Towards Efficient Classification of Gene Expression Data with Machine Learning

Master's Degree in Data Science

José Leonel de Sousa Febra

Leiria, September 2025

Towards Efficient Classification of Gene Expression Data with Machine Learning

Master's Degree in Data Science

José Leonel de Sousa Febra

Master's thesis carried out under the guidance of Doctor Carlos Fernando de Almeida Grilo, Professor at the School of Technology and Management of the Polytechnic Institute of Leiria, Doctor João Pedro Almeida Meneses, Researcher at the Centre for Rapid and Sustainable Product Development, and Doctor Paula Cristina Rodrigues Pascoal Faria, Professor at the School of Technology and Management of the Polytechnic Institute of Leiria.

Leiria, September 2025

Statement of Originality and Copyright

This dissertation is original, made only for this purpose, and all authors whose studies and publications were used to complete it are duly acknowledged.

Acknowledgements

I would like to express my gratitude to my supervisors, Doctor Carlos Grilo, Doctor João Meneses and Doctor Paula Faria, for their guidance and support throughout the course of this dissertation, particularly to Doctor Carlos Grilo for his patience, availability, and for continually encouraging me to question and reflect critically, which helped me to strengthen the analysis and justification underlying this work.

I would also like to thank the FCT project OptiBioScaffold (ref. PTDC/EME-SIS/4446/2020), led by Professor Paula Faria, which proposed the topic of this dissertation and reinforced the importance of connecting the models studied here with experimental research in this area of biology.

My sincere thanks go to Doctor Anabela Marto for her generosity in providing a computer during the final stage of this work, without which completing this dissertation on time would have been far more difficult.

I am also grateful to Miguel Felgueiras for suggesting that I explore an area previously unfamiliar to me, without which I might not have embarked upon this path.

Finally, and most importantly, I extend my deepest thanks to my wife, Ana, for the many conversations and thoughtful discussions that helped me clarify aspects of this work, and to my daughter, Ana Leonor, for the irreplaceable time lost to my absence during this journey — moments that can never be regained — and for their unconditional support, which as a family made the successful completion of this work possible.

Abstract

The growing availability of gene expression datasets offers new opportunities for applying machine learning to biological classification. These datasets are typically high-dimensional, limited in sample size, and experimentally diverse, posing both computational and biological challenges. This dissertation investigates how deep learning and classical machine learning models can classify gene expression profiles while evaluating the impact of reducing experimental and computational complexity, thereby lowering associated costs.

Three datasets were analysed: GSE3406, with temporal profiles of *Saccharomyces* species under stress; GSE1723, profiling *S. cerevisiae* under nutrient limitation and oxygen variation; and GSE6186, recording temporal expression during *Drosophila melanogaster* embryogenesis. Four models were compared — convolutional neural networks (CNN), long short-term memory networks (LSTMs), support vector machines (SVMs), and XGBoost — with hyperparameters optimised via the Optuna library and performance assessed through repeated experiments.

Results show that CNNs achieved the best performance in GSE3406, LSTMs were slightly superior in GSE6186, and CNN and XGBoost performed competitively in GSE1723. Comparable accuracy was often obtained under reduced experimental conditions, such as subsets of stimuli, nutrient regimes, or time points. Additionally, gene-level consistency analysis in GSE3406 identified genes consistently well or poorly classified, supporting dimensionality reduction and biological interpretation.

This work demonstrates the potential of deep learning for the classification of gene expression profiles, proposing strategies to simplify experimental design without compromising predictive performance.

Keywords: Gene expression, Deep learning, Machine learning, Classification, Optuna, Time series.

Resumo

A crescente disponibilidade de conjuntos de dados de expressão genética oferece novas oportunidades para a aplicação de técnicas de aprendizagem automática à classificação biológica. Estes tipos de dados são normalmente de elevada dimensionalidade, limitados em tamanho amostral e experimentalmente diversos, colocando constrangimentos tanto computacionais como biológicos. Esta dissertação investiga de que forma modelos de aprendizagem profunda e de aprendizagem automática clássica podem classificar perfis de expressão genética, avaliando simultaneamente o impacto da redução da complexidade experimental e computacional, e consequentemente a diminuição dos custos associados.

Foram analisados três conjuntos de dados: GSE3406, com perfis temporais de espécies de *Saccharomyces* sob diferentes estímulos; GSE1723, com perfis de *S. cerevisiae* em condições de limitação de nutrientes com distintos regimes de oxigénio; e GSE6186, que contém a expressão genética durante a embriogénese de *Drosophila melanogaster*. Quatro modelos foram comparados — redes neuronais convolucionais (CNN), redes de memória de longo curto prazo (LSTM), máquinas de vetores de suporte (SVM) e XGBoost — com hiperparâmetros otimizados através da biblioteca Optuna e com o desempenho avaliado através da repetição sistemática de experiências.

Os resultados mostram que as CNN atingiram o melhor desempenho no GSE3406, as LSTM foram ligeiramente superiores no GSE6186, e as CNN e as XGBoost tiveram desempenhos competitivos no GSE1723. Importa salientar que, foi possível obter um desempenho comparável em condições experimentais reduzidas como subconjuntos de estímulos, regimes nutricionais ou pontos temporais. Adicionalmente, a análise de consistência ao nível dos genes no GSE3406 identificou genes sistematicamente bem ou mal classificados, apoiando estratégias de redução da dimensionalidade e de interpretação biológica.

Este trabalho demonstra o potencial da aprendizagem profunda para a classificação de perfis de expressão genética, propondo estratégias para simplificar o desenho experimental sem comprometer a capacidade de previsão.

Palavras-chave: Expressão genética, Aprendizagem profunda, Aprendizagem automática, Classificação, Optuna, Séries temporais.

Table of Contents

Statement of Originality and Copyright iii
Acknowledgements iv
Abstract v
Resumo vi
List of Figures x
List of Tables xi
List of Abbreviations and Acronyms xii
1. Introduction 1
1.1. Objectives of the Study 1
1.2. Methodology 1
1.3. Structure of the Document 3
2. Literature Review 5
2.1. Technologies for Measuring Gene Expression 5
2.2. Classification Based on Gene Expression 6
2.3. Traditional Data Analysis Techniