best fit on the test set, biasing its assessment on that data.

k-fold Cross-Validation and Early Stopping

Some studies in the foodomics literature explored the integration of both the ES technique and the k-fold CV method. However, this integration is not straightforward, and the lack of clear guidelines in the literature has led to the development of diverse approaches, each implementing these techniques in different ways.

The study by [109] models the antioxidant properties of peptides using CNNs. Here, the ES mechanism was exclusively integrated during the k-fold CV training phase. In this implementation, the ES mechanism in the k-fold CV evaluation was based on the validation fold, which was also used to evaluate the model's performance for model selection.
On the other hand, when the model parametrized with the best hyper-parameters was trained on the entire training set for the evaluation on the test set, the ES mechanism was not utilized, and training was halted at the 400^{th} epoch (as shown in Figure 2.4); which is generally advised against in the literature [55, 130] as it can easily produce overfitted or underfitted models.

In [147], for the identification of spoilage bacteria in milk, after dividing the data into train and test splits, the training data was used for HPO with LOO CV. The best hyper-parameters were then used to build the ML and NN models for training on the whole training set and evaluating on the test set. In this case, when training the NN models with the best hyper-parameters, ES was employed to halt the training, but the stopping mechanism monitored the loss on the test data split (which was then used for model evaluation on unseen data).

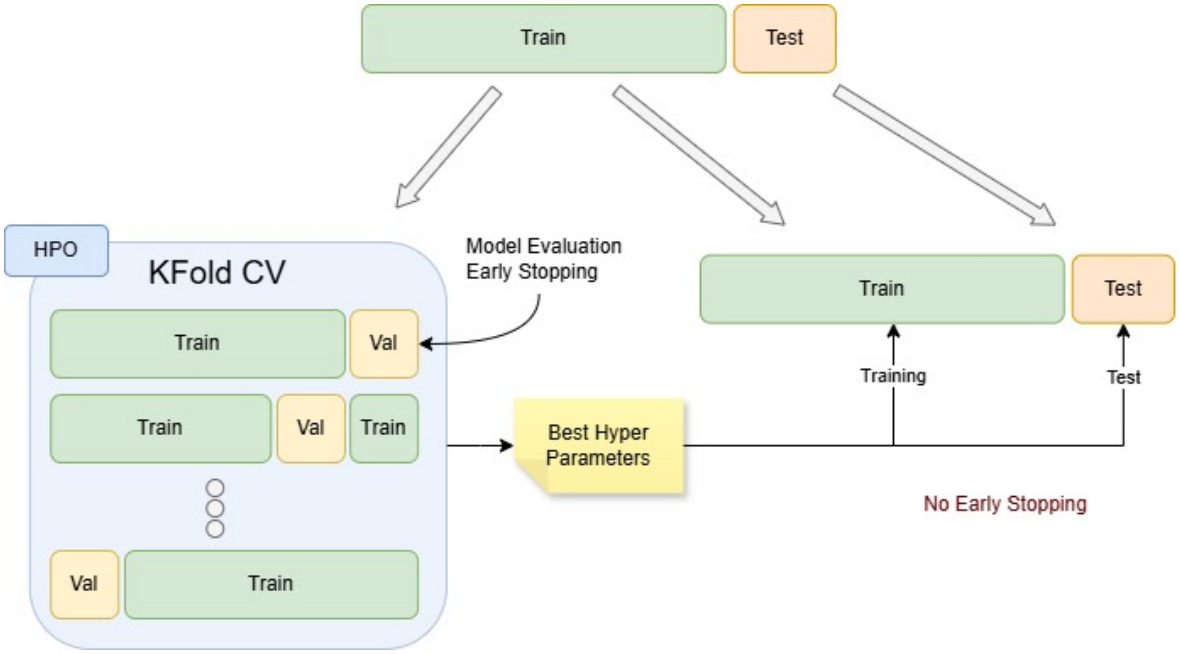


Figure 2.4: *Workflow used to combine k-fold CV and ES in [109]. The dataset is split into training and test sets, and k-fold CV is applied on the training set.*

Also [150], in the analysis of fermentation liquids, the dataset was initially split into train and test, with the first used with k-fold CV for feature selection. Later, the MLP model was trained on the training set (using the best features) and evaluated on the test split, similarly to [147], the testing data was also used for the ES mechanism.

Using the test data for ES monitoring undermines the fundamental principle of keeping test sets unseen. This approach invalidates the test evaluation and compromises the credibility of the model results.

K-Fold Cross-Validation for Test Splits and Early Stopping Monitoring In certain scenarios, k-fold CV has been employed to generate test splits, an alternative to the widely used train-test splitting setup, where the validation folds of the k-fold CV function as test folds in the different k-fold iterations. This strategy produces a more robust assessment by testing the model's generalizability across the multiple disjoint test splits generated by the k-fold CV method. However, in the study [144], the k-fold CV folds were used for both: evaluating the model on unseen data and for the ES mechanism (Figure 2.5). This practice diverges from established guidelines in the literature, such as those suggested by [68], as the test data is also used to influence the model's training and leads to optimistic evaluations.

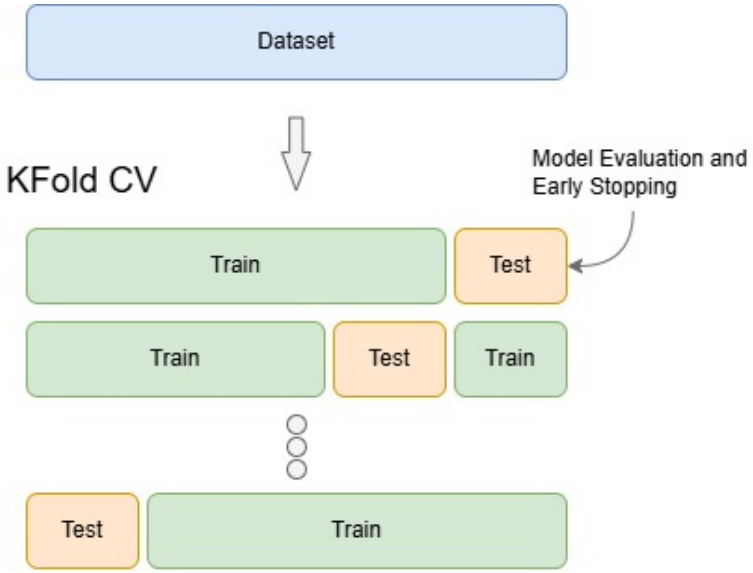


Figure 2.5: *In this setup, the dataset goes through a k-fold CV, where the evaluation folds function as test partitions in the k iterations. In this case, the evaluation of model performance and monitoring of the ES mechanism was done on the test fold.*

Alternative Methods for Integrating Early Stopping and K-Fold Cross-Validation

Surprisingly, the majority of the literature explored new and unconventional strategies for integrating the ES and k-fold CV methods in their studies. These approaches, introduced and solely utilized by the originating researchers, stand in contrast to the more widely adopted strategies discussed previously. By showing these approaches, we aim to present the more unconventional solutions brought forward and highlight advantages and discrepancies with the literature recommendations.

One Fold for Test and One for Validation This approach consists of modifying the k-fold CV by allocating not only one of the k folds for the model evaluation but also one further fold to trigger the ES mechanism. The study that presented this approach investigated the drying kinetics of sliced carrots using an MLP [48]. The authors implemented a pipeline where the 25% of the dataset was used for HPO with a 10-fold CV, where an additional fold served for monitoring the loss for the ES, leaving the eight remaining folds for training. Subsequently, another round of k-fold CV was conducted to evaluate the model initialized with the best hyper-parameters found during HPO. This time, the k-fold's validation folds function as test splits, and the k-fold evaluation is performed on the whole dataset, with 8 folds (Figure 2.6). A similar methodology was also applied in a previous study by the author in modeling the drying kinetics of the Echinacea angustifolia root [47]. In this study, a feed-forward NN with one hidden layer was first evaluated on 25% of the dataset for the purpose of HPO with a

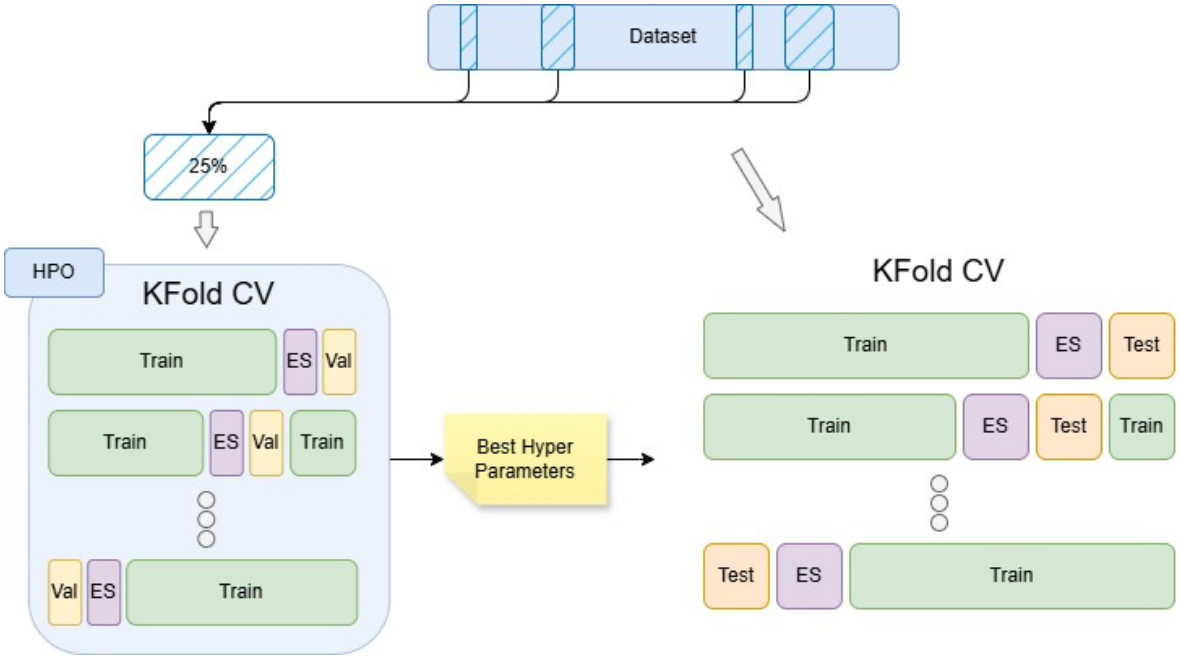


Figure 2.6: *Alternative use of k-fold CV with a separate fold for ES monitoring. A 25% subsample is used for HPO, and the selected hyperparameters are later evaluated using k-fold CV.*

10-Fold CV, where one fold was used for evaluation and another for ES (Figure 2.6). Then the best hyper-parameters configuration was used for evaluating the network on the whole dataset, this time by using a train, validation and test splitting, where the latter two were used respectively for ES monitoring and final model evaluation.

Even though this approach is unique in the foodomics literature and at least rare in the broader field of NN applications, it presents some advantages: the use of one fold for the ES procedure and a separate one for evaluating the model during k-fold CV avoids data leakage between the split used for model evaluation and the one for ES monitoring. Such unconventional designs, where ES monitoring and evaluation are assigned to distinct folds, explicitly separate model selection from model assessment. This separation helps reduce the risk of information leakage between training, monitoring, and evaluation, thereby strengthening the validity of the reported performance estimates.

On the other hand, the pipeline that features first an HPO with k-fold CV (on 25% of the dataset) and then a second evaluation (in the first case with k-fold CV and in the second with train, val, test splits) that involves the data used in the HPO suggests a risk of data leakage [22].

Inner Validation Split for Early Stopping A conceptually similar approach to the one described in Section 2.4.4 was adopted in a series of studies on wheat breeding

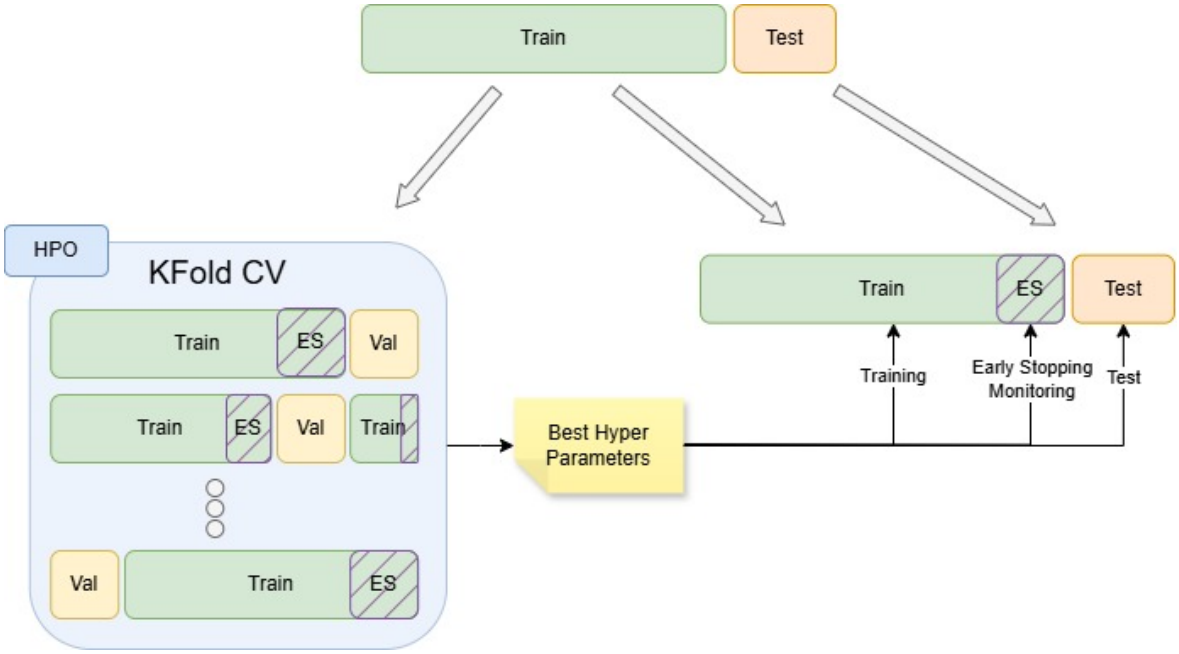


Figure 2.7: *Use of an inner validation split for ES during k-fold CV. A portion of the training data is reserved for ES monitoring while the remaining data are used for model fitting.*

using MLPs and CNNs [125, 126, 127]. The dataset was first partitioned into training and testing splits, and then the training samples were used for HPO with k-fold CV. Uniquely, here a portion of the training folds was allocated for ES monitoring, while the validation fold was used for performance evaluation (Figure 2.7). However, in [125, 126], a strict maximum number of epochs was also set as a hyper-parameter, with values of 200 and 150, which may have led to potential conflicts with the ES mechanism, like in the case of [88], highlighted previously in Section 2.4.4.

A comparable strategy was adopted by [121] for leaf disease classification with CNNs, where the validation fold, rather than the training folds, was split into two parts: one for ES monitoring and the other for model evaluation.

Both these approaches present the advantage of avoiding optimistic biases given by the ES based on the validation folds, as in the case of the approach in Section 2.4.4.

Early Stopping on the Train Data Split In [155], the study aimed to classify wines and liquors by analyzing sensor data. Here, ES monitoring was uniquely based on the training data split. However, using the train data for both fitting the models and monitoring of the ES is not advised in the literature; ES is ideally based on a distinct data split other than the training one. Using ES on the training split is likely to lead to overfitting [68] and, eventually, expose to the risk of memorization of the training samples [39].

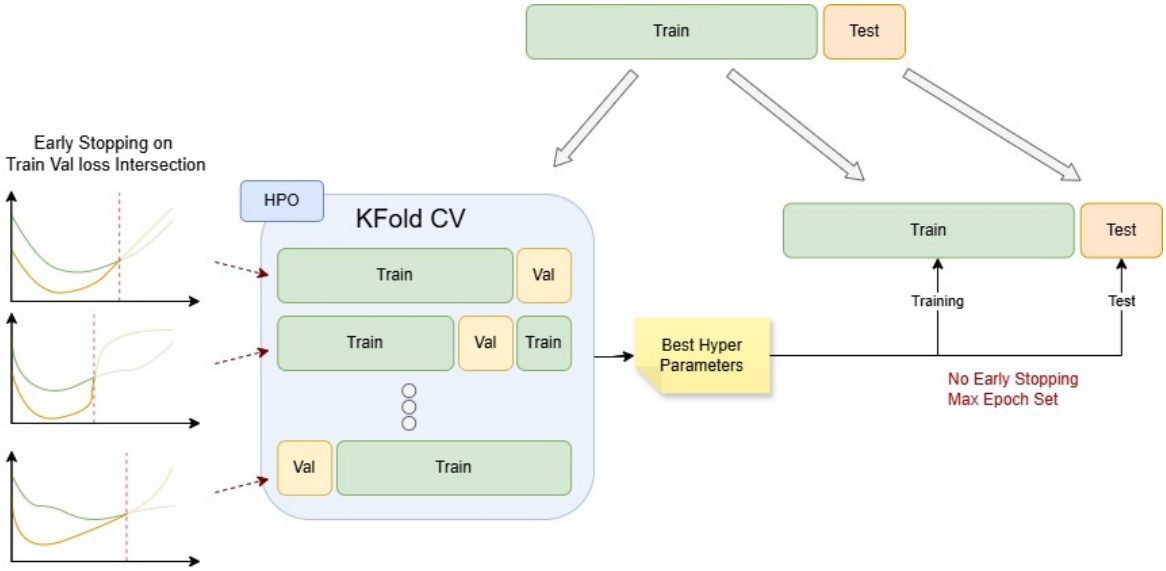


Figure 2.8: *Selection of the stopping epoch from validation loss curves during k-fold CV. The epoch chosen from the validation curves is later used as a fixed stopping point when retraining the model.*

Best Epoch Manually Selected from Validation Curves In this study [104] about predicting food adulteration through CNNs, the NN model underwent ES based on the validation fold during k-fold CV. However, for subsequent training on the entire dataset aimed at evaluating on a test set, the stopping epoch was predetermined based on the intersection of training and validation loss curves observed during the prior k-fold CV training iterations (Figure 2.8).

Basing the ES of the training on the point where the training and validation loss curves intersected diverges from the typical ES approach, which only considers the trend of the validation loss [70]. Moreover, in the evaluation of a test set, the ES was based on the k-fold CV iterations, which might not represent well this second learning process.

Evaluation of All the K-Fold Cross-Validation Models on Test Lastly, in the study by [146], NN models were trained using k-fold CV with the ES mechanisms monitoring the validation fold. In this setup, all the k models trained during the k-fold CV were evaluated on the test set, which was previously separated from the data used for k-fold CV, and their metrics were then averaged (Figure 2.9). This approach was motivated by the observation that the CV performances did not align with the performance on the test split.

This approach was also followed in the study from [88]. Here, a feed-forward NN with a single hidden layer is trained to determine the firmness of apple fruits through spectral scattering imagery. The dataset was split into train and test partitions, and the

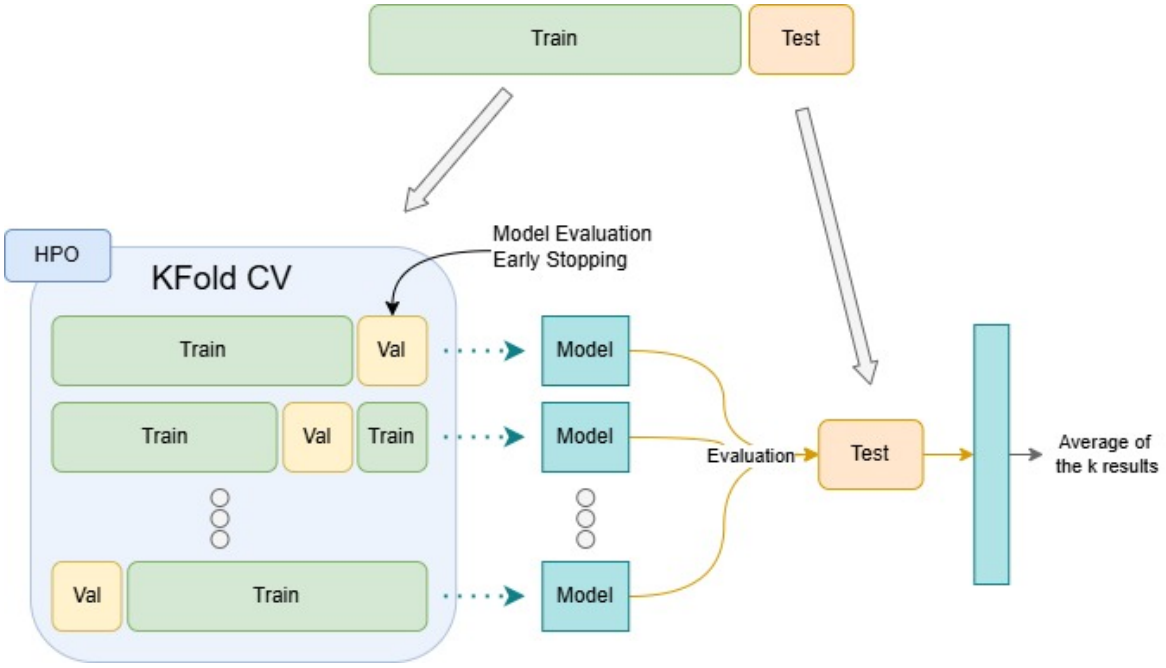


Figure 2.9: *Evaluation of all models trained during k-fold CV on the held-out test set. Performance metrics from the k models are averaged.*

train was used for HPO with k-fold CV. The ES mechanism was based on the validation fold of the k-fold CV split; thus, this fold served the dual purpose of stopping the NN's training with the ES mechanism and simultaneously assessing model performance for the HPO. Then, the k models trained during CV, with the best hyper-parameters, were evaluated on the test partition.

This approach seems to implicitly propose an ensemble model that recalls a variation of the Bagging (bootstrap aggregating) technique. Although in this case, the performance metrics are averaged across the 10 different models, instead of the model predictions. Regarding the ES mechanism during the k-fold CV, the use of the validation folds for both triggering the ES and evaluating the model has exposed the HPO process to the risk of data leakage [68].

2.4.5 Recommended Approach

As highlighted in Table 2.1, many of the analyzed studies show recurrent risks such as data leakage, biased performance estimates, or suboptimal use of ES. This motivates the need for clearer methodological guidelines for combining k-fold CV and ES.

Instead of the many possible implementations found in the literature, the strategy that most consistently aligns with best practices, and doesn't expose to the data leakage risk, is to allocate a dedicated validation split for ES monitoring. This can be done either by further partitioning the training data, as in [125, 126, 127], by

Algorithm 1: K-Fold CV with inner split for ES

Input : Dataset $\mathcal{D}$; number of folds K; ES patience P.
Output : Cross-validated performance estimate.
Partition $\mathcal{D}$ into K folds $\{\mathcal{F}_k\}_{k=1}^{K}$;
for $k = 1$ **to** K **do**
  $\mathcal{D}^{\text{train}} \leftarrow \mathcal{D} \setminus \mathcal{F}_k$;
  $\mathcal{D}^{\text{test}} \leftarrow \mathcal{F}_k$;
  Split $\mathcal{D}^{\text{train}}$ into $\mathcal{D}^{\text{fit}}$ and $\mathcal{D}^{\text{ES}}$;
  Train model on $\mathcal{D}^{\text{fit}}$ with ES monitored on $\mathcal{D}^{\text{ES}}$;
  Restore best weights and evaluate on $\mathcal{D}^{\text{test}}$;
  Store test score s_k;
Report mean $\pm$ std of $\{s_k\}$;

partitioning the validation data as in [121] (see Section 2.4.4), or by designating one of the training folds, as in [48] (see Section 2.4.4).

This ensures that ES monitoring does not overlap with the evaluation data, preserving the integrity of the CV estimates. It is also essential not to set overly restrictive epoch limits, as these may conflict with ES.

Instead, we recommend against using the validation fold of k-fold CV both for ES monitoring and for performance evaluation (Section 2.4.4). While simpler and commonly adopted in practice, this strategy risks optimistic bias, since the same fold is reused for both stopping and evaluation.

To facilitate reproducibility, Algorithm 1 presents pseudocode for the recommended approach (k-fold CV with an inner split for ES). This pseudocode block specify the sequence of data splits, training, monitoring, weight restoration, and evaluation, and can serve as a template for future studies.

Further theoretical and empirical work is still needed to validate these practices in foodomics contexts. Nonetheless, adopting the outlined strategies can substantially reduce risks of leakage and bias, and provide more reliable and reproducible performance estimates.

2.4.6 Future Directions

There is still a notable gap in the literature when it comes to practical guidelines for integrating k-fold CV and ES. Future studies should not introduce additional ad-hoc variants, but rather evaluate the approaches already used, identifying in which settings they are effective and where they expose models to risks such as leakage or biased estimates. This will require both theoretical analyses of their statistical properties and empirical validation across datasets of different sizes, dimensionalities, and noise levels.

Moreover, similar concerns emerge in adjacent fields. In plant genomics, disease resistance, and cancer diagnosis, for example, k-fold CV has been used to generate test splits, but without specifying how ES monitoring was performed [50, 83, 144]. In antimicrobial resistance prediction, a conventional train, validation, and test split with ES on the validation partition was adopted [153], a reasonable solution but one that can be fragile when the data is limited. In viral genome prediction, two-stage CV pipelines have been applied on the same data for both HPO and evaluation [40], a strategy that mirrors the leakage risks already observed in foodomics (Section 2.4.4). Instead, [23] applied Monte Carlo CV with an independent split for ES monitoring, which avoids leakage but is generally regarded as less robust than k-fold CV. Taken together, these examples confirm that the risks of leakage or reliance on suboptimal validation strategies are not specific to foodomics, and that clearer guidelines are needed across several data-scarce domains.

The ultimate goal of future work should be to provide researchers with clear and reliable tools for model evaluation, grounded in solid theoretical bases and consistent empirical evidence. Establishing such standards would make it easier to compare results across studies, reduce the risk of misleading conclusions, and promote reproducibility. Over time, this could help foodomics move toward well-established practices, similar to how the train, validation, and test split became a reference standard in data-rich domains [138].

2.4.7 Limitations

This review is partly limited by the fact that the reconstruction of how k-fold CV and, in particular, the ES mechanism were applied is based only on the descriptions provided in the papers. While these were sufficient to infer the methodological choices, without access to implementation code a residual risk of partial misinterpretation or oversimplification remains. Nevertheless, the approaches analyzed still provide valuable insight, both in highlighting good practices and in revealing common pitfalls that can guide new practitioners in designing more robust evaluation pipelines.

In addition, this review does not aim to cover every study in the wider field. Instead, we focused on a representative set of works that met our inclusion criteria. This choice allowed us to analyze the methodologies in depth and to identify recurring issues and open challenges, rather than aiming for a superficial overview of the entire literature.

2.4.8 Conclusion

Our review of recent publications applying NNs in foodomics has revealed a landscape rich with experimentation but lacking a unified direction. Each study we examined has navigated the integration of k-fold CV and ES with a distinct strategy, underscoring the field's experimental approach toward method integration. While showcasing the adaptability of researchers, this diversity also highlights a significant gap in the literature: the absence of comprehensive guidelines and established approaches to effectively combine these methodologies.

Notably, the chronological progression of the reviewed studies does not show a clear trend toward improvement or convergence (as shown in Figure 2.3). Instead, most of the different approaches appear to have been used uniformly in recent years, with no dominant practice emerging. This highlights the need for common standards and a careful review of the methods being used.

As the analysis has shown, several practices carry recurrent risks, such as data leakage between ES monitoring and evaluation folds, the reuse of the same data for HPO and testing, or the reliance on validation strategies that are weaker than k-fold CV. At the same time, we highlighted approaches that offer more robust alternatives, and we proposed a recommended implementation that minimizes these risks by combining k-fold CV with an inner split for ES monitoring. This framework does not replace the need for theoretical justification and broader empirical validation, but it offers a practical starting point for future work.

Establishing clearer guidelines and validation protocols will be essential as the field evolves. They will provide researchers with more reliable tools for evaluation, reduce the risk of misleading conclusions, and support comparability and reproducibility across studies. In the longer term, such standards could help foodomics converge toward shared and stable methodological practices, improving the reliability and comparability of NNs studies in data-limited settings.

2.5 Hybrid Neural Networks as Multi-Objective Systems: Balancing Data Fidelity and Mechanistic Constraints

2.5.1 Introduction

The phenotype of a cell is a complex interplay between its metabolic network, consisting of thousands of biochemical reactions, and the regulatory mechanisms controlling

Table 2.1: *Summary of reviewed studies with details on CV, ES, integration strategies, and reported issues.*

Reference	CV method	ES applied	Integration strategy	Issues / Notes
[120]	k-fold for HPO + separate test	N – epochs as hyperparameter	Epochs tuned within k-fold HPO; best config evaluated on test set	Risk of over/underfitting
[89]	k-fold + separate test	N – manual epoch selection	Epochs fixed by inspecting validation curve before k-fold CV; then applied to all folds	Possible bias: data used for epoch selection also reused in test folds
[95]	k-fold + separate test	N – fixed epochs	Model trained for a predefined number of epochs	Risk of over/underfitting
[135]	Hold-out (train/val/test)	Y – ES on validation split	MLP: train/val/test with ES; classical ML: k-fold HPO	DL models not benefitting from k-fold; less robust on small datasets
[160]	Hold-out (train/val/test)	Y – ES on validation split	Train/val/test with ES on validation set	Unstable on small datasets
[4]	Hold-out (train/val/test)	Y – ES on validation split	Validation split used both for ES and model selection	Dual use of validation may reduce robustness
[112]	Hold-out (train/val/test)	Y – ES on validation split	ES applied on validation split; large validation portion (50%)	Reduced training size, obtained similar results without ES
[116]	Hold-out (train/test)	Y – ES on test split	Training halted using test data performance	Severe data leakage: stopping on test biases evaluation
[109]	k-fold + separate test	Y – during k-fold only	ES applied on validation fold during k-fold CV; final training on full train set fixed at 400 epochs	Final training without ES; risk of over/underfitting
[147]	LOO CV for HPO + separate test	Y – ES on test split	After HPO, NN trained on full train set with ES on test split	Severe data leakage: test data influences training
[150]	k-fold for feature selection + separate test	Y – ES on test split	After feature selection, MLP trained on train set with ES on test split	Severe data leakage: stopping on test biases evaluation
[144]	k-fold used to generate test splits	Y – ES on test folds	Test folds used both for evaluation and ES monitoring	Data leakage: optimistic evaluation due to test-driven stopping
[47]	k-fold + separate test	Y – separate ES fold	HPO on 25% data with k-fold: one fold for eval, one for ES; final assessment with train/val/test	ES/eval separation avoids leakage, but HPO data reused downstream (risk of leakage)
[48]	Two-stage k-fold	Y – separate ES fold	First: HPO on 25% data with k-fold, then: k-fold eval on whole dataset. In both cases, one fold for eval and one for ES	ES/test separation avoids leakage, but HPO data reused downstream (risk of leakage)
[125, 126, 127]	k-fold on training + separate test	Y – inner split from training folds	Part of train reserved for ES monitoring within each training fold, max epochs included in HPO	Clear separation avoids leakage, ES may conflict with fixed max epochs
[121]	k-fold used to generate test splits	Y – inner split from validation fold	Validation folds subdivided: one part for ES, one for evaluation	Clear separation between ES and eval avoids leakage
[155]	k-fold for model evaluation	N – ES on training split	ES monitored on training loss	High risk of overfitting and memorization
[104]	k-fold + separate test	N – intersection-based ES	During k-fold, stopped at train/val loss intersection; final training without ES, fixed epochs	Final training without ES; risk of over/underfitting
[88, 146]	k-fold + separate test	Y – ES on validation folds	k models trained with ES during k-fold; all evaluated on test split; metrics averaged	Ensemble like model, but metrics instead of predictions averaged

diverse cellular functions. Mechanistic models, such as GEMs, provide a structured framework to integrate and connect available biological knowledge and to study emergent properties of cellular systems. GEMs mathematically represent cellular metabolism, summarizing known biochemical processes within an organism. One common approach to simulate cellular behavior using GEMs is Constraint-Based Modeling (CBM). Among these methods, FBA is particularly notable. FBA applies linear programming to optimize the distribution of metabolic fluxes, typically maximizing an objective such as biomass production under stoichiometric and environmental constraints. However, the predictive power of mechanistic models like FBA is limited by the completeness of current biological knowledge. Moreover, FBA often admits multiple feasible solutions. In such cases, parsimonious Flux Balance Analysis (pFBA) is commonly used to select the solution with the lowest overall flux, based on the assumption that cells minimize enzyme usage [79]. While effective in many scenarios, this assumption represents a simplification that does not fully capture the complexity of cellular regulation.

On the other hand, data-driven ML models can extract patterns from high-dimensional datasets, such as multi-omics data, without requiring explicit mechanistic assumptions. These models often demonstrate strong predictive performance, but their application in biology is limited by the scarcity of large, well-annotated datasets, due to experimental cost and complexity. In recent years, ML approaches have been explored for metabolic flux prediction using multi-omics data. Although FBA remains the standard mechanistic framework for this task, integrating omics information into CBM pipelines remains challenging [92]. Notably, recent work has shown that purely ML-based models trained on omics data can outperform FBA-based approaches in flux prediction [56].

Given the complementary strengths and limitations of mechanistic and data-driven approaches, there has been growing interest in combining ML with GEM-based modeling [157]. Hybrid models aim to integrate mechanistic knowledge into learning-based frameworks, leveraging biological constraints to guide model predictions. Recently, [51] introduced artificial metabolic networks, a NN framework embedding FBA constraints into the learning process, and demonstrated its effectiveness in predicting microbial growth rates from environmental conditions.

In this work, we build upon this line of research by adopting one of the hybrid configurations proposed by [51] and extending it to integrate multi-omics data as input. The resulting architecture, referred to here as a MINN, combines a data-driven NN with a mechanistic layer enforcing FBA constraints. The focus of this study is not on proposing a new hybrid architecture per se, but on addressing a central challenge that arises when training such models: the coexistence of competing optimization objectives.

In particular, when experimental flux measurements are not fully consistent with

the solution space defined by FBA constraints, the optimization of data-driven accuracy and mechanistic feasibility can become conflicting. Naively optimizing both objectives simultaneously may lead to unstable training dynamics or biased solutions. In this context, we frame the learning problem as a multitask setting, where fitting experimental data and satisfying mechanistic constraints are treated as distinct but coupled learning objectives.

To address this challenge, we investigate training strategies that dynamically balance these objectives during learning. Rather than assigning fixed weights to the data-driven and mechanistic losses, we explore multitask learning approaches in which training is initially guided toward mechanistic feasibility and progressively shifts toward data-driven accuracy. This curriculum-style optimization allows the model to first converge to biologically plausible solutions before refining predictions based on experimental data.

We evaluate this approach using the dataset analyzed by [56], which characterizes the metabolic behavior of *Escherichia coli* under different growth rates and single-gene knockouts [67]. As previously observed, experimentally measured fluxes may lie outside the feasible space of the GEM, exacerbating the conflict between mechanistic and data-driven objectives [136]. We therefore investigate several strategies to mitigate this conflict within the multitask training framework.

To summarize, in this work:

- We study the training of hybrid mechanistic–data-driven models from a multitask learning perspective, focusing on the dynamic balancing of mechanistic and data-driven objectives.
- We implement and evaluate a MINN architecture with multi-omics integration using an early concatenation strategy.
- We benchmark the predictive performance of the proposed approaches against pure ML methods on the dataset from [67].
- We investigate the impact of inconsistencies between experimental flux data and the FBA solution space.
- We explore and compare different hybrid optimization strategies to manage conflicts between learning objectives.

Through these analyses, we provide a methodological perspective on how hybrid ML–FBA models can be trained effectively when mechanistic and data-driven objectives compete. The results highlight the importance of training strategies in hybrid modeling and demonstrate how multitask optimization can improve both predictive accuracy and biological plausibility. In addition, this work aims to serve as a practical guide for selecting suitable MINN configurations depending on the specific modeling objective.

2.5.2 Related Works

Genome-Scale Metabolic Models and Flux Balance Analysis

GEMs are comprehensive reconstructions of an organism's metabolic network, representing the set of biochemical reactions encoded by its genome. They provide a structured and formalized representation of metabolism by integrating information on genes, enzymes, metabolites, and reactions [100, 114]. GEMs have been developed primarily for microorganisms, but reconstructions also exist for more complex systems, including multicellular organisms.

A typical microbial GEM includes hundreds to thousands of reactions and metabolites, with increasing complexity in multicompartment organisms such as yeast [132]. To analyze such large-scale networks, CBM approaches are commonly employed. Among these, FBA is one of the most widely used methods. FBA relies on the steady-state assumption, under which metabolite concentrations are assumed to remain constant over time, and metabolic fluxes are constrained by stoichiometry and reaction bounds [25, 111].

Under this formulation, metabolism can be expressed as a linear programming problem, where a feasible flux distribution is obtained by optimizing a predefined objective function, most commonly biomass production or the yield of a specific metabolite. While this formulation enables efficient simulation of metabolic behavior, stoichiometric constraints alone are often insufficient to identify biologically realistic flux distributions. As a result, FBA solutions are typically refined by incorporating additional context-specific constraints, such as nutrient availability or measured exchange fluxes, derived from experimental data or prior assumptions [115].

Despite its success, FBA often admits multiple feasible solutions that satisfy the same constraints and objective. To address this ambiguity, pFBA is frequently adopted to select solutions with minimal overall flux, based on the assumption that cells tend to minimize enzyme usage. While this assumption has proven effective in many applications, it represents a simplified view of cellular regulation and may fail to capture more complex metabolic strategies under certain conditions.

Integrating FBA and ML for Enhanced Metabolic Predictions

As discussed above, FBA provides a powerful framework to exploit the information encoded in GEMs for predicting cellular metabolic behavior. However, it presents several limitations. First, its predictive accuracy strongly depends on the availability of experimentally measured exchange fluxes. Second, the integration of multi-omics data is challenging, as measurements must be translated into flux constraints through iterative and often manually curated procedures. Third, GEM-based approaches focus exclusively on metabolism and typically do not capture the broader cellular

state. Finally, FBA solutions often favor high-yield metabolic strategies, potentially failing to represent alternative phenotypes characterized by high-rate, low-yield flux distributions [46, 139].

In parallel, ML approaches have gained attention as an alternative for metabolic prediction tasks, driven by their ability to extract patterns from high-dimensional data without explicit mechanistic assumptions [7, 56, 151]. While ML models can achieve strong predictive performance, their application in microbial physiology is limited by the small size of available datasets and by their black-box nature, which hampers mechanistic interpretability. At the same time, this flexibility makes ML models well suited for integrating heterogeneous data sources, including omics layers that cannot be directly incorporated into GEMs.

Motivated by the complementary strengths of mechanistic and data-driven approaches, several studies have explored their integration. As reviewed in [124] and [157], most existing methods rely on loose couplings, where ML is used either to generate inputs for FBA [38, 73, 103] or to post-process FBA outputs [36, 93]. While effective in specific settings, these approaches do not fully integrate mechanistic constraints within the learning process.

To date, only a limited number of works have proposed hybrid models that tightly couple FBA and ML. [51] introduced Artificial Metabolic NN, which embed FBA constraints directly into a NN through a mechanistic layer and a custom loss function, following the paradigm of knowledge-informed NNs such as PINNs [37]. More recently, [60] proposed FlowGAT, which combines GEM structure and FBA solutions within a graph NN to predict gene essentiality.

The MINN models considered in this work build upon the AMN framework and represent a tightly integrated hybrid approach that combines FBA constraints with multi-omics inputs for flux prediction. In this study, we further develop this framework by focusing on the optimization challenges that arise when data-driven objectives and mechanistic constraints are not fully aligned, and by exploring different strategies to mitigate conflicts between these competing objectives during training.

Multi-Task Learning and Multi-Objective Optimization

Multi-task learning (MTL) refers to the joint optimization of multiple learning objectives using a shared model representation [27]. By leveraging common structure across tasks, MTL can improve generalization and robustness compared to training separate models for each objective. In many practical applications, however, different tasks may not be fully aligned and can introduce competing optimization pressures during training.

A common approach to MTL consists of minimizing a weighted sum of task-specific loss functions. While simple to implement, this strategy implicitly assumes that tasks are compatible and that fixed loss weights can adequately balance their contributions.

When tasks compete, static weighting schemes may lead to unstable training dynamics or solutions that overly favor one objective at the expense of others.

To address these limitations, several works have framed multi-task learning as a multi-objective optimization problem, where each task corresponds to a distinct objective [129]. In this formulation, the goal is not to minimize a single scalar loss, but to identify solutions that achieve a suitable trade-off between competing objectives, often characterized through Pareto-optimality. This perspective has motivated the development of dynamic task-balancing strategies, including gradient-based methods [30], uncertainty-driven weighting [32], and scheduling approaches that adapt task importance during training.

This framework is particularly relevant for hybrid mechanistic–data-driven models. In such settings, mechanistic constraints introduce objectives that may conflict with data-driven predictive accuracy, especially when experimental observations are not fully consistent with the underlying mechanistic assumptions. Treating these objectives as separate tasks provides a principled way to analyze and manage their interaction during optimization.

In the context of the MINN framework, fitting experimental flux measurements and enforcing FBA constraints naturally define two competing learning objectives. The hybrid optimization strategies explored in this work - such as dynamic loss weighting and scheduled emphasis on mechanistic consistency - can therefore be interpreted as multi-task learning mechanisms. These approaches guide the training process toward mechanistically feasible solutions in early stages, before progressively shifting the focus toward improving data-driven predictive performance.

2.5.3 Methods

Dataset

The dataset analyzed in this work was originally published by [67] and consists of 29 chemostat experiments, in which *E. coli* was grown in glucose minimal medium. Wild-type strain K-12 was grown at 5 different dilution rates (D = 0.1, 0.2, 0.4, 0.5, and 0.7 h^{-1}), while 24 different single-knockout mutant strains were cultivated at fixed dilution rate (D = 0.2 h^{-1}). The same dataset was already used by [56] to test traditional ML for the prediction of metabolic fluxes from multi-omics data.
The dataset includes transcriptomic, proteomic, and fluxomic measurements. Transcript levels of 79 genes were measured using microarrays, while LC-MS/MS quantitative proteomics was used to quantify 60 proteins. ^{13}C-labeled metabolomics experiments were analyzed with MFA to estimate 47 metabolic fluxes: 37 reactions of the central carbon metabolism, 9 exchange fluxes (production or consumption of external metabolites) and biomass growth.

The metabolic model used by [67] to perform MFA is a core model that mainly rep-

resents the central carbon metabolism of *E. coli* and how it connects to the measured external metabolites. This model is much smaller and less complete than the GEM [52] integrated in the MINN. For this reason, the fluxomics data from [67] lie outside the solution space [136]. In most of our analyses we used the original fluxomics data, to highlight the ability of the MINN to reconcile MFA fluxomics data with the structure of the full-size GEMs. However, to investigate the impact of this discrepancy, we repeated some of the analyses with a second set of fluxes, now residing in the FBA solution space. This second set of fluxomics data is composed of the fluxes with the minimum Euclidean distance from the original ones, following an approach detailed in the supplementary material of [92] and we refer to it as *FBA fit* data.

GEM preparation

In this section we describe all the genome-scale metabolic reconstructions utilized to build the MINNs. The most recent GEM available for E. coli K-12 is iML1515 [101], but we opted for iAF1260 [52]. The two differ mainly for the more comprehensive coverage of accessory pathways of iML1515, which are relevant in complex environments like the human gut, but not for growth on minimal medium. On the other hand, the size of the GEM can heavily affect the complexity of the MINN: using a smaller GEM would improve the efficiency of our hybrid model by reducing the computational resources required for training. iAF1260 is reasonably smaller than iML1515 (2382 reactions vs. 2712) and is also the same model used by [56] in their analyses.

We applied two reduction strategies to decrease model complexity. First, Flux Variability Analysis (FVA) was used to remove reactions with zero flux under glucose minimal medium conditions, yielding the *FVA-reduced* GEM. Second, following [51], we generated 2000 FBA simulations under randomized glucose uptake rates and single-gene knockouts, and removed reactions consistently carrying zero flux, obtaining the *FBA-reduced* GEM. To investigate the impact of an extreme decrease in the genome-scale reconstruction size, we also built a MINN using the e_coli_core model [110], a manually reduced GEM focused on central carbon metabolism, which is the smallest model available in the BiGG database.

In Table 2.2 we summarize the dimensions of each GEM. The GEMs were downloaded from the BiGG database ([74]) and handled/modified using CBMPy 0.8.4 ([108]). In each model, reversible reactions were split into a forward and a reverse reaction using the built-in CBMPy function `cbmpy.CBTools.splitReversibleReactions`.

MINN architecture

This study introduces a MINN architecture designed to predict multiple fluxes using multi-omics data, which provide key insights for metabolic predictions but are challenging to integrate with FBA ([92]). Figure 2.10 illustrates the structure of our

	GEMs	
GEM name	original splitted reactions	reduced and splitted reactions
iAF1260	2957	NA
iAF1260 *FVA-reduced*	2957	1873
iAF1260 *FBA-reduced*	2957	587
e_coli_core	115	NA

Table 2.2: *Dimensions of all the GEM used in this analysis*

MINN architecture, built to predict fluxes measured in the dataset from [67] using proteomics, transcriptomics, and the measurements of two exchange fluxes, namely (*R_EX_glc__D_e*, *R_EX_o2_e*). The data are integrated using an early concatenation strategy [3], where the three omics datasets are combined into a single matrix that is fed into the MINN.

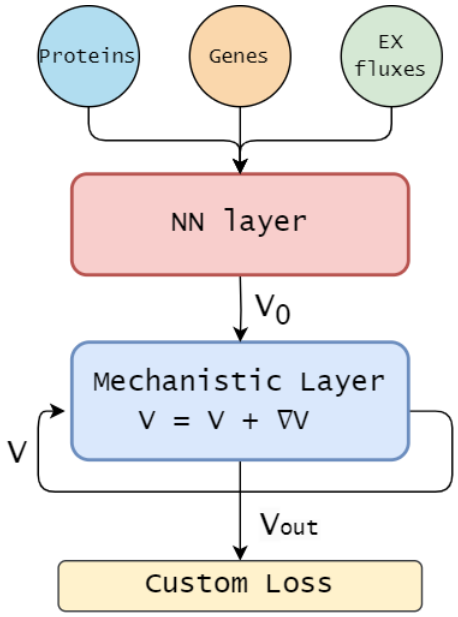


Figure 2.10: *Schematic representation of the MINN architecture. Multi-omics inputs (proteomics, transcriptomics, and exchange fluxes) are concatenated and passed through a feed-forward NN to produce an initial flux estimate, which is then refined in a mechanistic layer to obtain the final flux distribution.*

The first component in Figure 2.10 is a feed-forward NN whose input dimension equals the number of data features d_{in} and whose output dimension equals the number of reactions in the GEM, d_{out}.The output V_0 represents the NN's initial guess for the flux distribution. The second component, the mechanistic layer in Figure 2.10, consists of a gradient descent optimization loop. The mechanistic layer refines V_0 by performing

a gradient-descent step on the FBA loss L_{FBA}, producing the final prediction V_{out}. The NN weights are trained using a standard back-propagation algorithm and a custom loss function that considers both the data error and the FBA constraints L_{MINN}.

To formalize this, let the input data be $X \in \mathbb{R}^{N \times d_{in}}$ where N is the mini-batch dimension in the back-propagation algorithm and d_{in} is the number of features of X. The first NN part can be expressed as:

$$V_0 = \sigma(XW^h + b^h)W^{out} + b^{out} \tag{2.1}$$

with σ the ReLU activation function, $W^h \in \mathbb{R}^{d_{in} \times d_h}$, $b^h \in \mathbb{R}^{1 \times d_h}$, $W^{out} \in \mathbb{R}^{d_h \times d_{out}}$, $b^{out} \in \mathbb{R}^{1 \times d_{out}}$, the weights matrices and the biases of the input and hidden layer respectively, where d_h is the hidden layer dimension and d_{out} is the output dimension, which coincides with the dimension of the flux distribution. Then, V_0 is refined in the mechanistic layer through a gradient descent optimization. For simplicity, the notation refers to the simple case when the loop has one iteration:

$$V_{out} = V - lr\frac{\partial L_{\text{FBA}}}{\partial V} \tag{2.2}$$

$$L_{\text{FBA}} = \frac{1}{m}|SV|^2 + \frac{1}{n_{\text{in}}}|\text{ReLU}(P_{\text{in}}V - V_{\text{in}})|^2 + \frac{1}{n}|\text{ReLU}(-V)|^2$$

The first element represents the steady-state constraint of FBA, with S as the stoichiometric matrix of the GEM. The term m denotes the number of metabolites and serves as the normalization term. The second element represents the upper-bound constraint on the vector of fluxes V_{in}. Here, P_{in} represents the projection matrix that projects the flux distribution vector V into the dimension of V_{in}, while the normalization term n_{in} stands for the number of bounded fluxes. Lastly, the last element symbolizes the lower bound constraint, which is required since the GEM is built to ensure that all the fluxes are positive.

The custom loss used to train the weights of the MINN is:

$$\begin{aligned} L_{\text{MINN}} &= L_1 + L_2 + L_3 + L_4 \\ &= \frac{|P_{\text{ref}}V - V_{\text{ref}}|}{V_{\text{ref}}} + \frac{1}{m}|SV|^2 + \frac{1}{n_{\text{in}}}|\text{ReLU}(P_{\text{in}}V - V_{\text{in}})|^2 + \frac{1}{n}|\text{ReLU}(-V)|^2 \end{aligned} \tag{2.3}$$

where V_{ref} is the vector with the measured fluxes and P_{ref} a projection matrix that projects V to the dimension of V_{ref}; while the other elements represent the FBA constraints and are the same of L_{FBA}.

In [51], the authors used Mean Squared Error (MSE) as L_1 because they wanted to predict a single flux, specifically the growth rate. In our work, we use the Normalized

Error (NE) [56] to have a scale-invariant L_1 when predicting multiple fluxes in order to avoid favoring reactions with higher flux values.

In addition, during our analysis a conflict between the data-driven and mechanistic losses emerged. In order to mitigate this issue, we multiply $L1$ with a constant c, which allow us to adjust the balance between the two losses:

$$L_{\text{MINN-balanced}} = c \cdot L_1 + L_2 + L_3 + L_4 \tag{2.4}$$

The c constant becomes a hyper-parameter of the model, tuned using k-fold CV and the optimized value determines the best balance between L_1 and $(L_2 + L_3 + L_4)$.

Hybrid Optimization Strategies for Data-Driven and Mechanistic Integration Equation 2.3 shows that the MINN training objective is composed of two competing components: a data-driven loss, L_1, which promotes agreement with measured fluxes, and a mechanistic loss, L_{FBA}, which penalizes violations of flux balance constraints. In Equation 2.4, we introduced a coefficient c to control the relative importance of these two objectives, either through manual tuning or HPO.

While this static weighting provides a first level of control, it treats the optimization as a single-objective problem and assumes that a fixed trade-off between data-driven accuracy and mechanistic consistency is optimal throughout training. In practice, however, these objectives are often in conflict and operate at different scales, making a fixed weighting suboptimal across training stages.

We therefore frame MINN training as a multi-objective optimization problem and introduce three complementary strategies to dynamically balance the data-driven and mechanistic objectives during training. These approaches are designed to manage the trade-off between predictive accuracy and mechanistic fidelity, while explicitly accounting for differences in loss scale and learning dynamics.

Bound on Mechanistic Loss The first strategy introduces a bound on the mechanistic loss to prevent the model from excessively violating flux balance constraints in pursuit of improved data-driven performance. A fixed threshold is defined for L_{FBA}, and when this threshold is exceeded, a multiplicative penalty is applied to amplify the contribution of the mechanistic loss.

This approach softly constrains the optimization within a feasible region defined by the mechanistic model, discouraging solutions that diverge too far from biologically plausible flux distributions. Figure 2.11 provides a graphical illustration of this mechanism, showing how the loss function transitions from a linear regime to a steeper penalty once the threshold is crossed.

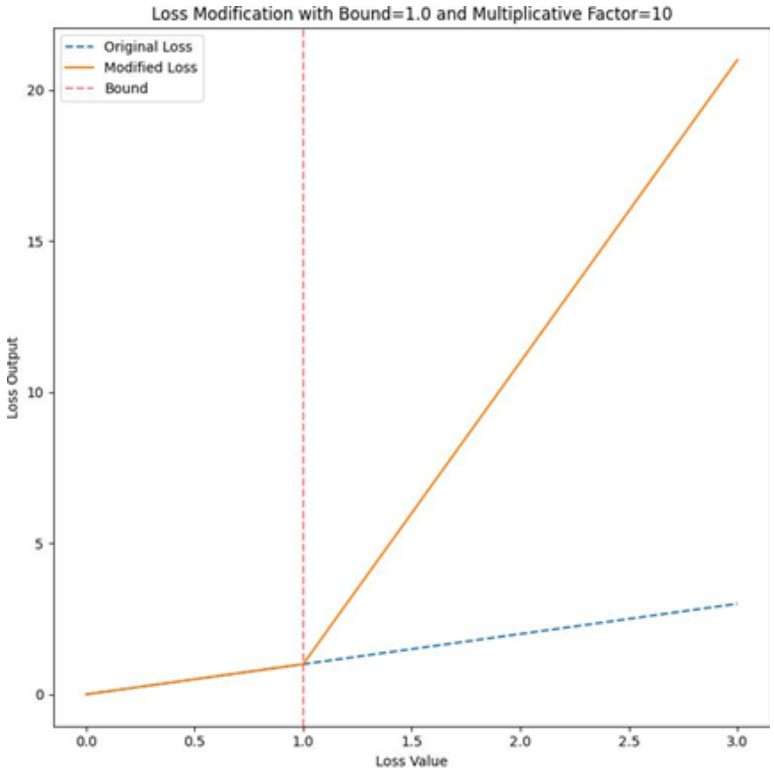


Figure 2.11: *Illustration of the mechanistic loss bound. The original loss (blue dashed line) increases linearly, while the modified loss (orange line) grows more rapidly once the mechanistic loss exceeds a predefined threshold (red vertical line). This mechanism penalizes excessive violations of flux balance constraints during training.*

Loss Balancing To address differences in scale between the data-driven and mechanistic losses, we also employed a loss balancing strategy inspired by [62]. In this approach, each loss term is normalized by the exponential moving average of its recent values. This normalization ensures that both objectives contribute comparably to gradient updates, preventing one loss from dominating the optimization due to scale alone.

By dynamically adjusting the relative magnitude of each loss, this strategy promotes more stable training and allows the model to jointly optimize both objectives without requiring manual tuning of fixed weights.

Loss Weight Scheduler Finally, we implemented a dynamic loss weight scheduler to explicitly control the relative importance of the mechanistic and data-driven objectives over the course of training. In the initial phase, the scheduler assigns greater weight to the mechanistic loss, guiding the model toward solutions that satisfy flux balance constraints. As training progresses, the scheduler gradually shifts emphasis toward the data-driven loss, allowing the model to improve predictive accuracy once

it has converged to a mechanistically plausible region of the solution space.

This curriculum-style optimization strategy reflects the sequential priorities of the learning process and provides a practical way to manage competing objectives. Figure 2.12 illustrates the evolution of loss weights over training epochs.

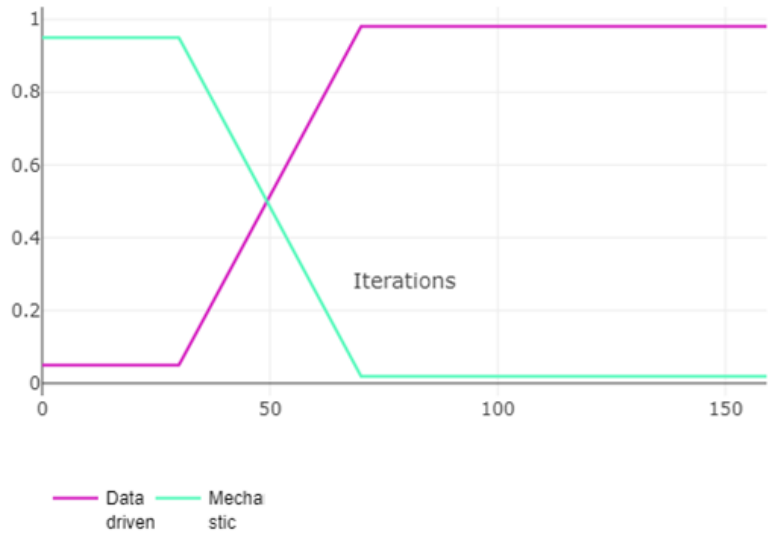


Figure 2.12: *Dynamic loss weight scheduling strategy. Training begins with a stronger emphasis on the mechanistic objective and progressively shifts toward the data-driven objective, allowing the model to first align with flux balance constraints before focusing on predictive performance.*

Together, these strategies provide a structured framework for balancing predictive accuracy and mechanistic fidelity in hybrid models, enabling controlled exploration of trade-offs between data-driven learning and constraint enforcement.

Computational setup

We adopted the same evaluation pipeline for all our MINN configurations to evaluate the MINN performance and have a fair comparison with the results obtained by [56]. It consists of a dual-loop CV process. The outer loop is a leave-one-out, and in each train loop, there is an inner loop of a k-fold with $k = 5$ to tune the hyper-parameters. The tuning concerns the dimension of the first hidden layer, the learning rate of the NN, the intensity of dropout and $L2$ regularization, and the c constant for the L_1 loss in the case of the MINN-c-balanced. For a consistent comparison with the work of [56], we employed identical metrics to evaluate all our experiments: the regression coefficient R^2, the Mean Absolute Error (MAE), the Root Mean Squared Error (RMSE) and the normalized error (NE). As described later, in some of our results, we also report the L_2 as a metric to measure the quality of the predicted flux distribution.

2.5.4 Results and Discussion

This work investigates how mechanistic constraints and data-driven objectives can be jointly optimized in hybrid neural–mechanistic models. While we report standard performance comparisons against pure ML approaches and mechanistic baselines such as pFBA, the primary focus of this analysis is on understanding and controlling the trade-off between predictive accuracy and mechanistic consistency within the MINN framework.

For clarity, we organize the results into three complementary groups. The first group (Table 2.3) establishes a performance baseline by comparing MINN against pure ML methods reported in [56], evaluating predictive accuracy on the 45 reference fluxes measured in the dataset. The second group (Table 2.4) addresses the core methodological contribution of this work by comparing different strategies for mitigating conflicts between data-driven and mechanistic objectives, assessing both flux prediction accuracy and the quality of the inferred flux distributions using L_2 as a proxy for mechanistic consistency.

MINN to predict measured fluxes

In the first group, as a baseline, we used the MINN architecture with an MSE as L_1 (MINN-MSE-base), as in [51]. As shown in Table 2.3, the results are already comparable with a Random Forest (RF), the best ML method in [56], and better than the NN approach. To avoid potential bias from the large discrepancies (up to two orders in magnitude) between the values of the fluxes we are predicting, we replaced the MSE in L_1 with a NE. As detailed in the Section 2.5.3, we multiply L_1 with a constant c, optimized during the CV. This method, incorporating the c parameter, is referred to as MINN-c-balanced, while the one without this adjustment as MINN-unbalanced. Although the change from MSE to NE ensures a scale-invariant L_1, it also reduces its magnitude, amplifying the conflict between losses as the mechanistic constraints gain more influence. For this reason, the MINN-unbalanced obtained worse results w.r.t. the MINN-MSE-base, but the MINN-c-balanced shows the best results, achieving comparable or better performance than the RF in three out of four metric averages.

Moreover, the MINN-c-balanced shows a marked reduction in standard deviation across splits. This suggests that the inclusion of biological constraints stabilizes the learning process, reduces overfitting, and leads to more robust and consistent predictions across the 29 leave-one-out evaluations.

Overall, these results indicate that the MINN achieves predictive performance comparable to, and in some cases exceeding, that of pure ML methods such as random forests and purely mechanistic approaches such as pFBA. In addition to predicting individual flux values, the MINN infers complete flux distributions, similarly to FBA-

	Dataset from [67]			
Model	R^2	MAE	RMSE	NE
pFBA*	0.823 ± 0.156	0.692 ± 0.733	1.058 ± 1.029	0.381 ± 0.185
NN*	0.967 ± 0.036	0.652 ± 0.945	0.936 ± 1.314	0.338 ± 0.338
RF*	0.970 ± 0.037	0.507 ± 0.804	0.729 ± 1.105	0.271 ± 0.347
MINN-MSE-base	0.950 ± 0.060	0.525 ± 0.525	0.736 ± 0.703	0.287 ± 0.282
MINN-unbalanced	0.951 ± 0.051	0.563 ± 0.739	0.814 ± 1.067	0.325 ± 0.442
MINN-c-balanced	0.950 ± 0.055	0.473 ± 0.480	0.678 ± 0.653	0.272 ± 0.280

Table 2.3: *Comparison of predictive performance between our proposed MINN-based approaches and purely mechanistic and ML methods from [56]. Metrics average and standard deviation over 29 leave-one-out splits.*

*results from [56]

based simulations.

The challenge of predicting multiple fluxes with heterogeneous magnitudes was effectively addressed by replacing the MSE with the normalized error (NE) in the data-driven loss term and introducing the balancing coefficient c to control the relative contribution of data-driven and mechanistic objectives. This design choice mitigates scale-induced imbalances between losses and improves both predictive accuracy and training stability.

Together, these results establish the MINN as a strong baseline relative to both pure ML and purely mechanistic approaches. Importantly, they provide the reference point for the analyses that follow, where the focus shifts to understanding how different optimization strategies influence the trade-off between data-driven performance and adherence to mechanistic constraints.

Balancing data-driven accuracy and mechanistic consistency

In this section, we analyze how different design and optimization choices affect the balance between predictive accuracy and adherence to mechanistic constraints in the MINN. In particular, we focus on the issue of conflicting objectives arising from the joint optimization of data-driven and mechanistic losses, as introduced in Section 2.5.3.

We investigate this problem from two complementary perspectives. First, we examine the impact of inconsistencies between the reference flux data and the feasible solution space of FBA, assessing how alignment with mechanistic constraints influences the quality of the predicted flux distributions. Second, we study how different hybrid optimization strategies explicitly control the trade-off between data-driven performance and mechanistic consistency during training.

Unless stated otherwise, all comparisons use the MINN-c-balanced configuration as a baseline, as it provides the best overall predictive performance among the tested models.

In the first case, we compare it with a configuration of the same MINN-c-balanced which uses different fluxes data, recalculated to be in the solution space of FBA. We described this process in 2.5.3 section. As shown in the first section of Table 2.4, this approach, named MINN-c-balanced *FBA fit*, does not reduce the quality of the fluxes prediction, represented by the four metrics, but it improves theL_2 by reducing its value by a third.

The results show that MINN maintains strong predictive performance even when the flux data are not in the FBA solution space. This suggests that its data-driven component can compensate for deviations from mechanistic constraints. However, using flux data that aligns with the FBA solution space improves the quality of the predicted flux distribution. This correction makes the optimization process easier by alleviating the issue of conflicting losses.

Regarding the hybrid optimization strategies, we compare the MINN-c-balanced with three other methods previously introduced in Section 2.5.3: *MINN-bound*, which penalizes violations of the mechanistic loss that exceed a threshold; *MINN-scheduler*, which gradually shifts the focus from mechanistic to data-driven loss during training; and *MINN-scheduler-bound*, which combines both approaches. From the second section of the table, we observe that the *MINN-c-balanced* model achieves the lowest RMSE, indicating the best performance on the data-driven task. However, this comes at the cost of a higher L_2, suggesting a trade-off where improved performance is achieved at the expense of mechanistic fidelity. On the other hand, the models incorporating a mechanistic bound, such as *MINN-bound* and *MINN-scheduler-bound*, show a small increase in RMSE and a modest reduction in L_2, suggesting a limited effect on improving the trade-off between mechanistic accuracy and data-driven performance. Interestingly, the *MINN-scheduler* model finds a better compromise between these objectives: at the cost of a moderate RMSE worsening, it achieves the lowest L_2 by a consistent margin. This demonstrates the strength of dynamic scheduling in keeping the solution close to the FBA feasible space, without heavily impacting data-driven performance.

Figure 2.13 provides a visual comparison of the performance of these methods, focusing on the trade-offs between mechanistic fit and data-driven task performance. The *MINN-scheduler* model demonstrates a balanced performance across both objectives, with a moderate decline in data-driven accuracy but a substantial reduction in mechanistic loss, positioning it much closer to the FBA feasible solution. On the other hand, the models incorporating a mechanistic bound (*MINN-bound* and *MINN-scheduler-bound*) show improvements in mechanistic fit but at a comparable cost in terms of data-driven performance.

	Dataset from [67]				
Model	R^2	MAE	RMSE	NE	L_2
MINN-c-balanced	0.950 ± 0.055	0.473 ± 0.480	0.678 ± 0.653	0.272 ± 0.280	$8.75 \cdot 10^{-5} \pm 2.95 \cdot 10^{-4}$
MINN-c-balanced *FBA fit*	0.957 ± 0.061	0.489 ± 0.497	0.706 ± 0.720	0.295 ± 0.403	$2.98 \cdot 10^{-5} \pm 8.75 \cdot 10^{-5}$
MINN-bound	0.949 ± 0.050	0.548 ± 0.556	0.790 ± 0.779	0.308 ± 0.314	$7.1 \cdot 10^{-5} \pm 2.1 \cdot 10^{-4}$
MINN-scheduler	0.949 ± 0.058	0.581 ± 0.823	0.833 ± 1.146	0.299 ± 0.320	$1.60 \cdot 10^{-5} \pm 7.63 \cdot 10^{-5}$
MINN-scheduler-bound	0.946 ± 0.062	0.560 ± 0.551	0.806 ± 0.761	0.304 ± 0.287	$7.47 \cdot 10^{-5} \pm 3.78 \cdot 10^{-4}$
MINN-divided loss	0.951 ± 0.050	0.489 ± 0.471	0.703 ± 0.648	0.281 ± 0.289	0.22 ± 0.29

Table 2.4: *Performance comparison of different methods addressing the issue of conflicting losses. Metrics average and standard deviation over 29 leave-one-out splits.*

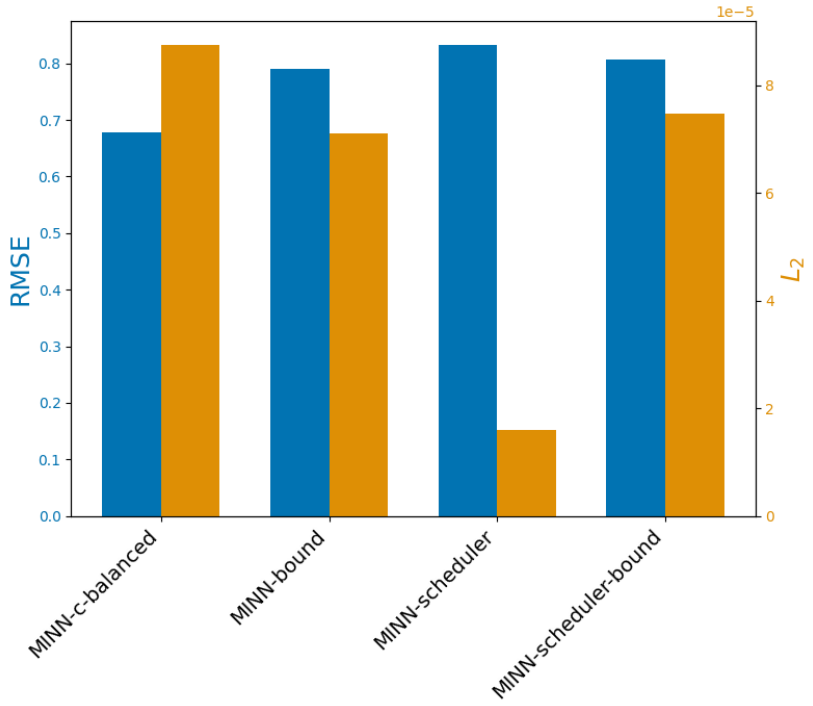


Figure 2.13: *Comparison of different methods based on data-driven task performance (RMSE) and mechanistic fit (L_2 loss), highlighting the trade-off between the two objectives.*

These results highlight a clear trade-off between optimizing for mechanistic fidelity and predictive accuracy. While the *MINN-c-balanced* method achieves the lowest RMSE, indicating better performance on the data-driven task, its high mechanistic loss shows that the model prioritizes predictive accuracy over adherence to mechanistic constraints. The models imposing a bound on the mechanistic loss, *MINN-bound* and *MINN-scheduler-bound*, show a even more evident trade-off between these objectives. While they slightly improve the mechanistic fit, this comes at the cost of data-driven performance. In contrast, the *MINN-scheduler* method effectively reduces the mechanistic loss, with only a marginal increase in RMSE.

Overall, these results highlight an inherent trade-off between predictive accuracy and mechanistic consistency in hybrid neural–mechanistic models. While static loss balancing can optimize data-driven performance, it does not fully resolve conflicts between objectives. Mechanistic bounds offer limited improvements, whereas dynamic scheduling provides a principled and effective way to navigate this trade-off.

In applications where mechanistic plausibility of the full flux distribution is critical, dynamic optimization strategies such as *MINN-scheduler* represent a robust solution, enabling controlled transitions between mechanistic alignment and data-driven re-

finement during training.

Finally, we evaluate a purely data-driven variant of the model, referred to as *MINN-divided loss*, in which the mechanistic component is entirely removed ($L_{\text{MINN}} = L_1$). As shown in the third part of Table 2.4, this configuration yields only marginal improvements in predictive performance, while causing a dramatic increase in the L_2 loss by several orders of magnitude.

This result confirms that the regularization effect responsible for producing biologically coherent flux distributions arises exclusively from the mechanistic layer. While a purely data-driven loss may suffice for pointwise flux prediction, explicit mechanistic constraints are essential to recover meaningful and qualitatively consistent flux distributions.

2.5.5 Conclusion

This work builds upon the hybrid neural–mechanistic framework introduced by [51], which integrates a genome-scale metabolic model within a NN to predict *E. coli* growth rates. We extend this approach by formulating a MINN that incorporates multi-omics data and addresses a more demanding prediction task: estimating intracellular metabolic fluxes across multiple *E. coli* single-gene knockout strains grown under minimal glucose conditions.

Our results show that the MINN achieves predictive performance comparable to, or better than, traditional ML approaches, while benefiting from a regularizing effect induced by mechanistic constraints. Beyond performance, we systematically analyzed how different architectural components and training choices influence both prediction accuracy and consistency with flux balance assumptions.

A central contribution of this study is the analysis of optimization conflicts that arise when data-driven objectives and mechanistic constraints are not fully aligned. By explicitly framing this setting as a multi-objective optimization problem, we investigated how different hybrid optimization strategies affect the trade-off between predictive accuracy and mechanistic fidelity. The proposed strategies provide practical mechanisms to control this balance during training, rather than relying on fixed or implicit assumptions.

For multi-omics integration, we adopted an early concatenation strategy. While this choice enabled a clear assessment of the hybrid learning dynamics, more advanced integration schemes may further improve predictive performance, as suggested in recent work [24]. Future extensions of the MINN framework could therefore explore mixed or hierarchical integration strategies.

Assessing the interpretability and biological plausibility of predicted flux distributions remains an open challenge. As an initial step, we monitored the L_2 distance to flux balance constraints as a proxy for mechanistic consistency, providing a quantita-

tive measure of how closely predictions adhere to FBA assumptions.

The framework was evaluated exclusively on *E. coli*, a well-studied model organism. Extending this approach to other microorganisms and to scenarios involving more complex or secondary metabolic pathways represents an important direction for future work, where the complementary role of mechanistic knowledge may become even more pronounced.

Overall, this work provides practical guidance for selecting MINN configurations based on modeling objectives. When predictive accuracy is the primary goal, the standard MINN configuration is sufficient. When improved mechanistic consistency is required, the hybrid optimization strategies proposed here can be used to reduce constraint violations while preserving good predictive performance.

2.6 Discussion and concluding remarks

The contributions presented here focus on methodological aspects of model training and evaluation, including loss formulation and optimization strategies, and show how these choices critically shape the reliability and interpretability of DL results in high-dimensional, small-sample omics settings.

The first study highlighted how widely adopted practices such as k-fold CV and ES can interact in unintended ways when applied to NNs. By systematically reviewing existing literature and common implementations, we showed that inconsistent integration of these techniques can introduce bias, data leakage, or overly optimistic performance estimates. The proposed evaluation framework provides a practical guideline to mitigate these risks and establishes a clearer methodological baseline for future studies.

The second study explored methodological issues that emerge in hybrid and mechanism-informed NN models, where data-driven objectives coexist with mechanistic constraints. By framing these models as multi-objective learning systems, we analyzed how conflicts between objectives influence training dynamics, predictive performance, and adherence to mechanistic consistency. The proposed optimization strategies illustrate how explicit control of loss interactions can be used to navigate trade-offs between accuracy and constraint satisfaction.

Taken together, the two studies demonstrate that methodological rigor in both evaluation and optimization is central to the successful application of ML in omics research. Evaluation bias and objective conflict are distinct but related manifestations of a common issue: the interaction between multiple components of the learning pipeline. Addressing these interactions explicitly is essential for building models that are not only accurate, but also reliable, interpretable, and comparable across studies.

The methodological insights developed in this chapter provide a foundation for the applied analyses presented in the following chapters, where these principles are

leveraged to study real-world foodomics problems and ML workflows in more complex experimental settings.

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

II/1. Systematic analysis of how k-fold CV and ES are combined in NN models applied to omics data, including identification of recurrent methodological pitfalls such as data leakage and biased performance estimation.

II/2. Proposal of a practical and reproducible evaluation framework for integrating k-fold CV and ES in small-sample, high-dimensional omics settings.

II/3. Development and analysis of multi-objective optimization strategies for mechanism-informed NN models, with a focus on balancing data-driven accuracy and mechanistic consistency.

II/4. Design and implementation of dynamic loss balancing strategies in a MINN to control trade-offs between competing optimization objectives.

Chapter 3

Applied Machine Learning, Explainability, and Workflow Engineering in Foodomics

This chapter focuses on the application of ML methods to omics data, with particular attention to model performance, robustness, and experimental practice. It addresses two related challenges: the development of explainable DL models for wheat SNP data, and the management of complex experimental workflows arising in applied ML research. Through two complementary studies, the chapter illustrates how applied modeling choices and structured experimentation jointly contribute to reliable and interpretable omics data analysis.

3.1 Introduction

Applied ML models for omics data analysis operate in settings characterized by high-dimensional input spaces, limited sample sizes, and heterogeneous sources of variability. In these conditions, the practical application of DL methods raises challenges that go beyond model architecture design, and concern both the robustness of learned models and the reliability of the experimental process used to develop them.

From an applied perspective, two aspects are particularly critical. The first concerns the use of DL models for genotype-to-phenotype prediction, where achieving strong predictive performance must be balanced with stability across experimental settings and interpretability of model behavior. In omics-based applications, such as foodomics and crop science, models are often evaluated on limited data, making them sensitive to training choices, regularization strategies, and sources of noise in the input data.

The second aspect concerns the organization and management of applied ML experiments. DL studies in omics typically involve complex workflows, including mul-

tiple preprocessing variants, model configurations, HPO procedures, and evaluation protocols. When these elements are handled in an ad hoc manner, it becomes difficult to compare results systematically, trace modeling decisions, or build upon previous experiments in a transparent way.

From a research practice perspective, these challenges motivate the adoption of structured approaches to experimentation that support systematic exploration, comparison, and analysis of applied ML models. Rather than focusing on deployment, such approaches aim to improve how experiments are conducted during the research phase, enabling clearer insights into model behavior and design choices.

The contributions presented in this chapter address these applied challenges through two complementary studies. The first presents an applied DL analysis of wheat SNP data, showing that the integration of modern training practices, robustness-oriented strategies, and explainability methods leads to consistent performance improvements over previously reported results on the same dataset. The second investigates MLOps frameworks as a means to structure and support experimental workflows in scientific ML.

3.2 Background

Applied ML research in foodomics and related omics domains is shaped by practical constraints that differ from those typically encountered in benchmark-driven DL studies. These constraints include limited data availability, high-dimensional feature spaces, challenges related to model robustness and interpretability, and the need to manage increasingly complex experimental workflows.

The following sections provide background on the use of DL models in applied omics problems, discuss strategies to improve model robustness through training and optimization choices, introduce explainability considerations, and outline practical issues related to experiment management. Together, these aspects motivate the applied studies and technical contributions presented in this chapter.

3.2.1 Deep Learning for Applied Omics Problems

DL models are increasingly applied to omics data in applied domains such as foodomics, agronomy, and crop science. Compared to traditional ML approaches, NNs can model complex non-linear relationships between a large number of molecular features and phenotypic traits, making them suitable for genotype-to-phenotype prediction and related tasks.

However, applied omics problems differ substantially from standard DL benchmarks. Datasets often contain tens of thousands of input features, such as SNP markers, while the number of samples is limited. In addition, the data are noisy

and heterogeneous, and measurements may be affected by technical variability and uncertainty. These characteristics make DL models prone to overfitting and unstable performance if not carefully designed and trained.

For this reason, applied omics studies frequently rely on simple NN architectures, such as MLPs, and focus mainly on predictive accuracy. In many cases, less emphasis is placed on systematic training protocols and on assessing how models behave across different data splits or experimental conditions. Strengthening these aspects is crucial for improving the robustness and and reliability of DL models in applied omics research.

3.2.2 Model Robustness Through Data Augmentation and Hyper-Parameter Optimization

Improving the robustness of DL models is particularly important in applied omics studies, where limited data amplify the risk of overfitting. Two complementary strategies to address this issue are data augmentation and systematic HPO.

While data augmentation is widely used in domains such as computer vision and natural language processing, it remains largely unexplored in omics-based DL. In this context, augmentation cannot rely on semantic transformations, but it can be designed to reflect realistic sources of noise in biological measurements, such as missing values, genotyping errors, or uncertainty introduced during imputation. Introducing controlled perturbations to the input features can encourage models to learn more stable representations and reduce sensitivity to individual features.

In parallel, the choice of hyper-parameters plays a critical role in determining model performance and generalization. In many applied omics studies, hyper-parameters are selected manually or through simple grid search, which may be inefficient and incomplete. Bayesian HPO [105] offers a principled alternative by modeling the relationship between hyper-parameters and model performance, allowing the search process to focus on promising regions of the configuration space. Integrating these strategies offers a systematic way to improve model robustness while making efficient use of limited computational and data resources.

3.2.3 Explainable Deep Learning for Omics Applications

Despite their predictive power, DL models are often criticized for their lack of interpretability. This limitation is particularly relevant in omics applications, where understanding which molecular features drive model predictions is essential for biological insight and downstream analysis.

Explainable Artificial Intelligence (XAI) methods aim to address this issue by providing post hoc interpretations of model behavior. In the context of omics data,

explainability techniques can be used to identify influential features, such as SNP markers, and to assess whether model predictions rely on biologically plausible patterns rather than spurious correlations.

Although explainability has gained attention in other application domains, it is still rarely integrated into DL studies in foodomics. Most existing works focus exclusively on predictive performance, without analyzing how models arrive at their predictions. Incorporating explainability methods into applied omics studies can therefore improve transparency, support model validation, and help bridge the gap between predictive modeling and domain-specific interpretation.

3.2.4 Experiment Management in Applied Omics Studies

Applied DL studies in omics often involve complex experimental workflows, including multiple preprocessing steps, model configurations, training runs, and evaluation protocols. As the number of experiments increases, managing these components using ad hoc scripts becomes difficult and error-prone.

Poor experiment organization can hinder systematic comparison between models, obscure the impact of individual design choices, and limit the reuse of results in follow-up studies. These challenges affect not only reproducibility, but also the efficiency and clarity of applied ML research.

Structured approaches to experiment management play an important role in enabling reliable and extensible research workflows, particularly in settings where extensive model exploration is required. These considerations align with emerging MLOps practices aimed at supporting systematic experimentation in scientific ML.

3.3 Related Works

Prior work relevant to this chapter spans two main areas: the application of DL models to omics-based prediction tasks, with particular attention to robustness and interpretability, and the organization of experimental workflows in scientific ML through MLOps practices.

3.3.1 Applied Deep Learning for Omics and Genotype-to-Phenotype Prediction

DL models have been increasingly adopted for omics-based prediction tasks, including genotype-to-phenotype modeling in agriculture and crop science. In this context, artificial NNs, CNN, and related architectures have been applied to predict agronomic traits from high-dimensional molecular inputs such as SNP markers [102, 125, 126].

These approaches demonstrate the potential of DL to capture complex non-linear relationships that are difficult to model using traditional statistical techniques.

Despite these advances, applied omics studies commonly operate under severe data constraints, where the number of features greatly exceeds the number of available samples. As a result, DL models are particularly sensitive to overfitting and performance instability across data splits. While standard regularization techniques such as dropout, weight decay, and learning rate scheduling are well established in the DL literature, their systematic adoption in foodomics and agronomic applications remains limited [4, 53, 82, 116]. Hyper-parameter selection is often performed using manual tuning or simple grid search strategies, despite the availability of more principled optimization methods [22, 105].

In parallel, explainability has emerged as an important consideration in applied ML, particularly in domains where model outputs must be interpreted and validated by domain experts. XAI methods, such as feature attribution techniques, have been applied in selected agricultural and plant science studies to identify relevant genomic regions or assess model behavior [106, 145]. However, the integration of explainability into DL-based omics studies remains inconsistent, and many works continue to focus primarily on predictive accuracy without analyzing the mechanisms underlying model predictions [47, 109, 126].

Together, these limitations highlight the need for applied DL studies in omics that combine careful training and optimization strategies with systematic robustness analysis and interpretability considerations.

3.3.2 Experiment Management and MLOps in Scientific Machine Learning

As ML models and experimental pipelines grow in complexity, the management of experiments has become an increasingly important aspect of applied ML research. In scientific settings, experiments often involve multiple datasets, preprocessing variants, model architectures, and hyper-parameter configurations, making it difficult to track results and compare design choices when workflows are managed in an ad hoc manner.

MLOps frameworks have been proposed to address these challenges by supporting experiment tracking, dataset versioning, pipeline orchestration, and result visualization [45, 96, 148]. While these frameworks are widely discussed in the context of industrial deployment and production systems, their adoption in research-oriented applications remains limited. In agronomy and foodomics, most published ML studies do not report the use of established MLOps tools such as MLflow [41], DVC [77], ClearML [34], Weights & Biases [21], even when experiments involve extensive model exploration [122, 126].

Recent work has begun to argue that MLOps practices can be beneficial even before deployment, by improving transparency, reproducibility, and experimental insight during the research phase [18]. In this perspective, MLOps frameworks are not viewed primarily as production infrastructure, but as tools to support systematic experimentation and informed model design. However, empirical studies showing how these frameworks can be integrated into applied omics workflows remain scarce, which limits our understanding of their benefits in scientific ML and motivates further investigation.

3.4 Explainable Deep Learning for Wheat SNP Prediction: Robust Training, Evaluation, and Feature Interpretation

3.4.1 Introduction

DL models have been increasingly explored for omics-based prediction tasks, including genotype-to-phenotype modeling in crop science and foodomics. These applications involve high-dimensional molecular inputs, such as SNP markers, combined with limited numbers of samples and heterogeneous sources of noise. In this setting, predictive performance is highly sensitive to training strategies, regularization choices, and data variability, making robust model design a central challenge for applied ML in omics.

Within the broader omics landscape, foodomics [26] has emerged as a domain that studies food and nutrition through integrated omics data and advanced analytical techniques. While artificial NNs have been successfully applied in various biomedical contexts, their adoption in foodomics and related agronomic applications remains comparatively limited. Existing studies have explored the use of NNs for tasks such as food quality assessment, chromatographic analysis, and microbial identification, as well as for genomic prediction in crop breeding [42, 53, 102, 125, 141]. These works demonstrate the potential of DL models to capture complex non-linear relationships in omics data, but they also highlight the challenges associated with low-sample, high-dimensional settings.

A common characteristic of many existing foodomics and agronomic DL studies is the reliance on relatively simple training pipelines, with primary emphasis placed on predictive accuracy. Standard techniques from the DL literature-such as dropout, batch normalization, learning rate scheduling, and systematic HPO-are often absent or only partially explored [53, 82, 116]. This is notable because such methods are particularly relevant in omics applications, where models are prone to overfitting and performance instability due to data scarcity.

Similarly, data augmentation strategies, which are widely used in other domains to improve robustness and generalization, have received little attention in SNP-based DL studies. Although augmentation in omics cannot rely on semantic transformations as in computer vision or natural language processing, controlled perturbations that reflect realistic sources of measurement noise or uncertainty may still provide a useful mechanism to stabilize model training. The limited exploration of these approaches suggests that the potential of modern DL practices has not yet been fully leveraged in foodomics.

Beyond predictive performance, interpretability represents a critical requirement for omics-based modeling. Understanding which molecular features drive model predictions is essential for model validation and for generating biologically meaningful hypotheses. While XAI methods have been applied in selected agricultural and plant science studies, their integration into DL models for foodomics remains inconsistent [106, 145]. Many NN-based studies report aggregate error metrics without analyzing the contribution of individual SNP markers, limiting insight into model behavior.

In this work, we address these limitations through an applied DL study on wheat SNP data. By combining modern training practices with data augmentation, systematic HPO, and post hoc explainability methods, we demonstrate that NN models can achieve consistent and statistically significant performance improvements over previously reported results on a publicly available wheat SNP dataset. In addition to improved predictive accuracy, the proposed approach provides feature-level insights into model predictions, supporting a more transparent and informed application of DL to SNP-based trait prediction in foodomics.

Objectives and Contributions

This study advances the application of NNs to SNP data analysis in foodomics by addressing both predictive performance and interpretability under realistic data constraints. The main contributions are as follows:

- **Statistically significant performance improvements**: The proposed NN model establishes new reference results on the wheat SNP dataset introduced in [126], consistently outperforming the previously reported NN and strong traditional baselines, including Random Forest, XGBoost, LASSO, and Ridge regression.

- **Adoption of modern deep learning practices**: The study applies training strategies that are standard in DL but largely absent from prior foodomics work, including model-level regularization (dropout, batch normalization), training control (learning rate scheduling), data-level augmentation, and Bayesian HPO [105].

- **Explainability of NN predictions**: Model interpretability is enhanced using SHapley Additive exPlanations (SHAP) [90] to identify influential SNP markers, enabling analysis of feature relevance and supporting biological interpretation of model predictions.

- **Support for reproducible experimentation**: To facilitate reuse and extension of the results, the study adopts reproducible research practices, including code and data sharing and controlled experimental configurations.

3.4.2 Methods

Dataset, Preprocessing, and Evaluation Scheme

Dataset Cleaning and Preprocessing The dataset used in this study is a nested association mapping (NAM) population described in [126]. It consists of genotypic and phenotypic data for 650 recombinant inbred lines from 26 NAM families, provided by Kansas State University. These lines were evaluated for five agronomic traits: grain yield, grain protein content, heading date, plant height, and test weight, measured in three growing seasons in 2014, 2015, and 2016.

In preprocessing of the dataset we mostly adhered to the preprocessing steps from [126], with the exception that we moved the scaling of the features and targets between 0 and 1 after splitting the dataset into training, validation, and testing sets. This ensures the scaler module is fitted only on the training data, as shown in the schema in Figure 3.1. For the data cleaning, we removed RILs with missing phenotypic information and filtered SNP markers with more than 20% missing data, a minor allele frequency of < 0.10, and RILs with $> 10\%$ missing genotypic data.

Evaluation Schema To assess the robustness of the model, we used 10 disjoint data splits via k-fold CV, shown on the left side of Figure 3.2. The models were trained on their respective training folds, which were further divided into training and validation sets for ES purposes when training NN models (as shown in Figure 3.1), and evaluated on the test fold acting as test data split for that iteration. The metrics from these independent evaluations were then aggregated to compute the final metrics of the k-fold CV and their standard deviations.

Furthermore, to have robust model selection and HPO, we employed a nested 10-fold CV on the training folds of the first iteration of the outer k-fold CV, shown on the right side of Figure 3.2. Although extending this procedure to all the outer k-fold CV iterations would give more accurate evaluations, it would be much more computationally demanding and prohibitive for the HPO procedure. In this context, the scaling between 0 and 1 of features and targets was performed after nested k-fold splitting and subsequent splitting of training data into training and validation

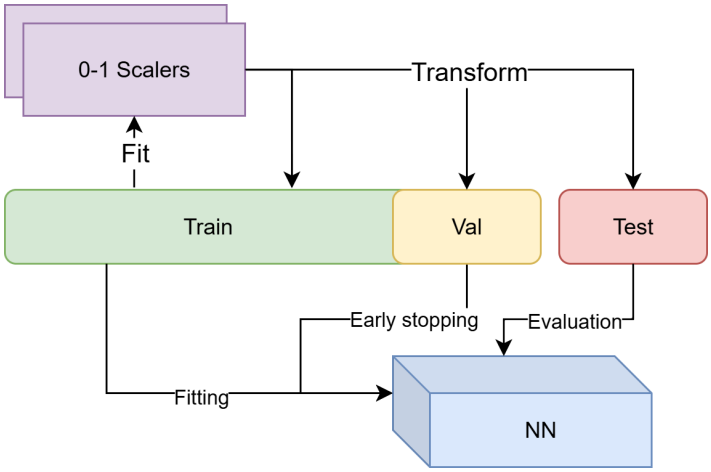


Figure 3.1: *Training pipeline after k-fold CV splitting. The training fold is further divided into training and validation subsets for ES, and feature scaling is fitted only on the training data to avoid leakage. The final model is evaluated on the test fold.*

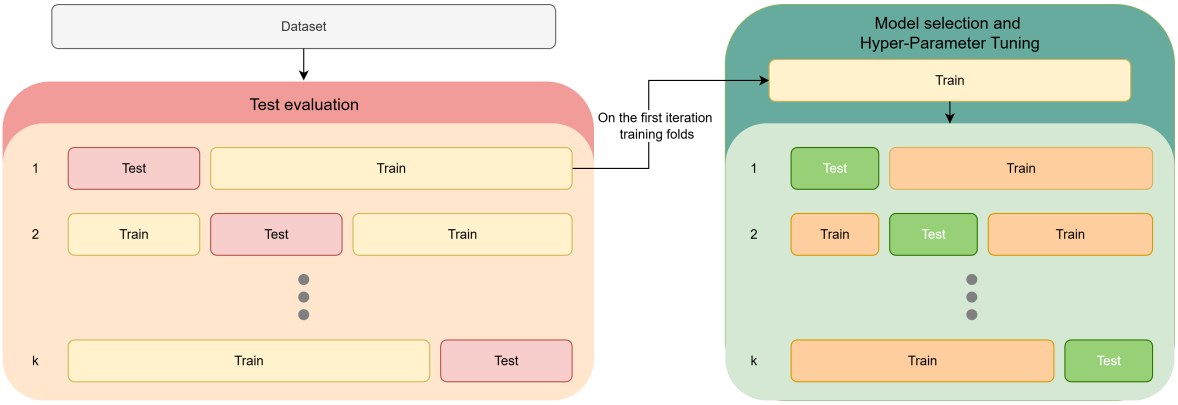


Figure 3.2: *Evaluation schema. The outer 10-fold CV estimates generalization performance, while a nested 10-fold CV on the training data is used for model selection and HPO.*

sets, adhering to the schema in Figure 3.1. Since no clear multimodal patterns were observed in the agronomic traits distributions, we did not apply stratification when performing the k-fold CV.

The dataset comprises five agronomic traits, each measured in three growing seasons; therefore, the models were trained and evaluated independently on each trait and growing season, resulting in a total of 15 model evaluations for each model.

Neural Network Model Design

We designed an MLP model to link SNP data with phenotypic traits, introducing several mechanisms that, while common in DL, to our knowledge, are novel in the foodomics field.

We implemented a learning rate scheduler to adaptively reduce the learning rate when the validation loss plateaued, facilitating better convergence toward local minima. Additionally, we introduced dropout within the NN architecture to improve generalization. We further explored the use of batch normalization and applied two forms of data augmentation: (1) flipping a portion of each sample's features independently (switching 0s to 1s and vice versa) and (2) suppressing a subset of features by assigning them a value of 0, regardless of their original state. These are not meant to simulate real genetic mutations, but rather the type of noise you can find in SNP data. Flipping features can be seen as small genotyping errors or allele miscalls, while masking mimics missing values or dropout during sequencing. In this way, the model learns to handle the same kind of uncertainty that also happens during genotype imputation, especially with rare variants or lower-density data, where accuracy is known to be lower [72, 94].

Moreover, to optimize hyper-parameters efficiently, we employed Bayesian optimization [105], which leverages a probabilistic model to guide the search process, enabling an effective exploration of the hyper-parameter space.

Baseline Models

As points of comparison, we implemented four baseline methods commonly adopted in the omics literature [63, 86, 87, 134]: RF, XGBoost, LASSO, and ridge regression. These baselines range from tree-based ensemble learners to linear penalized regressors, and are often used as reference models in genomic prediction.

To ensure compatibility across the methods, all baseline models were trained following the same evaluation scheme as the NN models (Section 3.4.2), with the exception that the training folds were not further split into training and validation subsets, since no ES mechanism was required (Figure 3.1). Moreover, HPO was carried out analogously to the NN experiments: an inner k-fold CV was performed on

the 2016 data from the first outer CV iteration, and the best configuration identified for each trait was subsequently applied to all other folds and growing seasons.

Implementation of Prior Work

To compare the results of our model with those from [126], we used the implementation made available by the authors. We initialized the MLP model with the best hyper-parameters reported in the study for the respective traits, and we trained and evaluated the models with the evaluation schema detailed in Section 3.4.2, to have the results on the same data splits.

Moreover, to enable a direct comparison with the study from [126], we also report the correlation between model predictions and target values (which was initially reported in [126]). Although we recommend considering the RMSE metric for model evaluation as, contrary to correlation, it is sensitive to shifts and dispersion, providing a more accurate assessment of the performance of the regression models. [69].

Statistical Tests

To assess the statistical significance of the performance differences between our model, the NN from [126], and the baseline models introduced in Section 3.4.2, we applied the paired Wilcoxon signed-rank test [149].

This non-parametric test evaluates whether the median of the RMSE differences between two models deviates significantly from zero, without requiring distributional assumptions. The paired t-test, a common parametric alternative, is not appropriate here because k-fold CV evaluations are not independent [43]. To compute the test statistic, we aggregated the k-fold evaluations per trait, yielding 30 evaluations per model per trait, and then calculated the pairwise differences in RMSE between our model and each alternative. For each model–trait comparison, we thus obtained both the median RMSE difference (negative values indicating superior performance of our model) and the associated p-value from the Wilcoxon test.

Finally, to account for the multiple comparisons performed, we controlled the family-wise type I error rate using the Holm–Bonferroni correction [61], and we report significance based on the adjusted p-values.

Explainability Tools

One of the challenges with NN models is their black-box nature, which makes it difficult to understand how they arrive at their predictions. This lack of transparency can make the models less reliable for experts in the field. To enable the interpretation of the prediction of the model, we integrated SHAP [90] in our study. This method

provides insights into the contribution of each feature to the model's predictions, allowing us to determine the most influential features (mutations).

By calculating SHAP values for each SNP in the dataset, we identified which genetic variants had the most significant impact on the model output. This enhanced the transparency of our model by showing which SNPs were driving the predictions for specific agronomic traits, enabling the validation of the model's understanding and thus increasing trust in its predictions.

In addition, highlighting the most critical SNPs in predicting key agronomic traits offers potential targets for genetic analysis in crop science. These SNPs can be subsequently investigated through targeted genetic analyses to validate their functional relevance, as demonstrated in studies linking SNPs to meiotic function through recombination rate mapping [72].

Reproducibility

To ensure the reproducibility of our implementation, we took several measures:

- Seeding the Code: We set random seeds throughout the code to ensure consistent results across different runs.
- Docker Containers: We used Docker to create a virtualized environment, making our results repeatable and reproducible across different machines.
- ClearML Framework[1]: We integrated ClearML, an open source framework, to introduce MLOps practices; in this way enhancing reproducibility by tracking git commits and managing datasets versioning.
- Dataset Publicly Available: We made the preprocessed dataset available in its ready-to-use format for model training to facilitate future research.

These measures ensure that our results can easily be replicated by other researchers, facilitating further advances in the field.

3.4.3 Results and Discussion

Hyper-Parameter Optimization

Data Splitting for Validation The assessment of model performance for HPO was conducted using a nested k-fold CV, as detailed in Section 3.4.2 and shown in Figure 3.2. Specifically, the outer k-fold CV was used for the final assessment of the model on unseen data, while the train folds from its first iteration were used for the inner

[1]ClearML Team, "ClearML: Open-source MLOps framework." Available: `https://github.com/clearml/clearml`, 2024.

10-fold CV to fine-tune the hyper-parameters. For each phenotypic trait, the HPO was performed once on the target of the growing season of 2016, and the best hyper-parameters were then used when working on the other years for that trait.

Search Space for Hyper-Parameters The following search space was used for all HPOs across the different phenotypic traits:

- **Masking Augmentation:** [0.1, no augmentation]
- **Noising Augmentation:** [0.1, no augmentation]
- **Number of Layers:** 1 to 5
- **First Layer Dimension:** 64 to 512, in steps of 32
- **Dropout Probability:** 0.0 to 0.6
- **Batch Normalization:** [True, False]
- **Weight Decay:** 0.00001 to 0.001 (log-uniform)
- **Optimizer:** ["sgd", "adam"]
- **Learning Rate:** 0.000001 to 0.05 (log-uniform)
- **Reduce Learning Rate on Plateau Scheduler:** [True, False]

To have only two hyper-parameters defining the width of the whole NN model, the dimensions of the hidden layers are calculated based on the number of layers and the first layer's dimension. The process involves making the last hidden layer one-fourth of the first and setting the other hidden layers' size equidistantly; so, for example, in an NN with four hidden layers and the first layer of size 128, the last hidden layer will count 32 neurons, and the three hidden layers will be 104, 80 and 56. The final layer is then added after the last hidden layer to output a single value for regression.

Distribution of Experiments The distribution of the trials and their outcomes, shown in Figure 3.3, demonstrates the effectiveness of the Bayesian strategy for HPO. Most of the experiments are concentrated in the lower part of the plot, indicating lower inner k-fold loss values. This suggests that the Bayesian optimization efficiently focused on the best combinations of hyper-parameters, achieving better results in less time. Moreover, from the few trials with higher validation loss, we can deduce that the strategy effectively explored the search space.

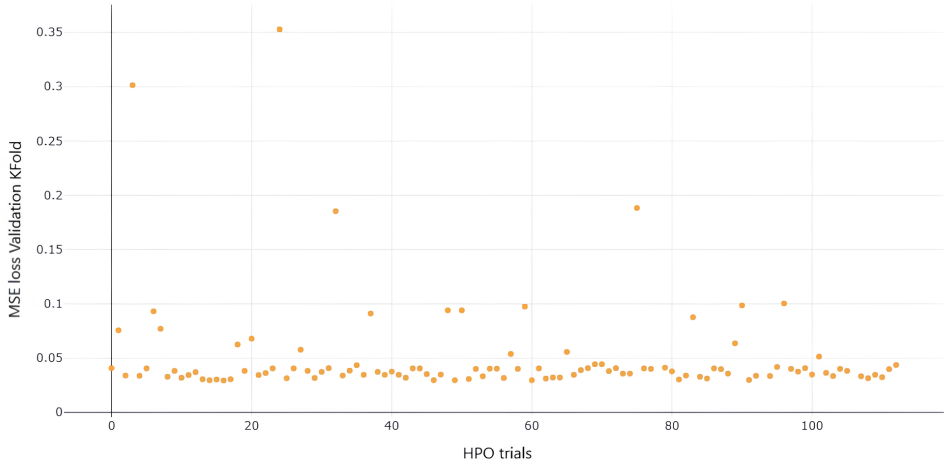


Figure 3.3: *HPO trials and their loss values from the experiments on the yield trait. The y-axis was limited to 0.4 to facilitate interpretation, excluding three trials with extremely high loss values.*

Analysis of Hyper-Parameter Importance The parallel coordinates plot in Figure 3.4 shows the influence of different hyper-parameters on the inner k-fold MSE loss for the yield trait. It does not appear that any single hyper-parameter consistently leads to better results. Instead, complex combinations of several hyper-parameters likely contributed to the improvement in performance.

Best Hyper-Parameter Sets The optimal hyper-parameter configurations for each trait are detailed in Table 3.1. The use of Masking Augmentation for traits such as Grain Yield, Plant Height, and Days to Heading suggests its efficacy in enhancing model generalization. The application of Dropout is particularly high for Days to Heading, indicating the need to reduce overfitting. It is interesting to observe that Batch Normalization was uniformly omitted, which might suggest its unsuitability for this dataset, probably due to the discrete nature of the data or small batch size. Instead, the First Layer Dimension shows significant variations across the different traits, especially the larger size used for Days to Heading, suggesting the need for more model capacity for specific traits (the hidden layers dimensions were derived as detailed in Section 3.4.3). The choice of optimizer was more consistent, with Adam only employed for Plant Height.

Results

In Table 3.2 and Figure 3.5 are the results in terms of RMSE of our NN, the one from [126], and the baseline models.

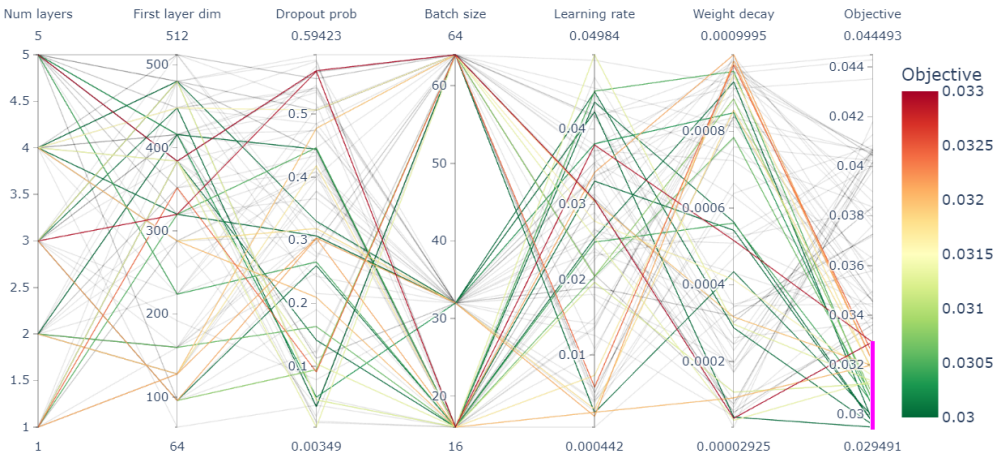


Figure 3.4: *Parallel coordinates chart of the best-performing experiments in the HPO, showing the influence of different hyper-parameters on the inner k-fold MSE loss. To aid interpretation, trials with a loss value below 0.033 are highlighted.*

A first observation is the clear separation between DL and traditional ML approaches. Across all traits, the ML baselines (RF, XGB, LASSO, ridge) perform at least an order of magnitude worse than the NN-based models, underlining their limitations in handling high-dimensional SNP data. Even for Grain Yield, where the baselines obtain their best results compared to the other traits, performance remains substantially below that of both NN implementations.

In addition, for Plant Height and Days to Heading, the ML baselines exhibit extremely wide bootstrap confidence intervals (Figure 3.5), indicating unstable generalization and poor robustness across folds and years. By contrast, the NN models yield much narrower intervals, showing more reliable predictive behavior.

The proposed NN achieved lower errors than the model from [126] on almost all targets, with an average RMSE improvement of 0.017 from the NN of [126] (corresponding to a 10.5% gain). When comparing the two NN approaches directly, shown in Figure 3.6a, our model shows consistent improvements, expecially for the protein content, test weight, and plant height traits where the RMSE reductions ranges between 11–19% while for the grain yield and days to heading the there is a performance gain of around 4.5%. Moreover the bootstrap confidence intervals suggest that the results gap between the models isn't much influenced by variations in performance, as we will allso see in the next Section 3.4.3.

Although we recommend considering the RMSE metric for model evaluation for the reasons explained in Section 3.4.2. We also report in Figure 3.6b and Table 3.3 the correlation between model predictions and target values (which was initially reported in [126]).

Hyper-parameter	Grain Yield	Grain Protein Content	Test Weight	Plant Height	Days to Heading
Masking Augmentation	True	False	False	True	True
Noising Augmentation	True	False	False	False	False
Number of Layers	2	3	3	4	2
First Layer Dimension	96	320	224	96	416
Dropout Probability	0.1771	0.0488	0.0316	0.0366	0.4444
Batch Normalization	False	False	False	False	False
Weight Decay	9.56×10^{-5}	1.82×10^{-4}	9.90×10^{-4}	3.40×10^{-4}	5.64×10^{-4}
Batch Size	16	16	16	64	16
Optimizer	sgd	sgd	sgd	adam	sgd
Learning Rate	0.0485	0.0280	0.0258	0.0049	0.0436
Reduce LR on Plateau Scheduler	False	False	True	True	True

Table 3.1: *Best hyper-parameter set found for each trait. As detailed in Section 3.4.3, the hyper-parameters were selected based on the training data of the first iteration of the outer k-fold CV and used for all the other iterations.*

Trait	Year	Ours	Sandhu et al.
Grain Yield	2014	**0.436 (0.128)**	0.414 (0.134)
	2015	**0.369 (0.069)**	0.244 (0.132)
	2016	**0.505 (0.093)**	0.398 (0.196)
Grain Protein Content	2014	**0.518 (0.098)**	0.398 (0.105)
	2015	**0.489 (0.066)**	0.332 (0.138)
	2016	**0.580 (0.063)**	0.469 (0.098)
Test Weight	2014	**0.552 (0.112)**	0.264 (0.082)
	2015	**0.489 (0.114)**	0.205 (0.125)
	2016	**0.410 (0.068)**	0.210 (0.097)
Plant Height	2014	**0.631 (0.094)**	0.473 (0.084)
	2015	**0.521 (0.071)**	0.167 (0.147)
	2016	**0.589 (0.097)**	0.406 (0.191)
Days to Heading	2014	**0.490 (0.096)**	0.275 (0.174)
	2015	**0.222 (0.146)**	0.095 (0.149)
	2016	**0.426 (0.096)**	0.163 (0.131)

Table 3.3: *Comparison of our model and the model from [126] by year and trait, based on the correlation between predictions and ground truths.*

Statistical significance Table 3.4 reports the median RMSE differences between our model and each alternative, together with the corresponding Wilcoxon test results [149]. In all cases, the median differences are negative, indicating lower errors for our model. After applying the Holm–Bonferroni correction [61], all comparisons remained significant at $\alpha = 0.05$, allowing us to conclude that our model achieves significantly better median performance than all alternatives across all traits.

It is worth noting that significance was obtained even for the smallest observed differences, such as Grain Yield and Days to Heading against the NN from [126],

		RMSE (mean ± sd)					
Trait	Year	Ours	Sandhu et al.	RF	XGB	LASSO	Ridge
Grain Yield	2014	0.140 (0.025)	0.138 (0.024)	0.280 (0.023)	0.279 (0.024)	0.300 (0.020)	0.281 (0.028)
	2015	0.150 (0.025)	0.178 (0.035)	0.226 (0.033)	0.235 (0.030)	0.234 (0.036)	0.227 (0.032)
	2016	0.175 (0.021)	0.179 (0.021)	0.658 (0.070)	0.654 (0.078)	0.686 (0.075)	0.670 (0.073)
Grain Protein Content	2014	0.138 (0.025)	0.155 (0.023)	0.927 (0.077)	0.935 (0.080)	0.992 (0.081)	0.889 (0.072)
	2015	0.133 (0.020)	0.156 (0.027)	0.991 (0.133)	1.004 (0.104)	1.078 (0.130)	1.001 (0.114)
	2016	0.146 (0.012)	0.169 (0.019)	1.120 (0.093)	1.122 (0.096)	1.184 (0.107)	1.083 (0.079)
Test Weight	2014	0.126 (0.016)	0.163 (0.018)	1.676 (0.167)	1.720 (0.206)	1.731 (0.194)	1.574 (0.176)
	2015	0.120 (0.023)	0.149 (0.017)	1.649 (0.262)	1.704 (0.254)	1.709 (0.222)	1.599 (0.238)
	2016	0.133 (0.027)	0.156 (0.024)	1.252 (0.190)	1.270 (0.215)	1.296 (0.158)	1.213 (0.199)
Plant Height	2014	0.158 (0.012)	0.182 (0.014)	3.022 (0.287)	2.901 (0.260)	3.192 (0.282)	3.244 (0.192)
	2015	0.117 (0.018)	0.132 (0.017)	0.860 (0.093)	0.847 (0.088)	0.881 (0.085)	0.811 (0.094)
	2016	0.143 (0.019)	0.159 (0.017)	3.288 (0.288)	3.239 (0.331)	3.524 (0.224)	3.409 (0.326)
Days to Heading	2014	0.166 (0.024)	0.181 (0.030)	2.316 (0.249)	2.326 (0.310)	2.367 (0.250)	2.193 (0.278)
	2015	0.108 (0.096)	0.093 (0.092)	2.108 (0.713)	2.188 (0.739)	2.140 (0.686)	2.037 (0.724)
	2016	0.170 (0.021)	0.185 (0.026)	3.010 (0.364)	3.142 (0.384)	3.096 (0.375)	2.927 (0.377)

Table 3.2: *Comparison of model performance (RMSE) by trait and year. Best results in bold.*

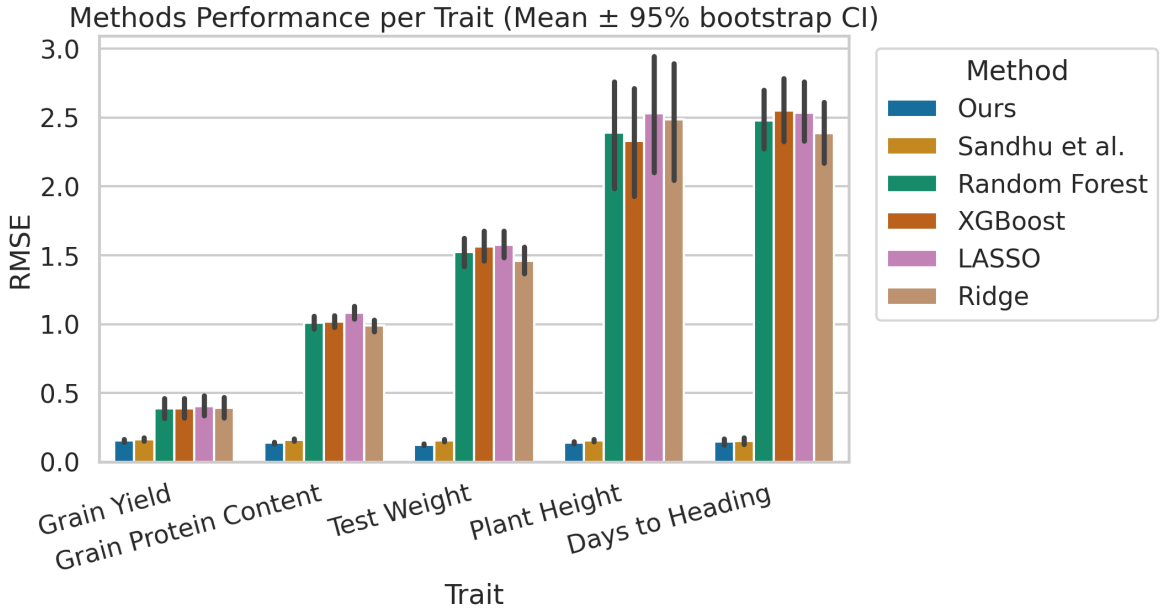


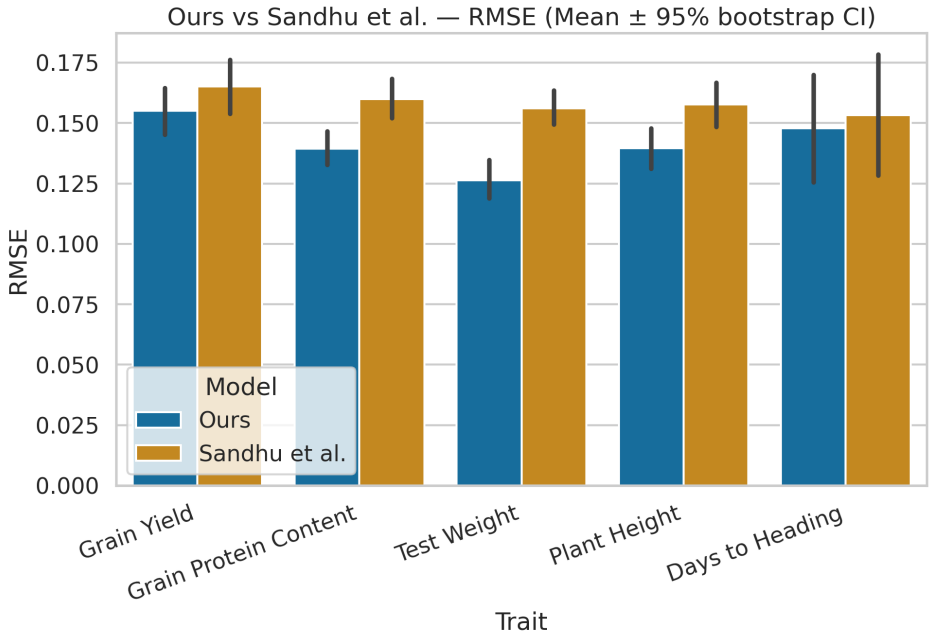
Figure 3.5: *Comparison between our model and the one from [126], and the baselines presented in Section 3.4.2, the RMSE of the years and k-fold evaluations are averaged per trait, and the lines represent the bootstrap confidence intervals. For a clearer comparison between our model and the one from [126], refer to Figure 3.6a.*

where median improvements were only a few thousandths in RMSE. The ability to detect significance in these cases reflects the stability of our improvements across folds, in contrast with the much larger gaps observed against the ML baselines, where differences reached one to three RMSE units depending on the trait. Taken together, these results provide strong evidence that the proposed enhancements deliver consistent and statistically reliable performance gains over both classical ML methods and prior NN approaches.

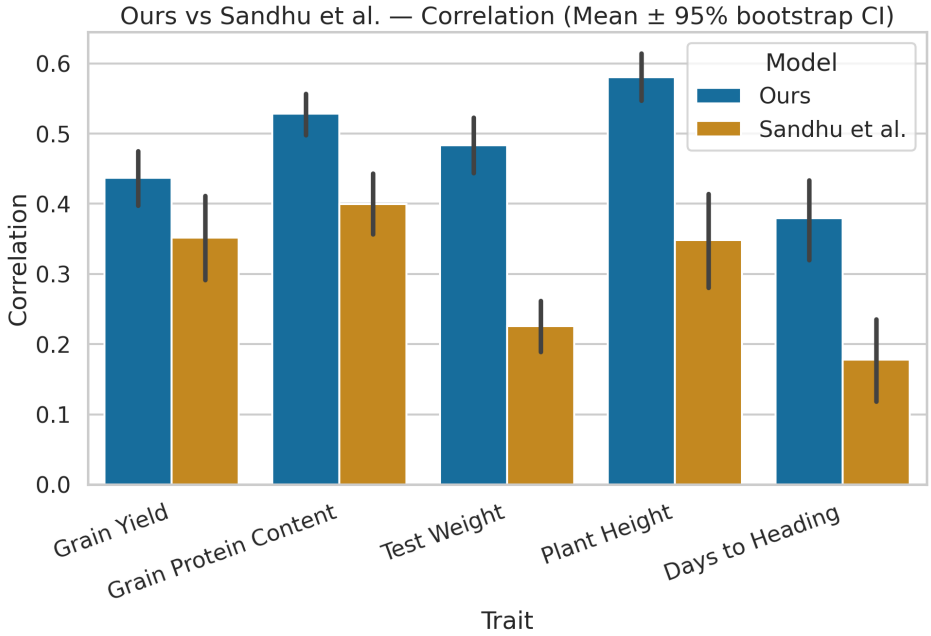
Explainability

To interpret the model's predictions, we analyzed the most important features using SHAP values. The SHAP values were calculated based on the 10 test folds of the outer k-fold CV (shown on the left of Figure 3.2) that were aggregated to extract insights from the whole dataset.

Figure 3.7a shows the top 20 features ranked by importance. The position of the dots, which are the test observations, with respect to the x-axis, indicates the strength of each feature's impact on the predictions. For instance, the first two features, SpringWheatNAM_tag_99945 and Kukri_c12738_882, with high values (homozygous mutations) lead to higher predictions of the test weight trait for 2015 while mutations on the SNP SpringWheatNAM_tag_82218 lead to a decrease in the



(a) *Comparison of the proposed NN and the model from [126] in terms of RMSE.*



(b) *Comparison of the proposed NN and the model from [126] in terms of correlation between predictions and ground truth.*

Figure 3.6: *Performance comparison between the proposed NN and the model from [126] across RMSE and correlation metrics. The metrics of the years and k-fold evaluations are averaged per trait, and the lines represent the bootstrap confidence intervals.*

Trait	Compared To	Median Diff (RMSE)	Wilcoxon p-value	Corrected p-value	Significance at 0.05
Grain Yield	Sandhu et al.	-0.005	4.97e-02	4.97e-02	Significant
	Random Forest	-0.144	1.86e-09	9.31e-09	Significant
	XGBoost	-0.141	1.86e-09	9.31e-09	Significant
	LASSO	-0.162	1.86e-09	9.31e-09	Significant
	Ridge	-0.143	1.86e-09	9.31e-09	Significant
Grain Protein Content	Sandhu et al.	-0.017	4.42e-06	4.42e-06	Significant
	Random Forest	-0.858	1.86e-09	9.31e-09	Significant
	XGBoost	-0.874	1.86e-09	9.31e-09	Significant
	LASSO	-0.916	1.86e-09	9.31e-09	Significant
	Ridge	-0.848	1.86e-09	9.31e-09	Significant
Test Weight	Sandhu et al.	-0.03	3.73e-09	9.31e-09	Significant
	Random Forest	-1.423	1.86e-09	9.31e-09	Significant
	XGBoost	-1.45	1.86e-09	9.31e-09	Significant
	LASSO	-1.434	1.86e-09	9.31e-09	Significant
	Ridge	-1.345	1.86e-09	9.31e-09	Significant
Plant Height	Sandhu et al.	-0.019	8.33e-07	8.33e-07	Significant
	Random Forest	-2.838	1.86e-09	9.31e-09	Significant
	XGBoost	-2.67	1.86e-09	9.31e-09	Significant
	LASSO	-3.089	1.86e-09	9.31e-09	Significant
	Ridge	-2.99	1.86e-09	9.31e-09	Significant
Days to Heading	Sandhu et al.	-0.006	2.19e-03	2.19e-03	Significant
	Random Forest	-2.173	1.86e-09	9.31e-09	Significant
	XGBoost	-2.326	1.86e-09	9.31e-09	Significant
	LASSO	-2.209	1.86e-09	9.31e-09	Significant
	Ridge	-2.048	1.86e-09	9.31e-09	Significant

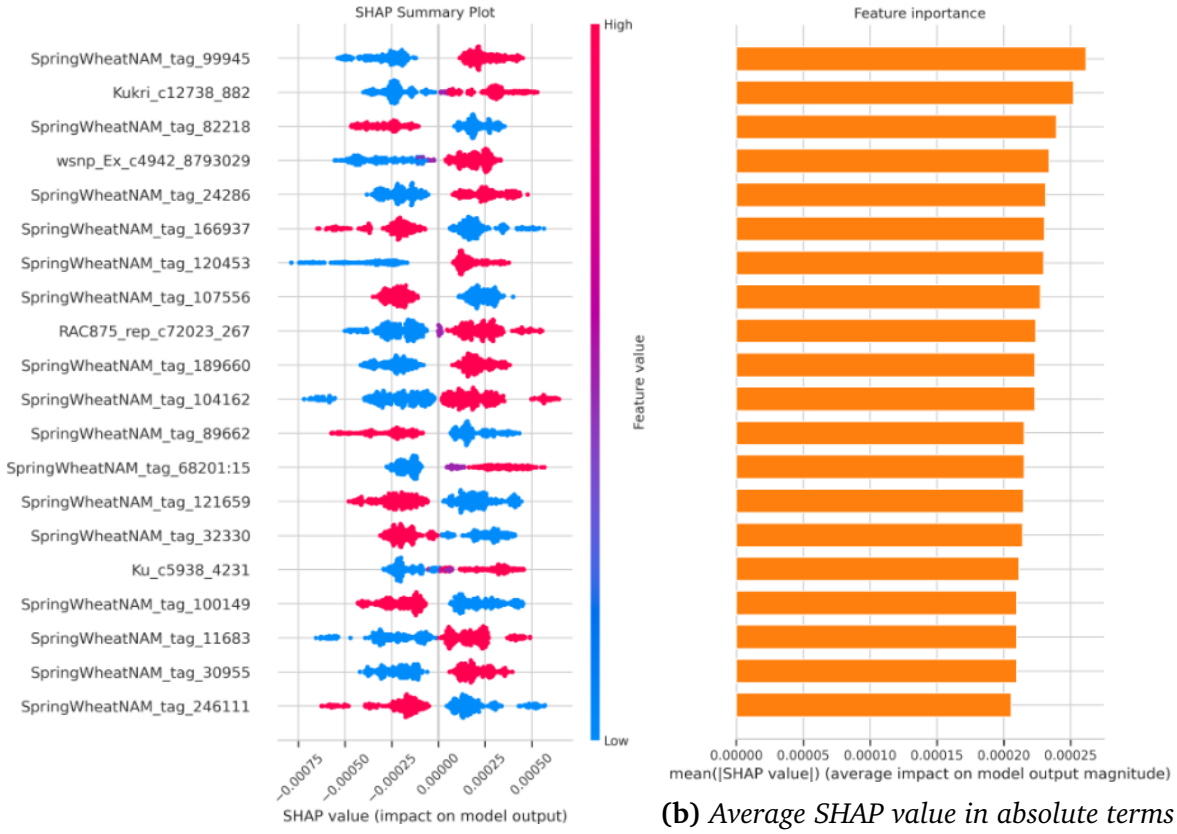
Table 3.4: *Wilcoxon signed-rank test results comparing our method to the model from [126] and the baselines in terms of median RMSE difference, raw and Holm-Bonferroni corrected p-values are reported, and the significance at $\alpha = 0.05$ of the corrected p-values.*

agronomic trait prediction, as the samples with mutations have negative SHAP values and vice-versa for the ones without mutations. Furthermore, Figure 3.7b presents the feature importance as a bar plot, showing the importance in absolute terms.

For completeness, the list of the top 40 SNP markers is reported in Table 3.5. While their functional annotation was beyond the scope of this study, these markers provide a reproducible set of candidates that can guide future genomic analyses and biological validation.

Given the high dimensionality of the dataset (approximately 44,000), understanding the distribution of feature importance is crucial. The cumulative percentage of absolute SHAP values in Figure 3.8 shows that, although the top features have the greatest impact on model predictions, a substantial number of additional features also make non-negligible contributions. Consistently, the distribution of SHAP values in Figure 3.9 indicates that while most features are of marginal importance, a subset clearly stands out with significant influence. This suggests that, although some features are pivotal for model predictions, a combination of several others might also contribute meaningfully to the model predictions.

The SHAP values also allow us to assess the model's robustness across different iterations of k-fold CV by comparing the important features for the 10 models. Re-



(a) *The dots represent test samples aggregated across different k-fold CV splits.*

(b) *Average SHAP value in absolute terms for each of the features.*

Figure 3.7: *Top 20 features and their impact on the predictions of the test weight trait for 2015. In Figure (a), the color indicates the feature value: high for homozygous mutations (red), low for no mutations (blue), and medium for heterozygous mutations (purple). The horizontal dispersion shows the impact on the model's predictions: dots further to the right indicate that the feature value pushed the prediction towards a higher test weight for that sample.*

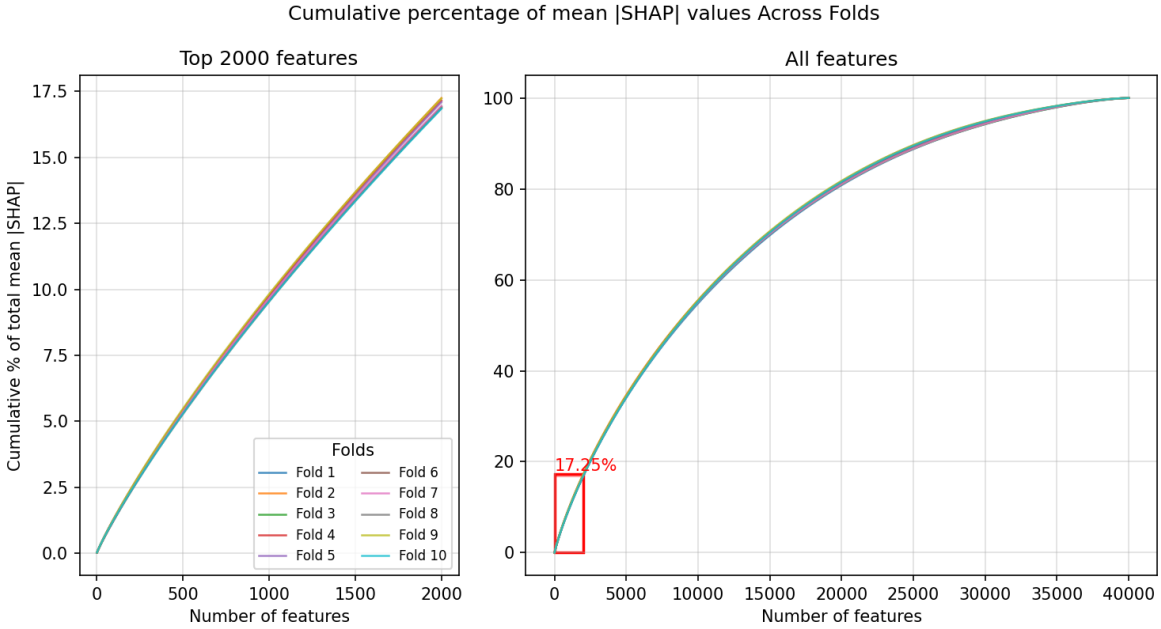


Figure 3.8: *Cumulative percentage of absolute SHAP values per feature. The left panel reports the cumulative contribution of the top 2000 features, while the right panel extends the analysis to all features in the dataset (the red rectangle represents the area shown in the left panel). Different colors correspond to the 10 folds of the CV performed on the Test Weight trait for the 2015 season.*

Feature	Feature
SpringWheatNAM_tag_99945	SpringWheatNAM_tag_46784
Kukri_c12738_882	SpringWheatNAM_tag_95802
SpringWheatNAM_tag_82218	SpringWheatNAM_tag_31634
wsnp_Ex_c4942_8793029	SpringWheatNAM_tag_102155
SpringWheatNAM_tag_24286	Ku_c5938_4221
SpringWheatNAM_tag_166937	SpringWheatNAM_tag_8006
SpringWheatNAM_tag_120453	SpringWheatNAM_tag_190470
SpringWheatNAM_tag_107556	Jagger_c3646_235
RAC875_rep_c72023_267	SpringWheatNAM_tag_80159
SpringWheatNAM_tag_189660	SpringWheatNAM_tag_185592
SpringWheatNAM_tag_104162	SpringWheatNAM_tag_33944
SpringWheatNAM_tag_89662	SpringWheatNAM_tag_63233
SpringWheatNAM_tag_68201:15	SpringWheatNAM_tag_22164:67
SpringWheatNAM_tag_121659	SpringWheatNAM_tag_80706
SpringWheatNAM_tag_32330	SpringWheatNAM_tag_69283
Ku_c5938_4231	SpringWheatNAM_tag_26764
SpringWheatNAM_tag_100149	Kukri_c40439_366
SpringWheatNAM_tag_11683	SpringWheatNAM_tag_60437
SpringWheatNAM_tag_30955	SpringWheatNAM_tag_163097
SpringWheatNAM_tag_246111	BobWhite_c4399_447

Table 3.5: *Top 40 most important SNP features for the test weight trait in 2015 (by mean absolute SHAP value).*

ferring again to Figure 3.7a, we can observe that the aggregated samples from the k-fold CV yielded similar results across different features. The dots of the same colors consistently appear on the same side of the x-axis, indicating that the 10 models, independently trained on different k-fold splits, consistently found the same direction of the interpretation for the mutations of the top features. This is a noteworthy result, given the extremely high number of features.

3.4.4 Conclusion

In this study, we proposed an explainable NN model for SNP data analysis in wheat, introducing regularization, optimization, and hyper-parameter search strategies that are largely unexplored in foodomics. With the proposed NN, incorporating data augmentation, dropout, batch normalization, learning rate scheduling, and Bayesian HPO, we advanced the use of NN in the field and established state-of-the-art results on the dataset from [126]. Our model consistently outperformed both the previously published NN and strong ML baselines across almost all traits and growing seasons,

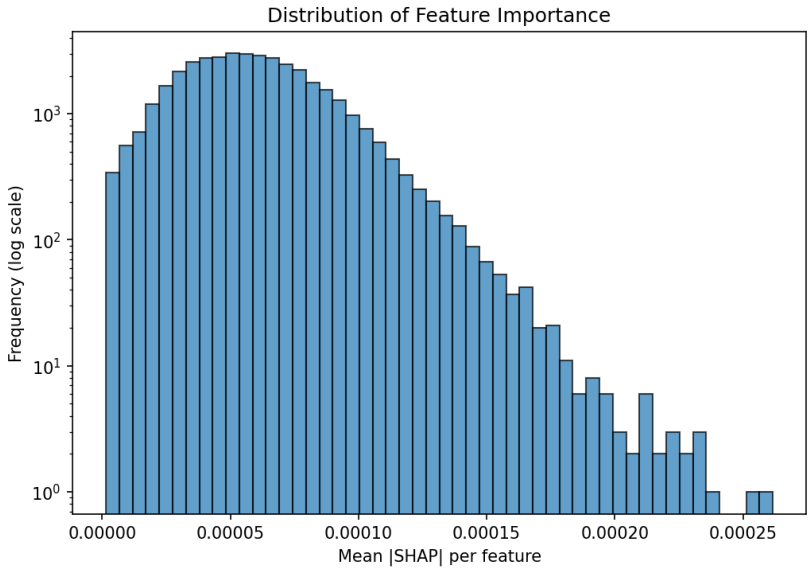


Figure 3.9: *Distribution of feature importance for the test weight trait in 2015, presented as a logarithmic histogram. The x-axis shows the mean of the absolute SHAP values per feature, and the y-axis shows the frequency on a log scale.*

with improvements that were not only consistent but also statistically significant.

Beyond predictive performance, using the SHAP method [90] for model explainability allowed the identification of the most influential SNP markers in predicting phenotypic traits. This step adds transparency, increases trust in model predictions, and provides insights that can support biological interpretation. While full functional annotation of these markers was beyond the scope of this work, the ranked list we provide offers a solid basis for future genomic studies and experimental validation.

Furthermore, with our efforts to ensure reproducibility through the integration of MLOps frameworks and the adoption of best code and data management practices, we facilitate researchers to use our work for future developments, fostering innovation in agricultural genomics.

3.5 Research-Stage MLOps for Agronomy: Structuring, Tracking, and Reproducing Machine Learning Experiments

3.5.1 Introduction

ML and DL methods have been increasingly applied in agricultural research, supporting a range of data-driven modeling tasks. Artificial NNs, in particular, have shown

promising results due to their ability to model complex relationships in heterogeneous data [16, 49, 76].

Complementing modeling approaches, MLOps refer to a set of practices and tools designed to support the management of ML experiments and workflows. Despite the ML literature stresses the importance of MLOps in ensuring the reproducibility, scalability, and operational efficiency of ML applications, the adoption of these methodologies is still limited in both research and industrial applications [45, 96, 148]. Only a limited number of studies have examined MLOps in agronomy, and most of this work has focused on industrial or production-oriented scenarios. particularly model deployment, monitoring, and operational robustness. For example, [6, 35] propose combinations of frameworks to support monitoring and deployment of ML models in agricultural settings, while [119] introduces a web-based interface to facilitate model accessibility in applied contexts. MLOps concepts have also been explored in resource-constrained settings, such as for the development of cost-efficient smart sensors [143].

Although these studies demonstrate the practical relevance of MLOps in applied agronomic systems, their focus is largely on post-development phases, and provides limited insight into how MLOps practices can support the research and experimentation stages of ML workflows. MLOps methodologies can be beneficial even before deployment, by structuring experimentation and improving traceability of modeling decisions.

By documenting experiments through version control and data versioning, MLOps practices can substantially improve reproducibility, particularly when combined with virtualization tools such as Docker [97]. In addition, these frameworks facilitate systematic logging and visualization of training dynamics and HPO outcomes, supporting more informed model design decisions and the management of complex experimental workflows.

Despite the potential MLOps practices, their adoption in the agronomic fields remains limited. In our review of nearly 80 recent agronomic studies that developed ML and DL models, with applications that spanned several objectives, including crop yield prediction [2, 8, 126], disease and stress detection [9, 99], precision agriculture [58, 137], greenhouse optimization [17, 81], automated fruit detection [15, 29, 123], and plant growth modeling [5, 158]; none of the studies mentioned any of the most widely used MLOps frameworks, MLflow [41], DVC [77], ClearML [34], Weights & Biases [21], Seldon [128], and TensorBoard [1]. The scarce adoption of MLOps practices during the research process suggests the need for standardized protocols for ML applications in agronomy and anticipates potential issues in experiment reproducibility and informed model design in the field.

In this work, we address this gap by exploring the potential of MLOps to support the research process in agronomy through a case study. Specifically, we integrate

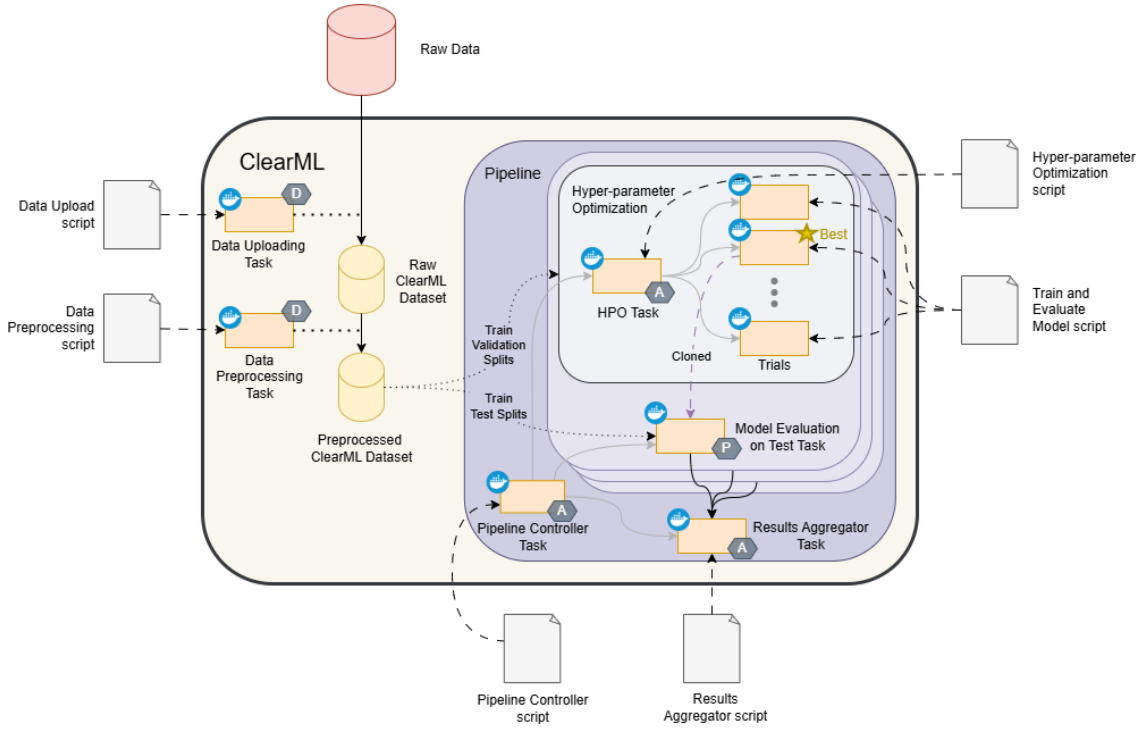


Figure 3.10: *High-level overview of integrating the ClearML MLOps framework into the study workflow. Existing scripts are wrapped as ClearML tasks and executed in Docker containers. Tasks marked* ***[D]*** *handle data preparation and versioning, tasks marked* ***[A]*** *orchestrate pipelines and hyperparameter tuning, and tasks marked* ***[P]*** *export trained models and results.*

ClearML, an open-source MLOps framework, into all stages of an ML/DL-based agricultural study. Our goal is not only to demonstrate the benefits of MLOps, but also to derive practical guidelines for how such frameworks can be integrated into research workflows. We focus on five core capabilities that MLOps frameworks are expected to support in scientific ML [18, 59]: experiment tracking, dataset management, pipeline automation, result visualization, and production readiness. By incorporating these practices across the ML life cycle, from preprocessing to workflow orchestration (Figure 3.10), we illustrate how MLOps can contribute to more reproducible, scalable, and transparent research workflows.

3.5.2 Framework and Tools

Several frameworks are available to support the integration of MLOps practices into ML projects. Among the most notable are ClearML [34], Weights & Biases [21], TensorBoard [1], and MLflow [41], each offering different strengths and capabilities tailored to different aspects of the research and development process.In this work, we

adopted ClearML due to its modular and comprehensive support for all stages of the ML pipeline. The framework integrates key functionalities such as experiment tracking, dataset versioning, pipeline orchestration, diagnostics, and model deployment within a unified environment. In contrast to tools like DVC, which are primarily focused on data and model versioning, TensorBoard, which is limited to visualization, or Weights & Biases, which focuses mainly on experiment tracking and HPO automation, ClearML offers a more complete solution. Compared to other end-to-end frameworks such as MLflow, it provides a flexible agent-based orchestration system and a solid data versioning feature, all with a native Docker integration. These features aligned well with the articulated pipeline implemented in this case study.

In this section, we give an overview of Docker first and then ClearML as those are the two frameworks that were used in the case study in section 3.5.3.

Virtualization

Container-based virtualization provides a lightweight and efficient way to ensure a consistent execution environment across different machines. Practices generally seen in the literature, such as specifying the project dependencies in a 'requirements.txt' file to improve reproducibility, do not take into account the configuration differences between machines, leading to potential errors and inconsistencies when replicating environments. These challenges are limited not only to the primary dependencies, but also to the sub-packages that play a crucial role in the background but not generally listed in the configuration file.

The Docker [97] virtualization framework addresses these issues by encapsulating the entire application and its dependencies in Docker containers. These containers provide a lightweight, isolated environment to run experiments. Containers can be created from Docker images that are built based on instructions specified in a Dockerfile. The advantage of using Docker images is that we can easily transfer them between computers, ensuring fully reproducible environments for running the experiments across multiple research centers.

MLOps Frameworks

MLOps frameworks are essential tools for integrating MLOps practices effectively and efficiently in ML and DL projects. Some tools like Tensorboard [1] or DVC [77] help in integrating some MLOps practices, while others like ClearML [34] and Wheight&Biases [21] help in integrating all the main practices with a holistic approach.

The ClearML framework is structured around the concept of 'tasks', which are fundamental units of execution that represent a stage of an ML workflow, including:

- Data preprocessing

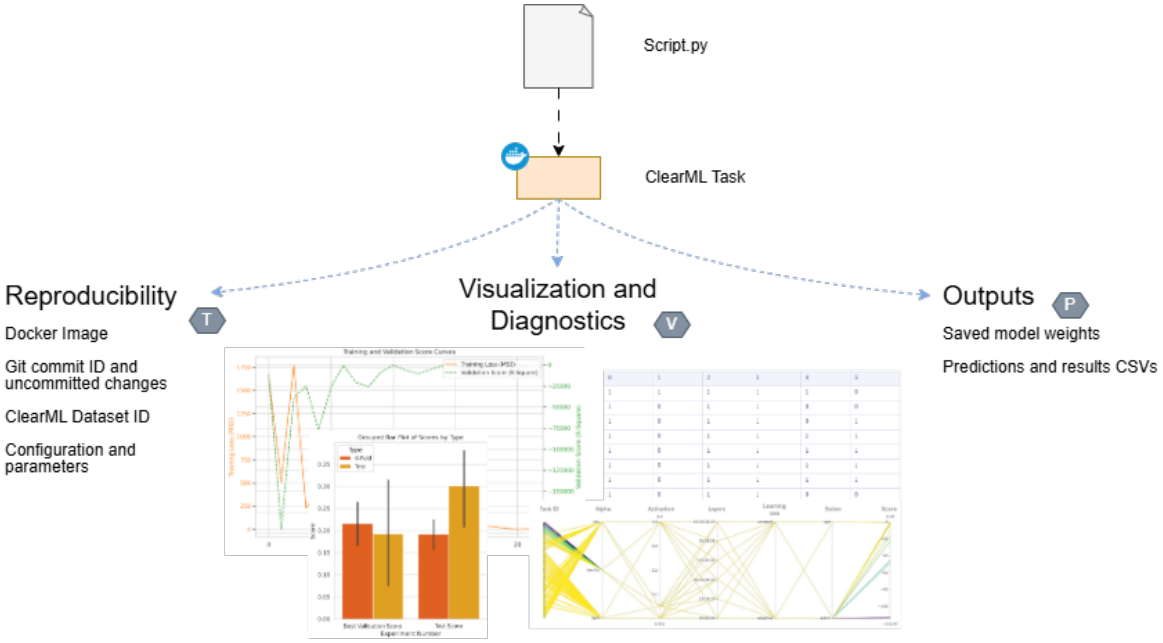


Figure 3.11: *Typical logs of a ClearML task. Items marked **[T]** capture experiment tracking information (environment, code, datasets); items marked **[V]** show training metrics and visualizations; and items marked **[P]** contain exported models and result artifacts.*

- Model training
- Hyper-parameter optimization
- Model evaluation

Those tasks can log and track all the parameters, plots, and outputs of the running experiment and register the details of the experiment and the configurations for reproducibility, as exemplified in the schema in Figure 3.11. In ClearML, tasks are generally executed inside Docker containers, ensuring standardized environments for running experiments.

Task Management Tasks in ClearML are fundamental units of execution that wrap experiments or processing steps and enable a detailed logger that keeps track of the configuration details, the parameters, the output metrics, and the plots the execution generates while running.

Moreover, these tasks can be used either to summarize the details of an experiment and its outputs or as a module in a workflow facilitating complex sequences through automatic cloning and launching of tasks, for example, for the purpose of HPO or complex pipelines.

Dataset Management ClearML also has a dataset versioning feature. This enables precise tracking of dataset changes over time, allowing for the verification of specific datasets and versions used for future tasks.

Hyper-parameter Optimization Agent The HPO Agent in ClearML automates the process of searching for the optimal set of values for the model's hyper-parameters from a predefined search space. This is done automatically by cloning the tasks and assigning them new parameters. The agent also logs the results of the child tasks, enabling the analysis of the effectiveness of the different hyper-parameters.

Pipelines The pipeline functionality of ClearML allows for the orchestration of complex workflows, where tasks can be chained or executed in parallel (on different GPUs if needed) based on dependencies and conditions. This feature enables the automation of end-to-end processes, from data preprocessing to training and evaluation, where different tasks can interact with each other by exchanging data.

Deployment in Production ClearML provides native support for model deployment through its ClearML Serving component, which allows trained models to be exposed via RESTful APIs. The models trained and evaluated from the previous tasks are saved, so they are accessible for inference purposes. Moreover, the use of containerized environments ensures a reproducible and portable setup, facilitating the integration of the model in external systems for production use. Moreover, ClearML Serving includes features for dynamic model updates and predictions monitoring, offering a scalable and robust framework for handling model inference in real-time contexts.

3.5.3 Case Study Implementation

Description of the Study

To demonstrate the benefits of using MLOps frameworks for structuring applied DL research in agronomy, we selected the study of [126]. This study was chosen not as a performance benchmark, but as a representative example of an applied DL workflow on publicly available foodomics data [31], for which both code and data are accessible.

The work from [126] focuses on predicting five agronomic traits, namely grain yield, grain protein content, heading date, plant height, and weight, from data collected between 2014 and 2016. The predictions are based on the single nucleotide polymorphisms (SNPs) markers of the different plant breeds. The authors presented two DL algorithms in their study, Multi-Layer Perceptron and Convolutional Neural Network, and compared their performance to a ridge regression model, the best linear unbiased predictor.

Initially, the dataset goes through a filtering process, followed by scaling and finally standardization. Then the analysis proceeds with splitting the data into train-test splits and training of the ML models with 200 replicates with different model initializations to evaluate the robustness of the NN.
Each of these 200 iterations involves HPO through a grid search strategy [80] in combination with k-fold CV [10] for each hyper-parameter combination. Ultimately, the model parametrized with the best hyper-parameters is trained on the full train data, and the evaluation is performed on the holdout test set.

To demonstrate the integration of MLOps practices, we focused on the MLP model and the Plant Height 2014 trait. For the purpose of showcasing the ClearML framework and its core functionalities, it was not necessary to evaluate all model architectures on every trait. Instead, selecting a representative model and trait was sufficient to highlight the advantages and applicability of the proposed MLOps approach. The Plant Height 2014 trait was selected because it showed the highest predictive performance among the 15 traits and growing seasons of the datasets. Similarly, the MLP was chosen due to its superior performance in the original study and its compatibility with key ClearML features, such as GPU execution and training diagnostics logging. In contrast, as the CNN implementation was based on Scikit-learn [117], it does not provide access to detailed training logs (e.g., learning curves) and lacks GPU support, making it less suitable for integration with a framework like ClearML.

Virtualization Through Docker

The first step to integrate MLOps practices in the project was to develop a virtual environment to ensure code reproducibility across machines. As in the study two programming languages were used, Python for training the NN and R for filtering the raw dataset, two Docker images were created, one for each programming language.

Dataset Preparation and Versioning

Once the environment was set up, the next step was to integrate the MLOps framework into the dataset preprocessing and management workflow of the project. First, we decided to exclude the dataset from the git version control to avoid overloading the repository; so the dataset file was untracked and added to the `.gitignore` file. Then, we integrated the ClearML data versioning by creating a dataset instance with the dataset file. This dataset instance can be fetched with one line of code. If no version is specified, the latest version will be downloaded locally and made available for any further execution. Even if this simple operation may seem superfluous, in this way, we have a copy of the dataset accessible in a read-only manner from different possible executions, avoiding the risk of eventual manipulation from parallel runs, which would lead to data corruption. Moreover, we also have the version of the

dataset instance, which becomes very convenient when the data preprocessing step is modified. When a modified dataset is uploaded under the same name, the old version will not be overwritten, but a new version will be created. This not only ensures that the previous version is available for further executions but also guarantees that each past execution is documented with the version of the dataset used, as we will see next, in Section 3.5.3.

To fully exploit the capabilities of the dataset feature of ClearML, the preprocessing was broken down into 6 steps, shown in the interactive schema in figure 3.12. On the right side of the figure, we can see the sequence of steps that generate a new preprocessed dataset. Between the bottom two blocks ("Original Dataset", "Dataset Filtered") the preprocessing was performed with the original R script, whereas to execute the rest of the blocks, the data preprocessing pipeline of the original Python script was broken in steps, to have a chain of dataset instances.

After filtering performed with the R script, the data was scaled with a min-max scaler that brought both features and targets between 0 and 1. To map back the model's predictions, the scaling parameters for the target were saved in a file in the dataset to be accessed later for results visualization. After scaling, the dataset was split into train and test to create a holdout set for evaluating the performance of the model on independent data, and the trait Plant Height of 2014 was selected. Finally, the features of the dataset were standardized and ready for use in model training. On the left side of the interactive dataset schema in figure 3.12, the versions of the dataset selected on the right side are displayed. Whenever this dataset is uploaded (for instance, when running the processing with different parameters), a new dataset version is generated instead of overwriting the dataset instance. The R script for the initial step was executed within the Docker container designed to run the R language, whereas the following steps were performed with the Docker container designed for running Python code.

Experiment Tracking and Modularization of DL Workflows

With the dataset ready for the model training, we integrated the ClearML framework into the training section. We decided to leverage the framework's full capabilities and showcase its extensive features. To do so, modifications to the original project's structure were necessary, but these adjustments did not alter the project's fundamental steps and logic, ensuring that the essence of the workflow remained intact.

The very first step was to add the initialization of the ClearML task. In this way, the framework can keep track of all the information regarding the script, the current git commit is reported as well as the uncommitted changes, the parameters of the executed task, such as the seed and the NN hyper-parameters, and all the logs, such as the console output and the plots generated during the execution.

Secondly, the data preprocessing was removed from the original script and sub-

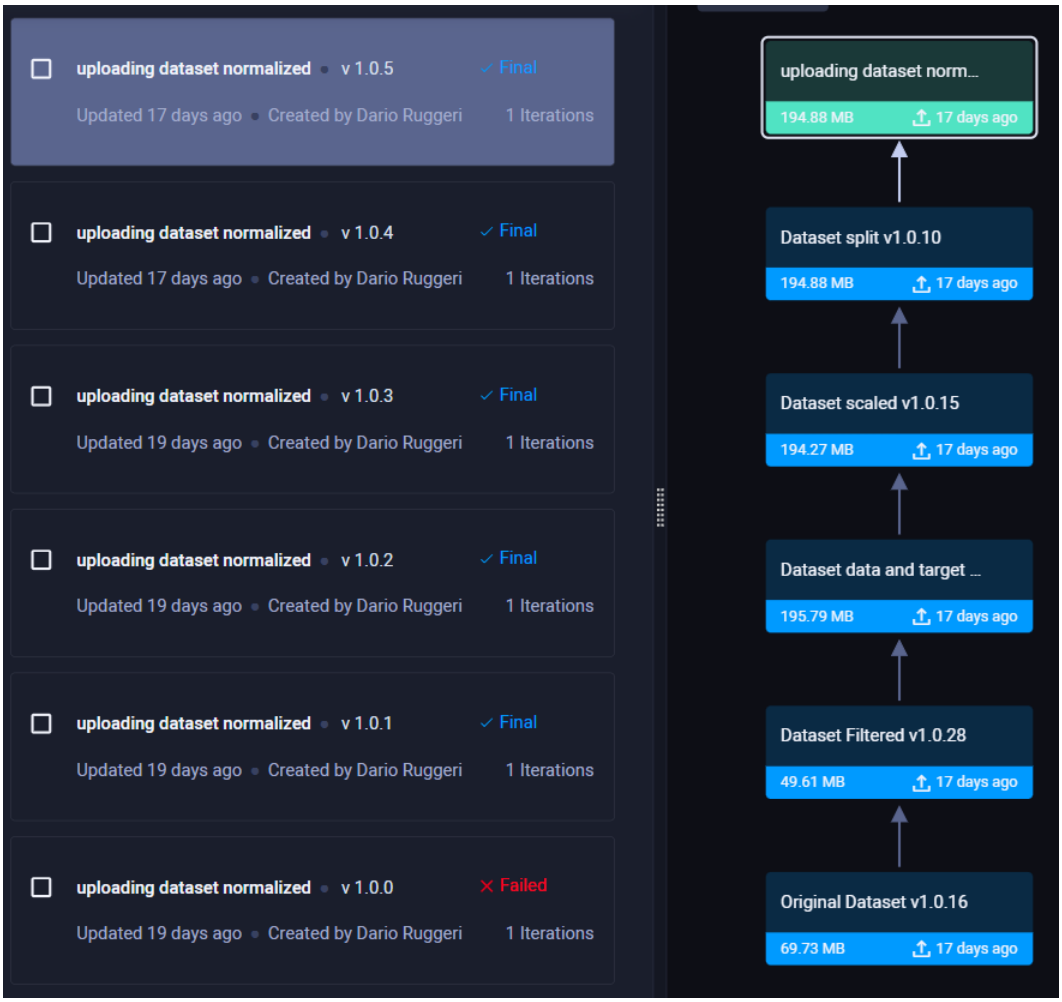


Figure 3.12: *In this screenshot of the ClearML web interface, the dataset instances generated during the integration of the MLOps framework in the preprocessing section of the project are displayed. On the right side the different dataset entities, while on the left the versions of the selected dataset entity are shown.*

stituted with the dataset fetching lines. As a reminder, the dataset preprocessing is done separately in our workflow, and the preprocessed dataset is saved in ClearML. Before training, the preprocessed dataset is downloaded from ClearML (if that dataset instance is not already present) and used as is. In this way, the training task is bound to the dataset used, and in the configuration, it is possible to track the dataset and its version.

To exploit all the features of the MLOps framework, we had to break the authors' original workflow into smaller ClearML tasks, like in the schema shown in figure 3.13. Each experiment was composed of two stages: an HPO for finding the best hyper-parameters, with a k-fold CV, and then the evaluation on the holdout split.

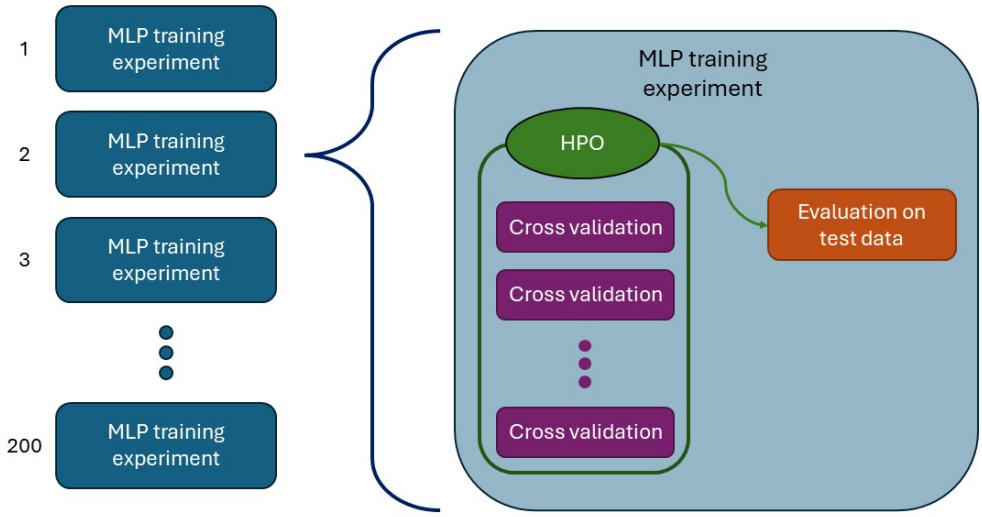


Figure 3.13: *Structure of the workflow designed by the authors in [126], each of the 200 experiment replicates feature first a HPO with k-fold CV and then a training of the model with the best hyper-parameters on the entire train set and evaluation on the holdout set.*

Training Diagnostics and Hyper-parameter Optimization The integration of ClearML in the authors' workflow started by implementing the task that evaluates the MLP model with the k-fold CV. To accomplish this, it was sufficient to first isolate the authors' code related to model creation and KFlod CV evaluation; and then to encapsulate this in a ClearML task by moving it to a separate script and initiating the ClearML task at the beginning. To integrate this task with the HPO agent, it was only necessary to log the average metric obtained on the k iterations of the CV and the

hyper-parameters of the model.

As soon as the execution of the code is encapsulated within a task, it becomes simpler to log further information and keep track of the results of a specific execution by simply reporting the plots and tables through the ClearML methods. These results are uploaded and become visible on the ClearML web interface. Moreover, diagnostics and results such as loss curves of the model (figure 3.16), results across the CV iterations (table 3.7), and predictions for a few samples (figure 3.19) are linked to the corresponding task and accessible on the same web page.

HPO in ClearML is performed by the ClearML agent. To set up the agent, we first defined the search space and then specified the base task, which performs the training and evaluation of the model in the CV framework. This way, the agent has all the information to perform the HPO by spawning clones of the given task, using different combinations of the hyper-parameters, and collecting their results.

This approach meticulously replicates the authors' workflow and offers multiple advantages, particularly in terms of transparency and the multi-level analysis of the results. Before integrating this framework, the information retrievable from the HPO was the performance of the models trained with the different hyper-parameter combinations. To derive meaningful insights about the NN's hyper-parameters, one would need to manually link this performance data with the specific hyper-parameters employed. The ClearML agent streamlines this process by reporting such figures automatically and in real time. As illustrated in figure 3.14, the dynamic parallel coordinate plot enables a direct correlation between the hyper-parameters and the scores achieved during CV.

The examination of the parallel coordinate plot can bring important insights and lead to alternative choices in the design of the HPO. For instance, it seems that most executions yield similar results, indicating that certain hyper-parameters have a marginal impact on the model's performance, suggesting potential benefits of expanding the search space. In addition, specific configurations, such as the identity activation function or the Adam optimizer for gradient descent, lead to significantly worse outcomes. This observation could suggest the transition from grid search strategy to more advanced Bayesian search [105], which can dynamically focus on more promising configurations, thereby optimizing the use of computational resources. While grid search was adopted here to remain consistent with the original study, it is worth noting that the ClearML HPO module fully supports more sophisticated optimization approaches such as Bayesian search, which could be integrated with minimal changes to the pipeline structure.

In addition to the parallel coordinate plot, the agent also reports a table (shown in figure 3.15) where the first column is a clickable link to the actual task run. In this way, we can access any task launched by the HPO agent and all its logs, such

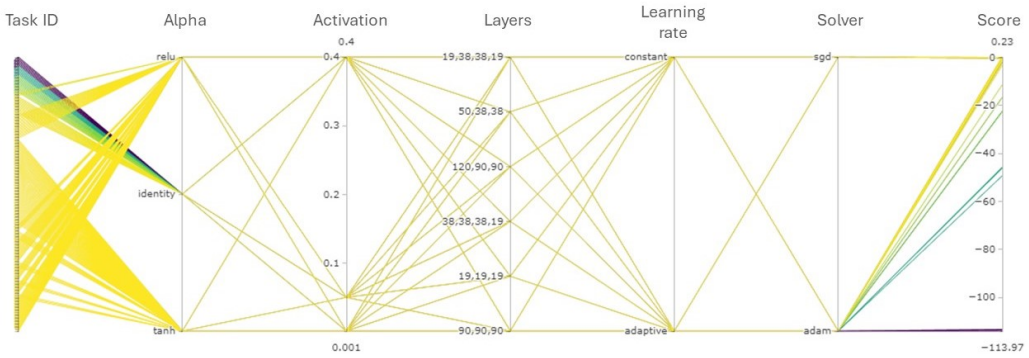


Figure 3.14: *Parallel coordinates plot automatically generated by the ClearML HPO module. Each line corresponds to one run, linking the chosen hyper-parameters to the resulting performance score.*

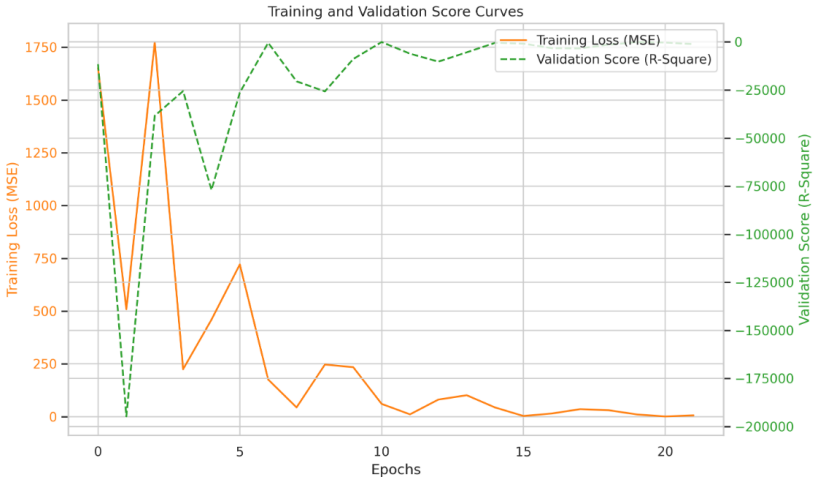
as the console log, all the plots and figures, and the reported results. Since we have logged the learning curves, for instance, we can inspect how one of the executions parameterized with the identity activation function behaves compared to the one with ReLU on the first k-fold: in figure 3.16 we can see that the model is unable to train when parameterized with the identity activation function, that is probably the reason of its low performances.

task id	objective	iteration	hyper-parameters/ac	hyper-parameters/al	hyper-parameters/hi	hyper-parameters/le	hyper-parameters/so	seed/seed	status
e30bb4cdfd884aadb(	0.113051976164330	0	relu	0.4	90,90,90	constant	adam	7220	completed
756a1e05f58f438a93	0.113051976164330	0	relu	0.4	90,90,90	adaptive	adam	7220	completed
67d172138db44a6e8	0.096296192825975	0	relu	0.05	90,90,90	constant	adam	7220	completed
23f0c8b43f504c6697	0.096296192825975	0	relu	0.05	90,90,90	adaptive	adam	7220	completed
06db4a06e43c40d18	0.079911506699541	0	relu	0.4	19,19,19	constant	adam	7220	completed
f1886814f0f04a58a7	0.079911506699541	0	relu	0.4	19,19,19	adaptive	adam	7220	completed
bf27df76b852472eb4	0.074362079339728	0	tanh	0.4	120,90,90	constant	sgd	7220	completed
92449b2214aa4bc7a	0.074362079339728	0	tanh	0.4	120,90,90	adaptive	sgd	7220	completed
85cd000fb05642e5b	0.074248589126536	0	tanh	0.001	120,90,90	constant	sgd	7220	completed
560522d23aa6465b8	0.074248589126536	0	tanh	0.001	120,90,90	adaptive	sgd	7220	completed
1ec0615c463741e8a	0.055440746161824	0	relu	0.001	38,38,38,19	constant	adam	7220	completed
70229e2853574fb7a	0.055440746161824	0	relu	0.001	38,38,38,19	adaptive	adam	7220	completed

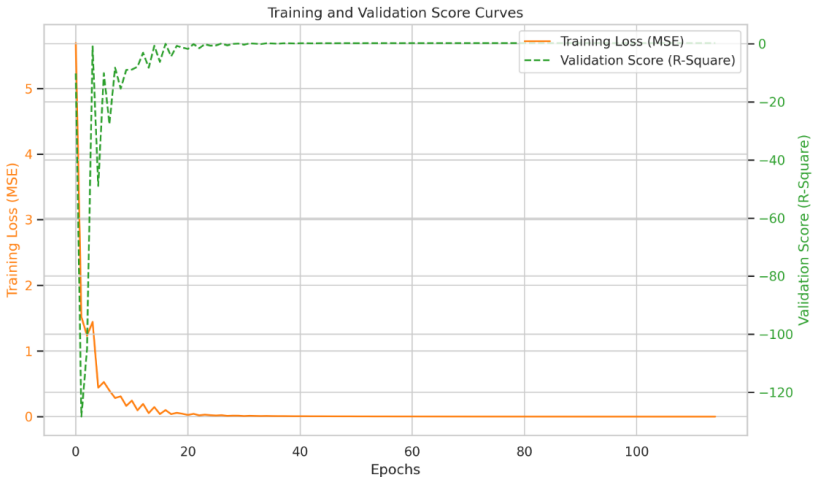
Figure 3.15: *This screenshot shows the dynamically updated table of the tasks executed by the ClearML HPO agent. In the table are summarized: the hyper-parameters, the result, and the link to the specific task; the link leads to the task page where all its logs and information can be analyzed.*

Moreover, we can inspect the results for the 5 folds of the CV to check the robustness of the model. Table 3.6 shows the results of the k-fold iterations logged by the task parameterized with the identity activation function. We can notice the large difference between the different evaluation folds both in terms of R^2 score and in terms of correlation (the metric used in the original study [126]).

Also, it is worth checking the results obtained with the best hyper-parameter sets



(a) *Learning process when the NN is parameterized with the identity activation function.*



(b) *Learning process when the NN is parameterized with the ReLU activation function.*

Figure 3.16: *Learning curves logged during training on the same fold of the k-fold CV. Orange shows the training loss, and green shows the validation R^2 score, allowing comparison of different activation functions.*

Table 3.6: *K-fold results when the NN is parametrized with the identity activation function.*

k-fold iteration	Best Validation Score	Test Score	Test Correlation
1	-58.148	-28.853	0.193
2	-22.412	-22.334	0.183
3	-2.063	-2.270	0.254
4	-1.338	-1.899	0.246
5	-1.583	-1.639	0.126
Average (STD)	-17.109 (24.640)	-11.399 (13.163)	0.200 (0.052)

on the same k-fold CV splits. This can be done by simply opening the task's page to consult the results; indeed, any task run in the HPO fashion has all these logs reported. In table 3.7, the model appears more robust, but still exhibits large deviations in some folds.

Table 3.7: *K-fold results when the NN is parametrized with the ReLU activation function, the best parameter set found by the HPO agent*

k-fold iteration	Best Validation Score	Test Score	Test Correlation
1	0.198	0.361	0.604
2	0.184	0.092	0.334
3	0.182	0.245	0.503
4	0.265	0.233	0.490
5	0.347	0.212	0.468
Average (STD)	0.235 (0.0711)	0.229 (0.0960)	0.480 (0.0968)

This analysis is just a taste of what insights we can gain from accessing the logs of the HPO agent's child tasks, an analysis that was difficult to perform on the original implementation.

Final Evaluation and Predictive with Best Hyper-parameters The last step of the workflow was to train the model with the best hyper-parameters on the whole training data split and then to test it on the holdout split. This ClearML task fetches the best configuration from the HPO task (hyper-parameters achieving the best results and the corresponding dataset version) and starts the model training. The results of this task are compared with those of the k-fold data splits, as illustrated in Figure 3.17, providing additional information on the robustness of the model. We can see that the holdout set results align well with the validation folds results of the CV. Indeed, the test set result falls in the middle of the CV results distribution for both the correlation and the R^2 score; however, we can notice that the distributions of the CV results are

pretty wide, which might suggest that the model is not very robust.

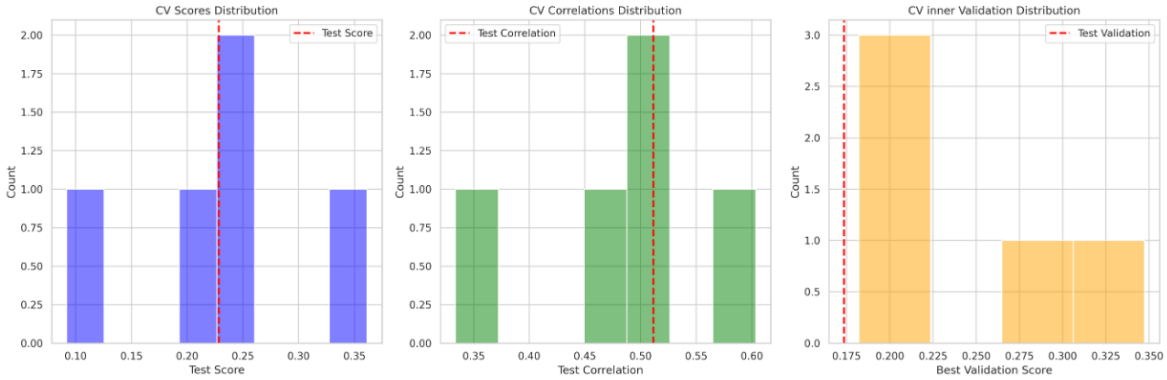


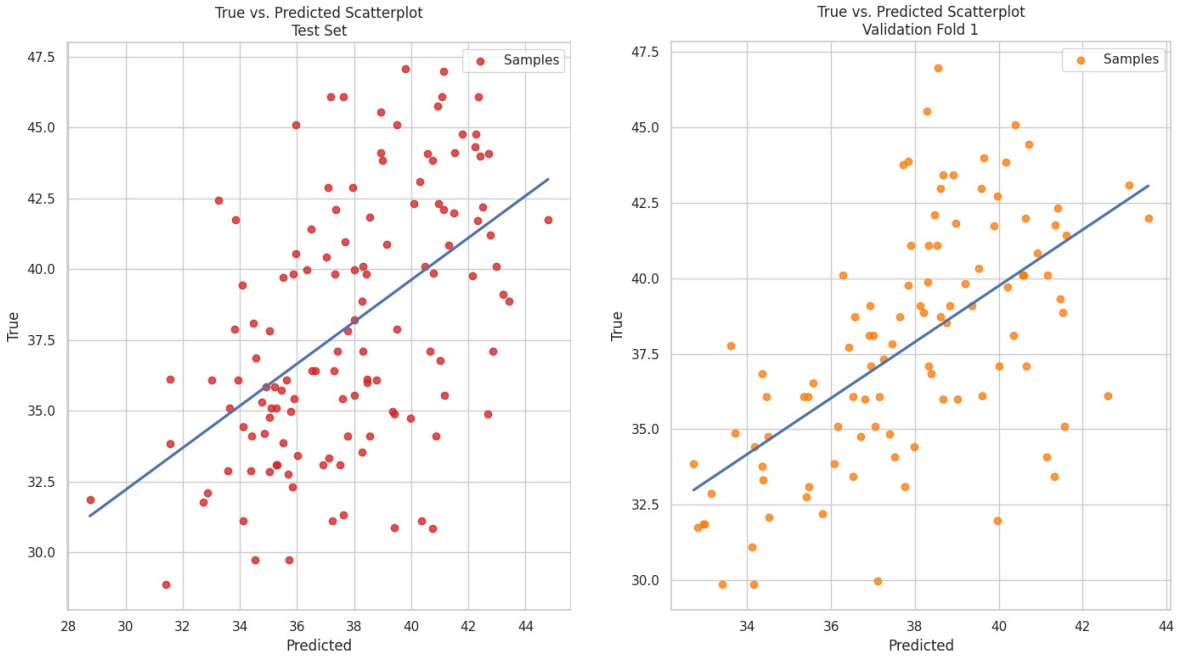
Figure 3.17: *Comparison between k-fold CV results and the holdout evaluation for the model trained with the best hyper-parameters. Each histogram shows the distribution of scores across folds: in blue the R^2 scores, in green the correlation between predictions and true values and in orange the R^2 score on the validation split made for the purpose of ES. The dashed red line marks the score obtained on the holdout set.*

Moreover, by reporting further plots, we can have a visual representation of the model predictions by navigating the corresponding task logs through the ClearML user interface. In our implementation, a scatterplot was reported to visualize the correlation metric (which was referred to as accuracy in the original implementation), and to show the model's predictive power, a barplot was reported, comparing true and predicted traits for some test or validation samples.

Figure 3.18 displays scatter plots for the task that optimized hyper-parameters on the holdout split and the first k-fold CV fold of the best-performing task in HPO. The fitted blue line cutting the plot represents the correlation between the predicted and true values. This figure provides a visual aid to understand how this metric relates to the prediction power of the model. It is important to note that as correlation is insensitive to shifts and scaling of the features, it may not be an ideal metric to evaluate the prediction error. In fact, a model with perfect correlation may still exhibit substantial prediction errors, either due to biases in predicted values or variability in predictions.

To effectively visualize the prediction error of the model in absolute terms, we can inspect the bar plot in Figure 3.19, the predictions of the model are shown for the first 60 test samples. This bar plot is also saved for other tasks trained on K-fold CV splits, showing the samples from the validation folds.

Pipeline Automation and Results Aggregation The last step of the ClearML integration was to design a workflow that runs the HPO agent and the evaluation of the best parameters on the holdout set one after the other, and repeats this process



(a) *Scatter plot showing the relationship between predicted and true values on the holdout set*

(b) *Scatter plot showing the relationship between predicted and true values of the first fold of the k-fold CV evaluation*

Figure 3.18: *Scatter plots comparing predicted and true values for the model with the best hyper-parameters. Panel **(A)** shows the holdout set, and panel **(B)** shows one fold from the k-fold CV. Points represent samples, and the blue line is the fitted correlation.*

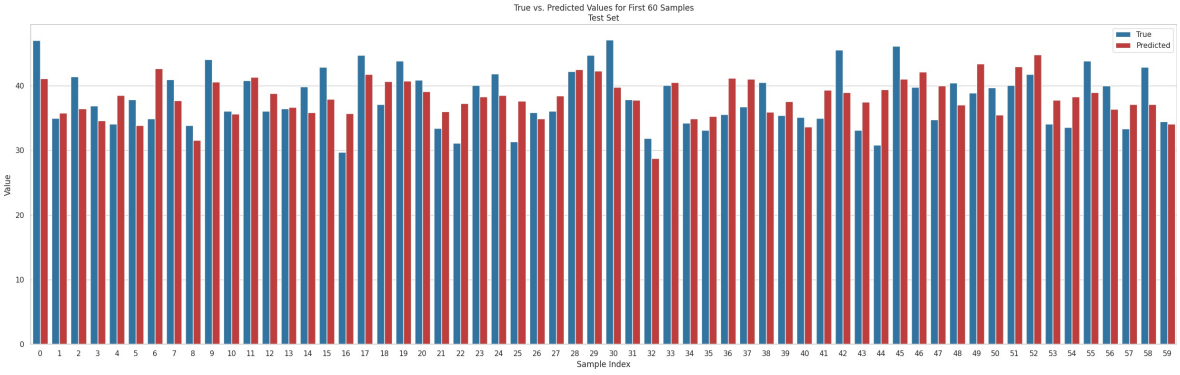


Figure 3.19: *Bar plot, logged in the ClearML task, comparing true versus predicted values for the first 60 samples from the holdout set, illustrating the model's predictive accuracy.*

several times to emulate the authors' replicates (200 in the original setup).

To implement this structure, we used the pipeline feature of ClearML. This modular approach allows us to run multiple tasks in a custom structure and aggregate their results. Moreover, in this way, all the involved tasks can be inspected individually to have detailed information about the execution and can even be run in parallel on different GPU cores.

Following the schema in Figure 3.13, the pipeline first creates an HPO task to find the best hyper-parameter, then launches an evaluation task to test these parameters on the test data. This sequence is repeated for each experiment replicate, then an aggregator task collects the results from each replicate and visualizes them in graphical summaries. For simplicity and to reduce computation time, only 8 replicates were generated instead of the 200 generated in the original study. While this reduction limits the statistical power of the evaluation, it is sufficient to illustrate the structure, execution, and monitoring capabilities of the proposed MLOps pipeline, which was the primary objective of this work.

Figure 3.20, represents the interactive schema of the pipeline, visualized also in the ClearML user interface. To avoid spawning too many tasks at the same time, each HPO task waits for the previous HPO task to finish. The evaluation task waits for its HPO task to get the best hyper-parameters, and the aggregator task has to wait for all the previous evaluations on the holdout set to start summarizing the pipeline results. To adhere to the authors' workflow but still maintain fully reproducible execution, the

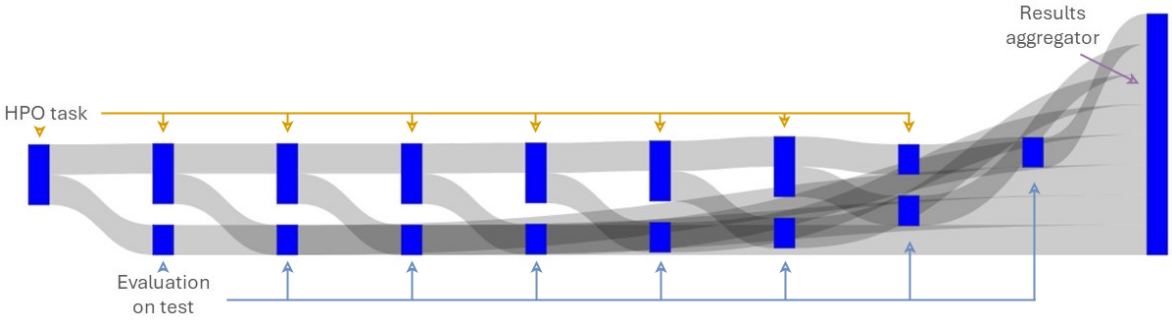


Figure 3.20: *Interactive pipeline view generated by the ClearML pipeline agent. Each box represents a task, and shaded connectors indicate execution dependencies. HPO tasks run first, followed by evaluation tasks and a final aggregation step.*

pipeline was initialized with a global seed that was then used to derive a different seed for each of the replicate tasks (as one replicate is composed of the HPO task and the holdout set evaluation task in our setup, the two were receiving the same seed). In this way, each experiment maintains its unique variability, but rerunning this pipeline with the same seed would give the same results.

Regarding results reporting, each of these tasks can be inspected as we did in the

previous sections 3.5.3, 3.5.3, revealing all the logged results, configurations, and diagnostics at any depth level. Moreover, since the results were logged inside tasks, they can be accessed at a later time to make a further analysis without the need to re-run the whole execution. Furthermore, all the models trained by the tasks are saved, which means that it is possible to access those models individually to make further predictions or an eventual fine-tuning.

The pipeline results are summarized in the aggregator task. It reports the results of the evaluations of the best hyper-parameters on the holdout set and the results of the same parameters on the k-fold CV. With such a task only responsible for generating plots and statistics of the previous tasks, we have a module that can store an in-depth analysis of the relationships between the experiment replicates, which can be inspected at any time. We can notice that the table documenting the best hyper-parameters found by the replicates (Figure 3.21) shows that diverse combinations of hyper-parameters emerged as optimal from the eight replicates.

Experiment n.	activation	alpha	hidden_layer_sizes	learning_rate	solver	R^2	Test Correlation
0	relu	0.4	[50, 38, 38]	constant	adam	0.184 (0.11)	0.446 (0.134)
1	relu	0.001	[19, 38, 38, 19]	adaptive	adam	0.278 (0.058)	0.54 (0.048)
2	relu	0.001	[38, 38, 38, 19]	constant	adam	0.195 (0.106)	0.463 (0.123)
3	relu	0.001	[19, 38, 38, 19]	adaptive	adam	0.201 (0.144)	0.392 (0.312)
4	relu	0.05	[19, 38, 38, 19]	constant	adam	0.131 (0.173)	0.322 (0.281)
5	relu	0.4	[19, 19, 19]	adaptive	sgd	0.167 (0.108)	0.433 (0.11)
6	relu	0.4	[19, 38, 38, 19]	constant	adam	0.221 (0.083)	0.505 (0.064)
7	relu	0.05	[19, 38, 38, 19]	constant	adam	0.155 (0.155)	0.421 (0.196)

Figure 3.21: *Best hyper-parameters found by the HPO agents of the 8 replicates*

Figure 3.22 shows the distribution of the results of the 8 experiment replicates on the holdout test.

With the workflow designed in modular way, becomes easier to have a general view of the ML project. indeed we can observe that every model evaluation task takes in input the dataset pre-divided into training and test sets. Consequently, the replicates assess the model performance on the same test data split; this reduces the effectiveness of the model robustness assessment, shifting the evaluation to the variability introduced by model initialization and robustness of the HPO procedure, as those remain as the only sources of variability.

This insight could drive to a different evaluation schema like the Monte Carlo CV or nested k-fold CV [10] to effectively assess the robustness of the DL model on different hold-out splits. On the other hand, in the current evaluation schema we can still assess the model robustness with the distribution of the K-fold CV results, saved in individual tasks and depicted in plots within the aggregator task. The violin plots, illustrated on the right panel of figure 3.23, show wide variations in the k-fold CV evaluations across the different tasks. Moreover, on the left panel of figure 3.23,

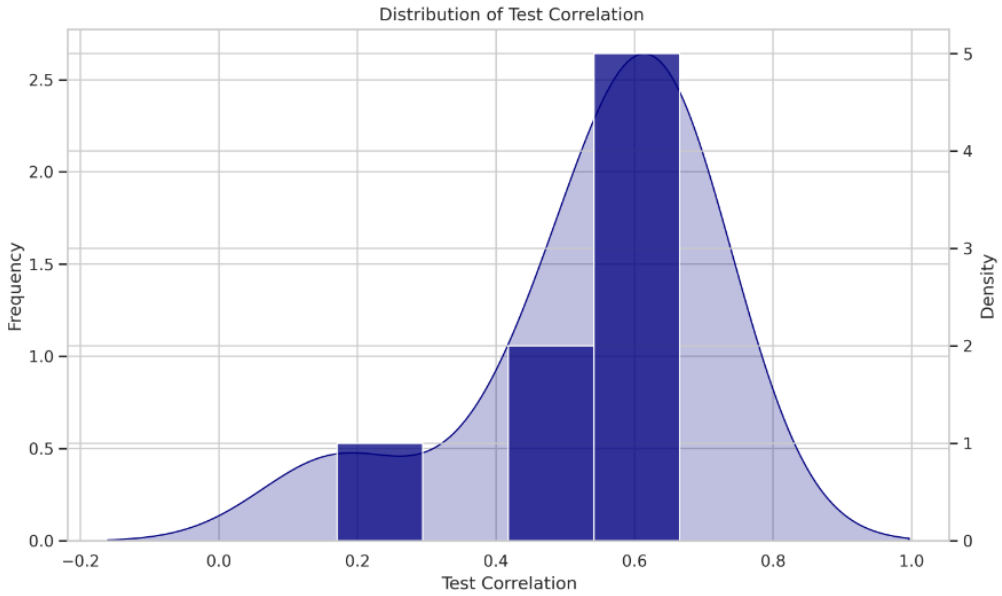


Figure 3.22: *Distributions of the 8 replicates' results on the holdout set. The metric used is the correlation between predictions and actual values, the same reported by the authors in the original study.*

the bar plot suggests that the evaluations on a unique holdout split may present a biased (in this case, optimistic) estimate of model performance when compared to the model assessment on different data splits produced by the k-fold CV evaluations; again advocating for assessment on different data splits in phase of model testing.

3.5.4 Challenges and Limitations

While the adoption of the MLOps frameworks can bring benefits in aspects such as reproducibility, transparency, and the long-term usability of the code base, it can also introduce some overhead, such as increased storage requirements for dataset and model versioning, longer runtimes due to the initializations of the tasks, and a learning curve when first integrating it into the workflow. In our analysis, these aspects are largely outweighed by the benefits of these frameworks. Moreover, ClearML offers enough flexibility to support a soft and gradual adoption, making it suitable even in contexts where a full MLOps setup is not necessary.

In parallel, it is important to consider potential failure modes that may arise during the execution of complex ML pipelines. For instance, if articulated workflows are not carefully designed, there might be mismatches between dataset versions across different tasks, or environment inconsistencies may emerge if the results from tasks that were run using an outdated Docker image are used by future tasks. Additionally, pipelines composed of many tasks can become difficult to navigate, and memory usage

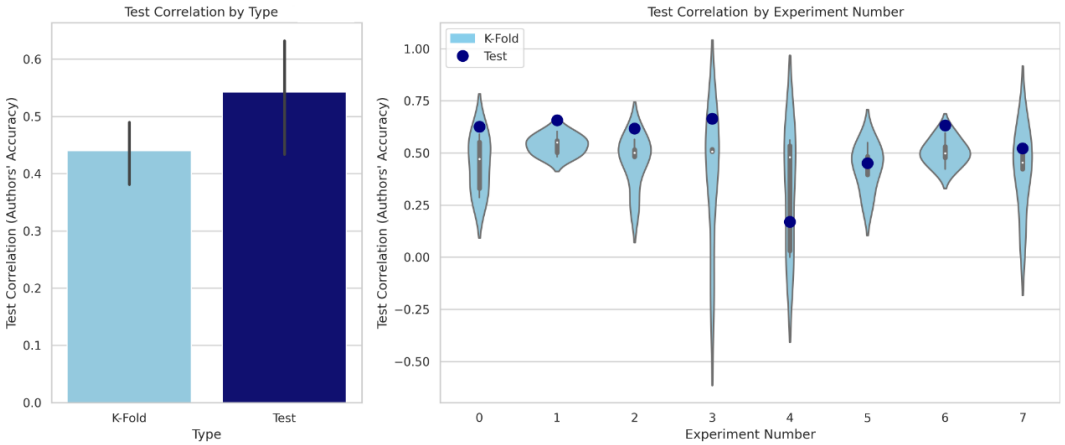


Figure 3.23: *Comparison between holdout evaluation and k-fold CV results across the pipeline's 8 replicates. The bar plot (left) summarizes mean performance with 95% bootstrap confidence intervals, while the violin plots (right) show the distribution of k-fold scores for each replicate together with the corresponding holdout score.*

can quickly grow if unnecessary model weights are saved, for example, those of the tasks generated by the HPO agents. Despite these possible sources of error, ClearML provides several mechanisms to mitigate them, including a web interface to manage executed tasks and free up storage, detailed task logging, version tracking, and the ability to resume the execution from failed stages of a pipeline or the possibility to roll back to previous configurations. This level of transparency and control helps preserve the integrity of the pipeline, even in the presence of these challenges.

3.5.5 Conclusion

In this study, we explore and showcase the transformative power of MLOps frameworks for ML-based research projects, which had no employment in the recent agronomic literature. By integrating the ClearML and Docker frameworks in a literature study that deployed a DL model in the foodomics research field [126], we not only highlight the potential of adopting these practices in the agronomic context, but also provide step-by-step guidelines for their efficient integration.

After a few adjustments to the original study's code base, we integrated the authors' complex workflow into the ClearML framework, demonstrating the benefits of MLOps practices. Indeed, by encapsulating the original execution within Docker containers and utilizing the ClearML framework to log configurations and version control information, such as the commit of the current code version, an enhanced level of reproducibility was achieved, ensuring long-term, well-documented reproducible executions.

Moreover, with a modular and structured approach, the authors' complex workflow

was broken down into smaller experiments that could be individually investigated, thereby acquiring insights from their executions at multiple depth levels. This structure, combined with detailed reporting of results and diagnostics, enabled a more informed understanding of the implementation, leading to improved experiment design and a higher quality research workflow. Moreover, since each trained model is saved and linked to its execution environment, the proposed pipeline also supports a seamless transition to production through the ClearML Serving feature.

To clearly highlight how the proposed architecture aligns with the core principles of MLOps, Table 3.8 summarizes the key capabilities addressed in this work and the corresponding components in our integration.

Table 3.8: *Overview of MLOps Principles and Their Implementation in the Proposed ClearML-Based Workflow.*

MLOps Capability	**Implemented Feature in ClearML**	**Section**
Experiment Tracking	Tasks track parameters, code version, logs, metrics, and outputs	3.5.3
Dataset Management	Dataset objects versioned across preprocessing stages	3.5.3
Pipeline Automation	Modular tasks executed in a coordinated workflow via ClearML Pipelines	3.5.3
Visualization	Dashboard supports visualizing learning curves, prediction plots, HPO metrics	3.5.3
Model Deployment	Trained models stored as artifacts, linked to execution environments, and exportable for deployment via ClearML Serving	3.5.2

This mapping highlights how the proposed pipeline meets the core requirements for building reproducible and production-ready ML systems, as emphasized in recent MLOps literature [18, 59].

With this work, we demonstrate how MLOps practices and frameworks offer a structured and reproducible approach to ML research in the agronomic field even prior to the deployment stage. As the field continues to evolve, the adoption of these practices will play a key role in advancing our understanding and leveraging the full potential of ML in agronomy and beyond.

3.6 Discussion and concluding remarks

This chapter examined applied ML for omics data from both a modeling and an experimental workflow perspective. Through two complementary studies, it demonstrated how advances in applied DL methods and in research-stage experimentation practices jointly contribute to more reliable and interpretable omics data analysis.

The primary contribution of the chapter is the applied DL study on wheat SNP data, which showed that integrating modern training practices into NN models can lead to consistent and statistically significant performance improvements over previously reported results. Beyond establishing new reference performance on a public dataset, the study emphasized robustness across experimental settings and incorporated explainability methods to provide insight into model behavior. By identifying influential SNP markers through post hoc interpretation, the work illustrates how DL models can support both accurate prediction and transparent analysis in foodomics applications.

Complementing this applied modeling contribution, the chapter also investigated the role of MLOps frameworks in structuring and managing complex ML experiments. Using a representative agronomic case study, this work demonstrated how experiment tracking, dataset versioning, modular workflow design, and systematic logging can improve transparency, traceability, and comparability of applied ML research. Importantly, the focus was placed on research-stage experimentation rather than deployment, and the study distilled a set of practical guidelines for integrating MLOps into scientific workflows, showing how these practices can support informed model development and iterative exploration in applied omics contexts.

Taken together, the results presented in this chapter show that strong applied ML outcomes in omics are not achieved through modeling choices alone. Instead, they emerge from the combination of robust model design, careful evaluation, interpretability considerations, and well-structured experimental workflows. This integrated perspective provides a foundation for conducting applied ML research in omics that is both scientifically rigorous and practically sustainable.

The author of this PhD thesis is responsible for the following contributions presented in this chapter:

III/1. Design of an applied DL framework for wheat SNP data analysis, integrating modern training practices tailored to high-dimensional, low-sample-size settings.

III/2. Rigorous empirical evaluation of the proposed framework, including statistical testing and comparison with previously reported baselines, leading to improved reference results on a publicly available wheat SNP dataset.

III/3. Integration of XAI methods into the applied DL pipeline, enabling feature-level interpretation of NNs predictions and transparent analysis of SNP relevance.

III/4. Investigation of MLOps frameworks for structuring and managing research-stage ML experiments in agronomy, illustrated through a case study on experiment tracking, dataset versioning, and modular workflow organization.

Declaration on the use of AI

During the preparation of this dissertation, AI-based language assistance tools were used exclusively for improving the readability and linguistic quality of the manuscript. In particular, Grammarly was used for spell-checking, grammar correction, and occasional synonym suggestions, while ChatGPT (base version) from OpenAI was occasionally used to improve sentence readability and phrasing.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. Software available from tensorflow.org.

[2] Emerson Abraham, Joao Reis, Aguinaldo de Souza, Marcos Morais, Oduvaldo Vendrametto, Pedro Neto, and Rodrigo Toloi. Time Series Prediction with Artificial Neural Networks: An Analysis Using Brazilian Soybean Production. 10(10):475, 2020.

[3] Nigatu Adossa, Sofia Khan, Kalle T. Rytkönen, and Laura L. Elo. Computational strategies for single-cell multi-omics integration. *Computational and Structural Biotechnology Journal*, 19:2588–2596, 2021.

[4] Zainal Ahmad, Mashitah Mat Don, Siti Hatijah Mortan, and Rabiatul Adawiah Mat Noor. Nonlinear Process Modeling of Fructosyltransferase (Ftase) Using Bootstrap Re-Sampling Neural Network Model. *Bioprocess and Biosystems Engineering*, 33(5):599–606, June 2010.

[5] Galih Kusuma Aji, Kenji Hatou, and Tetsuo Morimoto. Modeling the Dynamic Response of Plant Growth to Root Zone Temperature in Hydroponic Chili Pepper Plant Using Neural Networks. 10(6):234, 2020.

[6] Yaganteeswarudu Akkem, Saroj Kumar Biswas, and Aruna Varanasi. Smart Farming Monitoring Using Ml and Mlops. In Aboul Ella Hassanien, Oscar Castillo, Sameer Anand, and Ajay Jaiswal, editors, *International Conference on Innovative Computing and Communications*, pages 665–675. Springer Nature, 2023.

[7] Serhat Al, Fatma Uysal Ciloglu, Aytac Akcay, and Ahmet Koluman. Machine learning models for prediction of Escherichia coli O157:H7 growth in raw ground beef at different storage temperatures. *Meat Science*, 210:109421, April 2024.

[8] R. Alvarez. Predicting Average Regional Yield and Production of Wheat in the Argentine Pampas by an Artificial Neural Network Approach. 30(2):70–77, 2009.

[9] Jihen Amara, Bassem Bouaziz, and Alsayed Algergawy. A deep Learning-based Approach for Banana Leaf Diseases Classification. 2017.

[10] Sylvain Arlot and Alain Celisse. A survey of Cross-Validation Procedures for Model Selection. https://arxiv.org/abs/0907.4728v1, July 2009.

[11] Ibtissam Bakkouri and Karim Afdel. Multi-Scale Cnn Based on Region Proposals for Efficient Breast Abnormality Recognition. *Multimedia Tools and Applications*, 78(10):12939–12960, May 2019.

[12] Ibtissam Bakkouri and Karim Afdel. Computer-Aided Diagnosis (Cad) System Based on Multi-Layer Feature Fusion Network for Skin Lesion Recognition in Dermoscopy Images. *Multimedia Tools and Applications*, 79(29):20483–20518, August 2020.

[13] Ibtissam Bakkouri and Karim Afdel. MLca2F: Multi-Level Context Attentional Feature Fusion for CovId-19 Lesion Segmentation from Ct Scans. *Signal, Image and Video Processing*, 17(4):1181–1188, June 2023.

[14] Ibtissam Bakkouri, Karim Afdel, Jenny Benois-Pineau, and Gwénaëlle Catheline For the Alzheimer's Disease Neuroimaging Initiative. BG-3Dm2F: Bidirectional Gated 3D multi-Scale Feature Fusion for Alzheimer's Disease Diagnosis. *Multimedia Tools and Applications*, 81(8):10743–10776, March 2022.

[15] Suchet Bargoti and James Underwood. Deep Fruit Detection in Orchards, September 2017.

[16] Siti Bejo, Samihah Mustaffha, Wan Ishak, and Wan Ishak Wan Ismail. Application of Artificial Neural Network in Predicting Crop Yield: A review. *Journal of Food Science and Engineering*, 4:1–9, January 2014.

[17] Latifa Belhaj Salah and Fathi Fourati. A greenhouse Modeling and Control Using Deep Neural Networks. 35(15):1905–1929, 2021.

[18] Lisana Berberi, Valentin Kozlov, Giang Nguyen, Judith Sáinz-Pardo Díaz, Amanda Calatrava, Germán Moltó, Viet Tran, and Álvaro López Garc\ á. Machine Learning Operations Landscape: Platforms and Tools. *Artificial Intelligence Review*, 58(6):167, March 2025.

[19] Edward Bergman, Lennart Purucker, and Frank Hutter. Don't Waste Your Time: Early Stopping Cross-Validation. 2024.

[20] Daniel Berrar. Cross-Validation. January 2018.

[21] Lukas Biewald. Experiment Tracking with Weights and Biases, 2020. Software available from wandb.com.

[22] Bernd Bischl, Martin Binder, Michel Lang, Tobias Pielok, Jakob Richter, Stefan Coors, Janek Thomas, Theresa Ullmann, Marc Becker, Anne-Laure Boulesteix, Difan Deng, and Marius Lindauer. Hyperparameter Optimization: Foundations, Algorithms, Best Practices, and Open Challenges. *WIREs Data Mining and Knowledge Discovery*, 13(2):e1484, 2023.

[23] Vivian B. Brandenburg, Franz Narberhaus, and Axel Mosig. Inverse Folding Based Pre-Training for the Reliable Identification of Intrinsic Transcription Terminators. 18(7):e1010240, 2022.

[24] Mitja Briscik, Gabriele Tazza, László Vidács, Marie-Agnès Dillies, and Sébastien Déjean. Supervised multiple kernel learning approaches for multi-omics data integration. *BioData Mining*, 17(1):1–25, December 2024. Number: 1 Publisher: BioMed Central.

[25] Frank J Bruggeman, Robert Planqué, Douwe Molenaar, and Bas Teusink. Searching for principles of microbial physiology. *FEMS Microbiology Reviews*, 44(6):821–844, November 2020.

[26] Francesco Capozzi and Alessandra Bordoni. Foodomics: A new Comprehensive Approach to Food and Nutrition. *Genes & Nutrition*, 8(1):1–4, January 2013.

[27] Rich Caruana. Multitask Learning. *Machine Learning*, 28(1):41–75, July 1997.

[28] Gavin C Cawley and Nicola L C Talbot. On Over-fittingg inModel Selectionn andSubsequent Selection Biass inPerformance Evaluation. 2010.

[29] Steven W. Chen, Shreyas S. Shivakumar, Sandeep Dcunha, Jnaneshwar Das, Edidiong Okon, Chao Qu, Camillo J. Taylor, and Vijay Kumar. Counting Apples and Oranges With Deep Learning: A data-Driven Approach. *IEEE Robotics and Automation Letters*, 2(2):781–788, April 2017.

[30] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks. In *Proceedings of the 35th International Conference on Machine Learning*, pages 794–803. PMLR, July 2018.

[31] Alejandro Cifuentes. Food Analysis and Foodomics. *Journal of Chromatography A*, 1216(43):7109, October 2009.

[32] Roberto Cipolla, Yarin Gal, and Alex Kendall. Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, Salt Lake City, UT, USA, June 2018. IEEE.

[33] Lisa-Carina Class, Gesine Kuhnen, Sascha Rohn, and Jürgen Kuballa. Diving Deep into the Data: A review of Deep Learning Approaches and Potential Applications in Foodomics. *Foods*, 10(8):1803, August 2021.

[34] ClearML. ClearMl - Your entire Mlops stack in one open-source tool, 2024. Software available from http://github.com/allegroai/clearml.

[35] Antonio Carlos Cob-Parro, Yerhard Lalangui, and Raquel Lazcano. Fostering Agricultural Transformation through Ai: An Open-Source Ai Architecture Exploiting the Mlops Paradigm. 14(2):259, 2024.

[36] Christopher Culley, Supreeta Vijayakumar, Guido Zampieri, and Claudio Angione. A mechanism-aware and multiomic machine-learning pipeline characterizes yeast cell growth. *Proceedings of the National Academy of Sciences*, 117(31):18869–18879, July 2020.

[37] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. *Journal of Scientific Computing*, 92(3), July 2022.

[38] David Dai, Nicholas Horvath, and Jeffrey Varner. Dynamic Sequence Specific Constraint-Based Modeling of Cell-Free Protein Synthesis. *Processes*, 6(8):132, August 2018.

[39] Amit Daniely. Neural Networks Learning and Memorization with (Almost) No Over-Parameterization. In *Advances in Neural Information Processing Systems*, volume 33, pages 9007–9016. Curran Associates, Inc., 2020.

[40] Chandra Mohan Dasari and Raju Bhukya. Explainable Deep Neural Networks for Novel Viral Genome Prediction. 52(3):3002–3017, 2022.

[41] Databricks. MLflow: An Open Source Platform for the Machine Learning Lifecycle. GitHub repository, 2024. Available at `https://github.com/mlflow/mlflow`.

[42] Vadim Demichev, Lukasz Szyrwiel, Fengchao Yu, Guo Ci Teo, George Rosenberger, Agathe Niewienda, Daniela Ludwig, Jens Decker, Stephanie Kaspar-Schoenefeld, Kathryn S. Lilley, Michael Mülleder, Alexey I. Nesvizhskii, and Markus Ralser. Dia-PasEf Data Analysis Using FragPipe and Dia-Nn for Deep Proteomics of Low Sample Amounts. *Nature Communications*, 13(1):3944, July 2022.

[43] Thomas G. Dietterich. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. 10(7):1895–1923.

[44] Peter J. Edwards and Alan F. Murray. A study of Early Stopping and Model Selection Applied to the Papermaking Industry. *International Journal of Neural Systems*, 10(01):9–18, February 2000.

[45] Beyza Eken, Samodha Pallewatta, Nguyen Khoi Tran, Ayse Tosun, and Muhammad Ali Babar. A multivocal Review of Mlops Practices, Challenges and Open Issues, June 2024.

[46] Ibrahim E. Elsemman, Angelica Rodriguez Prado, Pranas Grigaitis, Manuel Garcia Albornoz, Victoria Harman, Stephen W. Holman, Johan Van Heerden, Frank J. Bruggeman, Mark M. M. Bisschops, Nikolaus Sonnenschein, Simon Hubbard, Rob Beynon, Pascale Daran-Lapujade, Jens Nielsen, and Bas Teusink. Whole-cell modeling in yeast predicts compartment-specific proteome constraints that drive metabolic strategies. *Nature Communications*, 13(1):801, February 2022.

[47] Koksal Erenturk, Saliha Erenturk, and Lope G. Tabil. A comparative Study for the Estimation of Dynamical Drying Behavior of Echinacea Angustifolia: Regression Analysis and Neural Network. *Computers and Electronics in Agriculture*, 45(1):71–90, December 2004.

[48] Saliha Erenturk and Koksal Erenturk. Comparison of Genetic Algorithm and Neural Network Approaches for the Drying Process of Carrot. *Journal of Food Engineering*, 78(3):905–912, February 2007.

[49] Axel Escamilla-García, Genaro M. Soto-Zarazúa, Manuel Toledano-Ayala, Edgar Rivas-Araiza, and Abraham Gastélum-Barrios. Applications of Artificial Neural Networks in Greenhouse Technology and Overview for Smart Agriculture Development. 10(11):3835, 2020.

[50] Amir Faghihi, Mohammadreza Fathollahi, and Roozbeh Rajabi. Diagnosis of Skin Cancer Using Vgg16 and Vgg19 Based Transfer Learning Models. 83(19):57495–57510, 2024.

[51] Léon Faure, Bastien Mollet, Wolfram Liebermeister, and Jean-Loup Faulon. A neural-mechanistic hybrid approach improving the predictive power of genome-scale metabolic models. *Nature Communications*, August 2023.

[52] Adam M Feist, Christopher S Henry, Jennifer L Reed, Markus Krummenacker, Andrew R Joyce, Peter D Karp, Linda J Broadbelt, Vassily Hatzimanikatis, and Bernhard Ø Palsson. A genome-scale metabolic reconstruction for Escherichia coli K-12 Mg1655 that accounts for 1260 Orfs and thermodynamic information. *Molecular Systems Biology*, (1):121, January 2007.

[53] Ying Feng, Zhangkai J. Cheng, Xianhu Wei, Moutong Chen, Jumei Zhang, Youxiong Zhang, Liang Xue, Minling Chen, Fan Li, Yuting Shang, Tingting Liang, Yu Ding, and Qingping Wu. Novel Method for Rapid Identification of Listeria Monocytogenes Based on Metabolomics and Deep Learning. *Food Control*, 139:109042, September 2022.

[54] Iván Sánchez Fernández, Edward Yang, Paola Calvachi, Marta Amengual-Gual, Joyce Y. Wu, Darcy Krueger, Hope Northrup, Martina E. Bebin, Mustafa Sahin, Kun-Hsing Yu, Jurriaan M. Peters, and on behalf of the TACERN Study Group. Deep Learning in Rare Disease. Detection of Tubers in Tuberous Sclerosis Complex. *PLOS ONE*, 15(4):e0232376, April 2020.

[55] Anjali Gautam and Vrijendra Singh. CNn-Vsr: A deep Learning Architecture with Validation-Based Stopping Rule for Time Series Classication. *Applied Artificial Intelligence*, 34(2):101–124, January 2020.

[56] Daniel M. Gonçalves, Rui Henriques, and Rafael S. Costa. Predicting metabolic fluxes from omics data via machine learning: Moving from knowledge-driven towards data-driven approaches. *Computational and Structural Biotechnology Journal*, pages 4960–4973, January 2023.

[57] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014.

[58] Timothy Green, Jose Salas, Ana Martinez, and Robert Erskine. Relating Crop Yield to Topographic Attributes Using Spatial Analysis Neural Networks and Regression. 139:23–37, 2007.

[59] Danylo O. Hanchuk and Serhiy O. Semerikov. Automating Machine Learning: A meta-Synthesis of Mlops Tools, Frameworks and Architectures. In *CEUR Workshop Proceedings*, volume 3917, pages 362–414. CEUR-WS.org, 2025.

[60] Ramin Hasibi, Tom Michoel, and Diego A. Oyarzún. Integration of graph neural networks and genome-scale metabolic models for predicting gene essentiality. *npj Systems Biology and Applications*, 10(1), March 2024.

[61] Sture Holm. A simple Sequentially Rejective Multiple Test Procedure. 6(2):65–70.

[62] Hanzhang Hu, Debadeepta Dey, Martial Hebert, and J. Andrew Bagnell. Learning Anytime Predictions in Neural Networks via Adaptive Loss Balancing, May 2018.

[63] Erik D. Huckvale, Christian D. Powell, Huan Jin, and Hunter N.B. Moseley. Benchmark Dataset for Training Machine Learning Models to Predict the Pathway Involvement of Metabolites. page 2023.10.03.560715.

[64] Tahir Hussain and Hayaru Shouno. MAgres-Unet: Improved Medical Image Segmentation Through a Deep Learning Paradigm of Multi-Attention Gated Residual U-net. *IEEE Access*, 12:40290–40310, 2024.

[65] Tahir Hussain, Hayaru Shouno, Abid Hussain, Dostdar Hussain, Muhammad Ismail, Tatheer Hussain Mir, Fang Rong Hsu, Taukir Alam, and Shabnur Anonna Akhy. EFfresNet-Vit: A fusion-Based Convolutional and Vision Transformer Model for Explainable Medical Image Classification. *IEEE Access*, 13:54040–54068, 2025.

[66] Tahir Hussain, Hayaru Shouno, Mazin Abed Mohammed, Haydar Abdulameer Marhoon, and Taukir Alam. DCssGa-Unet: Biomedical Image Segmentation with DenseNet Channel Spatial and Semantic Guidance Attention. *Knowledge-Based Systems*, 314:113233, April 2025.

[67] Nobuyoshi Ishii, Kenji Nakahigashi, Tomoya Baba, Martin Robert, Tomoyoshi Soga, Akio Kanai, Takashi Hirasawa, Miki Naba, Kenta Hirai, Aminul Hoque, Pei Yee Ho, Yuji Kakazu, Kaori Sugawara, Saori Igarashi, Satoshi Harada, Takeshi Masuda, Naoyuki Sugiyama, Takashi Togashi, Miki Hasegawa, Yuki Takai, Katsuyuki Yugi, Kazuharu Arakawa, Nayuta Iwata, Yoshihiro Toya, Yoichi Nakayama, Takaaki Nishioka, Kazuyuki Shimizu, Hirotada Mori, and Masaru Tomita. Multiple High-Throughput Analyses Monitor the Response of E. coli to Perturbations. *Science*, 316(5824):593–597, April 2007.

[68] Haider Khalaf Jabbar and Rafiqul Zaman Khan. Methods to Avoid Over-Fitting and Under-Fitting in Supervised Machine Learning (Comparative Study). In *Computer Science, Communication and Instrumentation Devices*, pages 163–172. Research Publishing Services, 2014.

[69] Gareth James. An Introduction to Statistical Learning: With Applications in R - Section: 2, Statistical Learning. In *An Introduction to Statistical Learning: With Applications in R*. SPRINGER, 1st edition edition.

[70] Ziwei Ji, Justin Li, and Matus Telgarsky. Early-Stopped Neural Networks Are Consistent. In *Advances in Neural Information Processing Systems*, volume 34, pages 1805–1817. Curran Associates, Inc., 2021.

[71] Yiru Jiang, Jing Luo, Danqing Huang, Ya Liu, and Dan-dan Li. Machine Learning Advances in Microbiology: A review of Methods and Applications. *Frontiers in Microbiology*, 13, 2022.

[72] Katherine W. Jordan, Shichen Wang, Fei He, Shiaoman Chao, Yanni Lun, Etienne Paux, Pierre Sourdille, Jamie Sherman, Alina Akhunova, Nancy K. Blake, Michael O. Pumphrey, Karl Glover, Jorge Dubcovsky, Luther Talbert, and Eduard D. Akhunov. The Genetic Architecture of Genome-Wide Recombination Rate Variation in Allopolyploid Wheat Revealed by Nested Association Mapping. 95(6):1039–1054.

[73] Minseung Kim, Navneet Rai, Violeta Zorraquino, and Ilias Tagkopoulos. Multi-omics integration accurately predicts cellular state in unexplored conditions for Escherichia coli. *Nature Communications*, 7(1), October 2016.

[74] Zachary A. King, Justin Lu, Andreas Dräger, Philip Miller, Stephen Federowicz, Joshua A. Lerman, Ali Ebrahim, Bernhard O. Palsson, and Nathan E. Lewis. BigG models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Research*, 44(D1):D515–D522, January 2016.

[75] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[76] Sebastian Kujawa and Gniewko Niedbała. Artificial Neural Networks in Agriculture. 11(6):497, 2021.

[77] Ruslan Kuprieiev, skshetry, Peter Rowlands, Dmitry Petrov, Paweł Redzyński, David de la Iglesia Castro, Alexander Schepanovski, Ivan Shcheklein, Gao, Batuhan Taskaya, Jorge Orpinel, Fábio Santos, Daniele, Ronan Lamy, Aman

Sharma, Zhanibek Kaimuldenov, Dani Hodovic, Nikita Kodenko, Andrew Grigorev, Earl, Nabanita Dash, George Vyshnya, Dave Berenbaum, maykulkarni, Max Hora, Vera, and Sanidhya Mangal. DVc: Data Version Control - Git for Data & Models, 2024.

[78] Tuija Leinonen, David Wong, Antti Vasankari, Ali Wahab, Ramesh Nadarajah, Matti Kaisti, and Antti Airola. Empirical Investigation of Multi-Source Cross-Validation in Clinical Ecg Classification. 183:109271, 2024.

[79] Nathan E Lewis, Kim K Hixson, Tom M Conrad, Joshua A Lerman, Pep Charusanti, Ashoka D Polpitiya, Joshua N Adkins, Gunnar Schramm, Samuel O Purvine, Daniel Lopez-Ferrer, Karl K Weitz, Roland Eils, Rainer König, Richard D Smith, and Bernhard Ø Palsson. Omic data from evolved \textitE. coli are consistent with computed optimal growth from genome-scale models. *Molecular Systems Biology*, 6(1):390, January 2010.

[80] Petro Liashchynskyi and Pavlo Liashchynskyi. Grid Search, Random Search, Genetic Algorithm: A big Comparison for Nas, December 2019.

[81] R. Linker, I. Seginer, and P. O. Gutman. Optimal Co2 Control in a Greenhouse Modeled with Neural Networks. *Computers and Electronics in Agriculture*, 19(3):289–310, March 1998.

[82] Jian Liu, Lixia Liu, Wei Guo, Minglang Fu, Minli Yang, Shengxiong Huang, Feng Zhang, and Yongsheng Liu. A new Methodology for Sensory Quality Assessment of Garlic Based on Metabolomics and an Artificial Neural Network. *RSC Advances*, 9(31):17754–17765, June 2019.

[83] Qi Liu, Shi-min Zuo, Shasha Peng, Hao Zhang, Ye Peng, Wei Li, Yehui Xiong, Runmao Lin, Zhiming Feng, Huihui Li, Jun Yang, Guo-Liang Wang, and Houxiang Kang. Development of Machine Learning Methods for Accurate Prediction of Plant Disease Resistance. 40:100–110, 2024.

[84] Shuying Liu and Weihong Deng. Very Deep Convolutional Neural Network Based Image Classification Using Small Training Sample Size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734, November 2015.

[85] Aleksander Lodwich, Yves Rangoni, and Thomas Breuel. Evaluation of Robustness and Performance of Early Stopping Rules with Multi Layer Perceptrons. In *2009 International Joint Conference on Neural Networks*, pages 1877–1884, June 2009.

[86] Vanda~ M. Lourenço, Joseph~ O. Ogutu, Rui~ A.P. Rodrigues, Alexandra Posekany, and Hans-Peter Piepho. Genomic Prediction Using Machine Learning: A comparison of the Performance of Regularized Regression, Ensemble, Instance-Based and Deep Learning Methods on Synthetic and Empirical Data. 25(1):152.

[87] Dennis N. Lozada, Karansher Singh Sandhu, and Madhav Bhatta. Ridge Regression and Deep Learning Models for Genome-Wide Selection of Complex Traits in New Mexican Chile Peppers. 24(1):80.

[88] Renfu Lu. Nondestructive Measurement of Firmness and Soluble Solids Content for Apple Fruit Using Hyperspectral Scattering Images. *Sensing and Instrumentation for Food Quality and Safety*, 1(1):19–27, March 2007.

[89] Luis Lugo and Emiliano Barreto Hernández. A recurrent Neural Network Approach for Whole Genome Bacteria Identification. *Applied Artificial Intelligence*, 35(9):642–656, July 2021.

[90] Scott M Lundberg and Su-In Lee. A unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[91] Peihua Ma, Zhikun Zhang, Xiaoxue Jia, Xiaoke Peng, Zhi Zhang, Kevin Tarwa, Cheng-I Wei, Fuguo Liu, and Qin Wang. Neural Network in Food Analytics. *Critical Reviews in Food Science and Nutrition*, 0(0):1–19, 2022.

[92] Daniel Machado and Markus Herrgård. Systematic Evaluation of Methods for Integration of Transcriptomic Data into Constraint-Based Models of Metabolism. *PLOS Computational Biology*, 10(4):e1003580, April 2014.

[93] Giuseppe Magazzù, Guido Zampieri, and Claudio Angione. Multimodal regularized linear models with flux balance analysis for mechanistic integration of omics data. *Bioinformatics*, 37(20):3546–3552, May 2021.

[94] Vivien Marquard, Lars Beckmann, Iris M. Heid, Claudia Lamina, and Jenny Chang-Claude. Impact of Genotyping Errors on the Type I error Rate and the Power of Haplotype-Based Association Methods. *BMC Genetics*, 10(1):3, January 2009.

[95] F. Mateo, R. Gadea, R. Mateo, A. Medina, F. Valle-Algarra, and M. Jiménez. Neural Network Models for Prediction of Trichothecene Content in Wheat. *World Mycotoxin Journal*, 1(3):349–356, August 2008.

[96] Beatriz M. A. Matsui and Denise H. Goya. MLops: A guide to Its Adoption in the Context of Responsible Ai. In *2022 IEEE/ACM 1st International Workshop on Software Engineering for Responsible Artificial Intelligence (SE4RAI)*, pages 45–49, May 2022.

[97] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2, 2014.

[98] Tamás Miseta, Attila Fodor, and Ágnes Vathy-Fogarassy. Surpassing Early Stopping: A novel Correlation-Based Stopping Criterion for Neural Networks. *Neurocomputing*, 567:127028, January 2024.

[99] Sharada P. Mohanty, David P. Hughes, and Marcel Salathé. Using Deep Learning for Image-Based Plant Disease Detection. 7, 2016.

[100] Jonathan Monk, Juan Nogales, and Bernhard O. Palsson. Optimizing genome-scale network reconstructions. *Nature Biotechnology*, 32(5):447–452, May 2014. Publisher: Nature Publishing Group.

[101] Jonathan M. Monk, Colton J. Lloyd, Elizabeth Brunk, Nathan Mih, Anand Sastry, Zachary King, Rikiya Takeuchi, Wataru Nomura, Zhen Zhang, Hirotada Mori, Adam M. Feist, and Bernhard O. Palsson. iMl1515, a knowledgebase that computes Escherichia coli traits. *Nature Biotechnology*, 35(10):904–908, October 2017. Number: 10 Publisher: Nature Publishing Group.

[102] Osval A. Montesinos-López, Abelardo Montesinos-López, Roberto Tuberosa, Marco Maccaferri, Giuseppe Sciara, Karim Ammar, and José Crossa. Multi-Trait, Multi-Environment Genomic Prediction of Durum Wheat With Genomic Best Linear Unbiased Predictor and Deep Learning Methods. *Frontiers in Plant Science*, 10, 2019.

[103] James Morrissey, Gianmarco Barberi, Benjamin Strain, Pierantonio Facco, and Cleo Kontoravdi. NExt-Fba: A hybrid stoichiometric/data-driven approach to improve intracellular flux predictions. *Metabolic Engineering*, March 2025.

[104] Swathi Sirisha Nallan Chakravartula, Roberto Moscetti, Giacomo Bedini, Marco Nardella, and Riccardo Massantini. Use of Convolutional Neural Network (Cnn) Combined with Ft-Nir Spectroscopy to Predict Food Adulteration: A case Study on Coffee. *Food Control*, 135:108816, May 2022.

[105] Vu Nguyen. Bayesian Optimization for Accelerating Hyper-Parameter Tuning. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 302–305, June 2019.

[106] Pierfrancesco Novielli, Donato Romano, Stefano Pavan, Pasquale Losciale, Anna Maria Stellacci, Domenico Diacono, Roberto Bellotti, and Sabina Tangaro. Explainable Artificial Intelligence for Genotype-to-Phenotype Prediction in Plant Breeding: A case Study with a Dataset from an Almond Germplasm Collection. 15.

[107] Min Oh and Liqing Zhang. DeepMicro: Deep Representation Learning for Disease Prediction Based on Microbiome Data. *Scientific Reports*, 10(1):6026, December 2020.

[108] Brett Olivier, Willi Gottstein, Douwe Molenaar, and Bas Teusink. CBmpy release 0.8.4, February 2023.

[109] Tobias Hegelund Olsen, Betül Yesiltas, Frederikke Isa Marin, Margarita Pertseva, Pedro J. García-Moreno, Simon Gregersen, Michael Toft Overgaard, Charlotte Jacobsen, Ole Lund, Egon Bech Hansen, and Paolo Marcatili. AnoxPepred: Using Deep Learning for the Prediction of Antioxidative Properties of Peptides. *Scientific Reports*, 10(1):21471, December 2020.

[110] Jeffrey D. Orth, R. M. T. Fleming, and Bernhard Ø. Palsson. Reconstruction and Use of Microbial Metabolic Networks: the Core Escherichia coli Metabolic Model as an Educational Guide. *EcoSal Plus*, 4(1), February 2010.

[111] Jeffrey D. Orth, Ines Thiele, and Bernhard Ø Palsson. What is flux balance analysis? *Nature Biotechnology*, 28(3):245–248, March 2010. Number: 3 Publisher: Nature Publishing Group.

[112] M. O' Farrell, E. Lewis, C. Flanagan, W.B. Lyons, and N. Jackman. Design of a System That Uses Optical-Fiber Sensors and Neural Networks to Control a Large-Scale Industrial Oven by Monitoring the Food Quality Online. *IEEE Sensors Journal*, 5(6):1407–1420, December 2005.

[113] Opeoluwa Oyedele. Determining the Optimal Number of Folds to Use in a K-fold Cross-Validation: A neural Network Classification Experiment. *Research in Mathematics*, 10, May 2023.

[114] Edward J. O'Brien, Jonathan M. Monk, and Bernhard O. Palsson. Using Genome-scale Models to Predict Biological Capabilities. *Cell*, 161(5):971–987, May 2015.

[115] B. O. Palsson. Systems Biology: Constraint-Based Reconstruction and Analysis, 2015.

[116] J. Paquet, C. Lacroix, and J. Thibault. Modeling of pH and Acidity for Industrial Cheese Production. *Journal of Dairy Science*, 83(11):2393–2409, November 2000.

[117] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[118] Marius-Constantin Popescu, Valentina Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer Perceptron and Neural Networks. *WSEAS Transactions on Circuits and Systems*, 8, July 2009.

[119] Kalari Prakash, Kanaparthi Sai Sreenidhi Harsha, Puchakayala Sai Jeevan, Lekshmi R. Chandran, and Angel T.S. Machine Learning-Based Crop Prediction: A way Towards Smart Farming. In *2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pages 623–628, 2022.

[120] Lulu Qi, Jialuo Du, Yue Sun, Yongzhao Xiong, Xinyao Zhao, Daodong Pan, Yueru Zhi, Yali Dang, and Xinchang Gao. Umami-MrnN: Deep Learning-Based Prediction of Umami Peptide Using Rnn and Mlp. *Food Chemistry*, 405:134935, March 2023.

[121] Redwan Ahmed Rizvee, Tasnim Hossain Orpa, Adil Ahnaf, Md Ahsan Kabir, Mohammad Rifat Ahmmad Rashid, Mohammad Manzurul Islam, Maheen Islam, Taskeed Jabid, and Md Sawkat Ali. LeafNet: A proficient Convolutional Neural Network for Detecting Seven Prominent Mango Leaf Diseases. 14:100787, 2023.

[122] D. Ruggeri and L. Vidács. INtrOduCinG mLopS tO fAciLitAte RepRodUciBle ModEl DevEloPmeNt On OmiCs DatA. pages 198–203, 2024.

[123] Inkyu Sa, Zongyuan Ge, Feras Dayoub, Ben Upcroft, Tristan Perez, and Chris McCool. DeepFruits: A fruit Detection System Using Deep Neural Networks. 16(8):1222, 2016.

[124] Ankur Sahu, Mary-Ann Blätke, Jędrzej Jakub Szymański, and Nadine Töpfer. Advances in flux balance analysis by integrating machine learning and mechanism-based models. *Computational and Structural Biotechnology Journal*, 19:4626–4640, 2021.

[125] Karansher Sandhu, Shruti Sunil Patil, Michael Pumphrey, and Arron Carter. Multitrait Machine- and Deep-Learning Models for Genomic Selection Using Spectral Information in a Wheat Breeding Program. *The Plant Genome*, 14(3):e20119, 2021.

[126] Karansher S. Sandhu, Dennis N. Lozada, Zhiwu Zhang, Michael O. Pumphrey, and Arron H. Carter. Deep Learning for Predicting Complex Traits in Spring Wheat Breeding Program. *Frontiers in Plant Science*, 11, 2021.

[127] Karansher Singh Sandhu, Meriem Aoun, Craig F. Morris, and Arron H. Carter. Genomic Selection for End-Use Quality and Processing Traits in Soft White Winter Wheat Breeding Program with Machine and Deep Learning Models. *Biology*, 10(7):689, July 2021.

[128] SeldonIO. Seldon Core: Open Source Mlops Framework for Deploying Machine Learning Models. GitHub repository, 2024. Available at `https://github.com/SeldonIO/seldon-core`.

[129] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 525–536, Red Hook, NY, USA, December 2018. Curran Associates Inc.

[130] Yang Shao, Gregory N. Taff, and Stephen J. Walsh. Comparison of Early Stopping Criteria for Neural-Network-Based Subpixel Classification. *IEEE Geoscience and Remote Sensing Letters*, 8(1):113–117, January 2011.

[131] Alex Sherstinsky. Fundamentals of Recurrent Neural Network (Rnn) and Long Short-Term Memory (LstM) Network. *Physica D: Nonlinear Phenomena*, 404:132306, March 2020.

[132] Vincent Somerville, Pranas Grigaitis, Julius Battjes, Francesco Moro, and Bas Teusink. Use and limitations of genome-scale metabolic models in food microbiology. *Current Opinion in Food Science*, 43:225–231, February 2022.

[133] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[134] Alexander Statnikov, Lily Wang, and Constantin F Aliferis. A comprehensive Comparison of Random Forests and Support Vector Machines for Microarray-Based Cancer Classification. *BMC Bioinformatics*, 9:319, July 2008.

[135] Gopaiah Talari, Rajat Nag, John O' Brien, Cronan McNamara, and Enda Cummins. A data-Driven Approach for Prioritising Microbial and Chemical Hazards Associated with Dairy Products Using Open-Source Databases. *Science of The Total Environment*, 908:168456, January 2024.

[136] Gabriele Tazza, Francesco Moro, Bas Teusink, and László Vidács. MEtaBolIc-InfOrmEd NeuRal NetWorK fOr MulTi-OmiCs DatA iNteGraTioN. In Jan F.M. Van Impe and Monika E. Pola\ 'nska, editors, *13th International Conference on Simulation and Modelling in the Food and Bio-Industry (FOODSIM 2024)*, pages 193–197. Eurosis-ETI, 2024. Publisher Copyright: ©2024, EUROSIS-ETI. All rights reserved.; 13th International Conference on Simulation and Modelling in the Food and Bio-Industry, FOODSIM 2024 ; Conference date: 07-04-2024 Through 11-04-2024.

[137] Y. Uno, S. O. Prasher, R. Lacroix, P. K. Goel, Y. Karimi, A. Viau, and R. M. Patel. Artificial Neural Networks to Predict Corn Yield from Compact Airborne Spectrographic Imager Data. 47(2):149–161, 2005.

[138] Rob van der Goot. We Need to Talk About Train-Dev-Test Splits. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4485–4494, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[139] Eunice van Pelt-KleinJan, Daan H. de Groot, and Bas Teusink. Understanding Fba Solutions under Multiple Nutrient Limitations. *Metabolites*, 11(5):257, May 2021. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

[140] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2023.

[141] Belén Vega-Márquez, Isabel Nepomuceno-Chamorro, Natividad Jurado-Campos, and Cristina Rubio-Escudero. Deep Learning Techniques to Improve the Performance of Olive Oil Classification. *Frontiers in Chemistry*, 7, 2020.

[142] Manuel Vilares Ferro, Yerai Doval Mosquera, Francisco J. Ribadas Pena, and Víctor M. Darriba Bilbao. Early Stopping by Correlating Online Indicators in Neural Networks. *Neural Networks*, 159:109–124, February 2023.

[143] Chandrasekar Vuppalapati, Anitha Ilapakurti, Karthik Chillara, Sharat Kedari, and Vanaja Mamidi. Automating Tiny Ml Intelligent Sensors DevOps Using Microsoft Azure. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 2375–2384, 2020.

[144] Kelin Wang, Muhammad Ali Abid, Awais Rasheed, Jose Crossa, Sarah Hearne, and Huihui Li. DNngP, a Deep Neural Network-Based Method for Genomic Prediction Using Multi-Omics Data in Plants. *Molecular Plant*, 16(1):279–293, January 2023.

[145] Peipei Wang, Melissa D. Lehti-Shiu, Serena Lotreck, Kenia Segura Abá, Patrick J. Krysan, and Shin-Han Shiu. Prediction of Plant Complex Traits via Integration of Multi-Omics Data. 15(1):6856.

[146] W. Wang and J. Paliwal. Generalisation Performance of Artificial Neural Networks for Near Infrared Spectral Analysis. *Biosystems Engineering*, 94(1):7–18, May 2006.

[147] Yi Wang, Yihang Feng, Zhenlei Xiao, and Yangchao Luo. Machine Learning Supported Single-Stranded Dna Sensor Array for Multiple Foodborne Pathogenic and Spoilage Bacteria Identification in Milk. 463:141115, 2025.

[148] Samar Wazir, Gautam Siddharth Kashyap, and Parag Saxena. MLops: A review, August 2023.

[149] R. F. Woolson. Wilcoxon Signed-Rank Test. In *Wiley Encyclopedia of Clinical Trials*, pages 1–3. John Wiley & Sons, Ltd.

[150] Libin Wu, Shenghang Zhang, Haiyong Weng, Shangpeng Sun, and Dapeng Ye. Integrating Micro-Hyperspectral Imaging and Machine Learning to Investigate Mie Scattering for Early Detection of Microbial Contamination in Liquid Fermentation Cultures. 257:118620, 2025.

[151] Thomas P. Wytock and Adilson E. Motter. Predicting growth rate from gene expression. *Proceedings of the National Academy of Sciences*, 116(2):367–372, December 2018.

[152] Yun Xu and Royston Goodacre. On Splitting Training and Validation Set: A comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning. *Journal of Analysis and Testing*, 2(3):249–262, July 2018.

[153] Pradeep Kumar Yadalam, Raghavendra Vamsi Anegundi, Prabhu Manickam Natarajan, and Carlos M. Ardila. Neural Networks for Predicting and Classifying Antimicrobial Resistance Sequences in \emphPorphyromonas\emph Gingivalis. 75(5):100890, 2025.

[154] Sanjay Yadav and Sanyam Shukla. Analysis of K-fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification. In *2016*

IEEE 6th International Conference on Advanced Computing (IACC), pages 78–83, February 2016.

[155] YAN YANG. A model Transfer Learning Framework With Back-Propagation Neural Network for Wine and Chinese Liquor Detection by Electronic Nose \textbar IeeE journals & Magazine \textbar IeeE xplore. https://ieeexplore.ieee.org/abstract/document/9107133, June 2020.

[156] Suqin Yuan, Runqi Lin, Lei Feng, Bo Han, and Tongliang Liu. Instance-Dependent Early Stopping, February 2025.

[157] Guido Zampieri, Supreeta Vijayakumar, Elisabeth Yaneske, and Claudio Angione. Machine and deep learning meet genome-scale metabolic modeling. *PLoS Computational Biology*, 15(7), July 2019.

[158] Wenjun Zhang, ChangJun Bai, and GuoDao Liu. Neural Network Modeling of Ecosystems: A case Study on Cabbage Growth System. 201:317–325, 2007.

[159] Yongli Zhang and Yuhong Yang. Cross-Validation for Selecting a Model Selection Procedure. *Journal of Econometrics*, 187(1):95–112, July 2015.

[160] Xianglin Zhu, Khalil Ur Rehman, Wang Bo, Muhammad Shahzad, and Ahmad Hassan. Data-Driven Soft Sensor Model Based on Deep Learning for Quality Prediction of Industrial Processes. *SN Computer Science*, 2(1):40, January 2021.

Summary

This PhD thesis examines methodological and applied challenges that arise when using Machine Learning (ML) and Deep Learning (DL) models for omics data analysis, with particular attention to high-dimensional and data-scarce settings common in agricultural and biological research. The work addresses issues related to evaluation reliability, training stability, interpretability, and organization of experimental workflows, combining methodological investigations with applied case studies in foodomics and agronomy.

The dissertation is structured into three main parts. Chapter 1 introduces the background and motivation, outlining the challenges associated with applying ML to omics data and defining the overarching research objectives. Chapters 2 and 3 present the core scientific contributions, grouped into two complementary research directions.

Work I. Methodological Contributions to Machine Learning for Omics Data

The first group of contributions focuses on methodological aspects that influence how reliably DL models can be trained and interpreted in omics applications. Chapter 2 investigates evaluation strategies and training procedures for Neural Network (NN) in small, high-dimensional datasets, together with hybrid modeling approaches that combine data-driven learning with mechanistic structure.

Two studies are presented. The first provides a systematic analysis of how Cross-Validation (CV) and Early Stopping (ES) are combined in foodomics NN research. It highlights common pitfalls such as data leakage and biased performance estimates, and proposes practical recommendations to support more reliable evaluation protocols. The second study examines hybrid and metabolic-informed NN models from a multitask learning perspective, analyzing how data-driven and mechanistic objectives interact during optimization and how loss formulation and task balancing affect performance, stability, and interpretability.

Together, these investigations offer methodological insights and practical guidance for designing, training, and evaluating DL models in omics settings.

Work II. Applied Deep Learning and Experiment Management in Agronomy

The second group of contributions focuses on applied DL models and experiment management practices in agronomic and foodomics research. Chapter 3 presents two applied studies that address predictive performance, interpretability, and reproducibility in real-world omics scenarios.

The first study develops an explainable NN framework for wheat Single-Nucleotide Polymorphism (SNP) analysis. By integrating modern DL practices - including regularization, data augmentation, and Bayesian hyper-parameter optimization - the model achieves statistically significant improvements over previously reported results on a public wheat SNP dataset. Explainable Explainable Artificial Intelligence (XAI) techniques are further used to identify influential SNP markers and support transparent genotype-to-phenotype interpretation.

The second study explores the role of Machine Learning Operations (MLOps) frameworks for organizing research-stage ML workflows in agronomy. Through a detailed case study, it demonstrates the integration of experiment tracking, dataset versioning, pipeline automation, and containerization, showing how such practices improve reproducibility, transparency, and scalability even before model deployment.

Contributions of the Thesis

In the **first group**, the contributions concern methodological aspects of ML for omics data, with emphasis on evaluation protocols, training strategies, and hybrid learning formulations (see Section 2.6):

II/1. Systematic analysis of CV–ES integration in foodomics NN studies, identifying methodological pitfalls and sources of bias.

II/2. Practical recommendations for combining ES and CV to improve reliability and comparability of performance estimates in small-sample omics settings.

II/3. Methodological investigation of hybrid and metabolic-informed NN from a multitask learning perspective, focusing on objective balancing and training stability.

II/4. Empirical evaluation of loss formulations and training strategies for hybrid models integrating data-driven and mechanistic objectives.

In the **second group**, the contributions relate to applied DL and experiment management in agronomy and foodomics (see Section 3.6):

III/1. Design and development of an applied DL framework for wheat SNP data analysis, integrating modern training practices to improve predictive performance and robustness.

III/2. Demonstration of statistically significant performance improvements for genotype-to-phenotype prediction using NN models on a publicly available dataset.

III/3. Integration of XAI methods to enable feature-level interpretation of DL predictions in omics data.

III/4. Demonstration of MLOps frameworks as tools for structuring and managing research-stage ML experiments, enhancing reproducibility and workflow transparency in agronomy.

Összefoglalás

A doktori értekezés a gépi tanulási (ML) és mélytanulási (DL) modellek omikai adatok elemzésében történő alkalmazási és módszertani kihívásait vizsgálja, különös tekintettel azokra a mezőgazdasági és biológiai kutatásokra, ahol alapvetően nagy dimenziószámú és kis elemszámú adatok állnak rendelkezésre. A munka a modellek értékelésének megbízhatóságával, a tanítás stabilitásával, a predikciók magyarázhatóságával, valamint a kísérleti munkafolyamatok szervezésével kapcsolatos kérdéseket tárgyalja, módszertani vizsgálatokat ötvözve élelmiszer-omikai és agronómiai esettanulmányokkal. Az értekezés három fő részre tagolódik. Az 1. fejezet bemutatja az elméleti hátteret és a kutatás motivációját, ismerteti az ML omikai adatokon történő alkalmazásának főbb kihívásait, valamint meghatározza az átfogó kutatási célokat. A 2. és 3. fejezet a legfontosabb tudományos eredményeket foglalja össze, két egymást kiegészítő kutatási irány mentén.

1. rész

Az eredmények első csoportja azokra a módszertani szempontokra összpontosít, amelyek meghatározzák, hogy a mélytanulási modellek mennyire megbízhatóan taníthatók és magyarázhatók omikai alkalmazásokban. A 2. fejezet a neurális hálózatok (NN) értékelési stratégiáit és tanítási eljárásait vizsgálja kis elemszámú, nagy dimenziójú adathalmazokon, továbbá olyan hibrid modellezési megközelítéseket elemez, amelyek az adatvezérelt tanulást mechanisztikus struktúrákkal ötvözik. A fejezet két tanulmányt mutat be. Az első szisztematikus elemzést ad arról, miként alkalmazzák együttesen élelmiszer-omikai neuronháló-kutatásokban a keresztvalidációt (CV) és a tanítás korai leállítását (ES). A tanulmány egyfelől rávilágít a szakirodalomban tapasztalható gyakori módszertani hibákra, mint például az adatszivárgás és a torzított teljesítménybecslés, másfelől gyakorlati ajánlásokat fogalmaz meg a megbízhatóbb értékelési protokollok kialakítására. A második tanulmány a hibrid és metabolikus információkat hasznosító neuronháló modelleket a többfeladatos tanulás (multitask learning) szempontjából vizsgálja. A kutatás bemutatja az adatvezérelt és a mechanisztikus célfüggvények optimalizálás során tapasztalható kölcsönhatását, továbbá vizsgálja, hogy a veszteségfüggvény definiálása és a tanítási feladatok közötti egyensú-

lyozás miként befolyásolja a teljesítményt, a stabilitást és az értelmezhetőséget. Ezek a vizsgálatok együttesen módszertani betekintést és gyakorlati útmutatást nyújtanak a DL modellek tervezéséhez, tanításához és értékeléséhez omikai környezetben.

2. rész

Az eredmények második csoportja az alkalmazott DL modellekre és a kísérletmenedzsment gyakorlatára fókuszál agronómiai és élelmiszer-omikai kutatásokban. A 3. fejezet két tanulmányt ismertet, amelyek a prediktív teljesítmény, a magyarázhatóság és a reprodukálhatóság kérdéseit tárgyalják valós omikai problémákon. Az első tanulmány egy magyarázható neuronháló-keretrendszert dolgoz ki búza egynukleotidos polimorfizmusainak (SNP) elemzésére. A korszerű mélytanulási eljárások (köztük a regularizáció, az adataugmentáció és a Bayes-féle hiperparaméter-optimalizálás) integrálásával a modell statisztikailag szignifikáns javulást ér el a szakirodalomban korábban közölt eredményekhez képest egy nyilvános búza SNP adathalmazon. A magyarázható mesterséges intelligencia (XAI) technikák alkalmazása lehetővé teszi a meghatározó SNP markerek azonosítását, támogatva a genotípus és fenotípus közötti kapcsolat átlátható értelmezését. A második tanulmány a gépi tanulási műveleteket támogató (MLOps) keretrendszerek szerepét vizsgálja a kutatási fázisban lévő ML munkafolyamatok szervezésében az agronómia területén. Részletes esettanulmányon keresztül mutatja be a kísérletkövetés, az adatverzió-kezelés, a folyamatautomatizálás és a konténerizáció integrációját, ezzel szemléltetve ezeknek a gyakorlatoknak a reprodukálhatóságra, átláthatóságra és skálázhatóságra gyakorolt pozitív hatását.

Az értekezés tudományos eredményei

Az első eredménycsoport az omikai adatokon alkalmazott gépi tanulás módszertani kérdéseit tárgyalja, különös tekintettel az értékelési protokollokra, a tanítási stratégiákra és a hibrid tanulási megközelítésekre (lásd 2.6. alfejezet):

I/1. A keresztvalidáció (CV) és a korai leállás (ES) integrációjának szisztematikus elemzése élelmiszer-omikai neurálishálózat-kutatásokban, a módszertani hibák és a torzítási források azonosításával.

I/2. Gyakorlati ajánlások kidolgozása a korai leállás és a keresztvalidáció kombinálására a teljesítménybecslés megbízhatóságának és összehasonlíthatóságának javítása érdekében kis elemszámú omikai adatkörnyezetben.

I/3. Hibrid és metabolikus információkat hasznosító neurális hálózatok módszertani vizsgálata a többfeladatos tanulás (multitask learning) perspektívájából, különös tekintettel a célfüggvények súlyozására és a tanítási stabilitásra.

I/4. Veszteségfüggvény-formulák és tanítási stratégiák empirikus értékelése az adatvezérelt és mechanisztikus célokat integráló hibrid modellek esetében.

A második eredménycsoport az alkalmazott mélytanulással és a kísérletmenedzsmenttel foglalkozik az agronómia és a élelmiszer-omika területén (lásd 3.6. alfejezet):

II/1. Alkalmazott mélytanulási keretrendszer tervezése és fejlesztése búza SNP-adatok elemzésére, amely a prediktív teljesítmény és a robusztusság növelése érdekében modern tanítási eljárásokat integrál.

II/2. Statisztikailag szignifikáns teljesítményjavulás elérése a genotípus-fenotípus előrejelzésben neurálishálózat-modellek alkalmazásával, nyilvánosan elérhető adathalmazon.

II/3. Magyarázható mesterséges intelligencia (XAI) módszerek integrálása a mélytanulási előrejelzések jellemző-szintű (feature-level) értelmezésének lehetővé tételére omikai adatok esetében.

II/4. Az MLOps keretrendszerek alkalmazhatóságának bemutatása a kutatási fázisban lévő gépi tanulási kísérletek strukturálására és kezelésére, növelve ezzel az agronómiai ML kutatások reprodukálhatóságát és átláthatóságát.

Publications

Journal publications

[1] **D. Ruggeri** and L. Vidács. Advancing Wheat Single-Nucleotide Polymorphism Data Analysis with Explainable Deep Learning Models. *Applied Artificial Intelligence*, 39(1):2565169, 2025.

[2] **D. Ruggeri**, G. Tazza, and L. Vidács. Introducing MLOps to Facilitate the Development of Machine Learning Models in Agronomy: A Case Study. *IEEE Access*, 13:122059–122070, 2025.

[3] **D. Ruggeri** and L. Vidács. KFold Cross-Validation and Early Stopping in Foodomics Neural Networks: Practices, Pitfalls, and Recommendations. *IEEE Access*, 13:190820–190832, 2025.

[4] **G. Tazza**, D. Ruggeri, and L. Vidács. Improving Microbiome-Based Disease Prediction With SuperTML and Data Augmentation. *IEEE Access*, 13:144505–144515, 2025.

[5] **G. Tazza**, F. Moro, D. Ruggeri, B. Teusink, and L. Vidács. MINN: A Metabolic-Informed Neural Network for Integrating Omics Data into Genome-Scale Metabolic Modeling. *Computational and Structural Biotechnology Journal*, 27:3609–3617, 2025.

Full papers in conference proceedings

[6] **D. Ruggeri** and L. Vidács. Introducing MLOps to Facilitate Reproducible Model Development on Omics Data. In *Proceedings of the 13th International Conference on Simulation and Modelling in the Food and Bio-Industry (FOODSIM)*, pp. 198–203, 2024.

Acknowledgments

First of all, I would like to thank my supervisor, László Vidács, for directing my PhD studies, as well as Péter Pusztai for his constant support.

I am deeply grateful to Gabriele Tazza, who has been both a colleague and a dear friend. Sharing these years in Szeged with you has been an unforgettable experience, both professionally and personally. My thanks also go to my colleagues from the E-MUSE ITN – Alina Kyrylenko, Francesco Moro, Mitja Briscik, Geoffrey Roudaut, Rina Mekuli, Michèle Bou Habib, and Teresa Osset, among many others – for the inspiring collaborations, the stimulating discussions, and the many wonderful travels and moments we shared along the way. I would also like to thank Federico Cortese for the unexpected and intellectually stimulating collaboration we started.

I am especially thankful to my girlfriend, Dalila Amodio, for always being by my side, for her patience, and for her constant presence throughout this journey. My deepest gratitude goes to my family – Mariella Dolci and Domenico Ruggeri – for their love, encouragement, and unwavering support, which made this path possible. I would also like to thank Davide Ruggeri, my brother and friend, for always being open to discussions, suggestions, and shared reflections.

I would also like to thank my friends Domiziano, Federico Cortese, Lorenzo Laccone, Andrea Grassi, and Simone Micheli for making every return to Rome feel like coming home, and to Marco Bredice for the wonderful time we spent together in Bologna. I am also grateful to Luca Turnassi, whose unexpected time in Szeged made this journey even more special, and to Seppe Bomeré for being such a supportive and encouraging friend during the early stages of my PhD.

Looking back, this PhD has been much more than an academic journey: it has been a period of growth, challenge, discovery, and meaningful human connections. Each of the people mentioned here, in different ways, has contributed to making these years memorable and enriching. For that, I am truly grateful. I will carry not only the knowledge and experience gained during this time, but also the friendships, generosity, and support that accompanied me throughout this path.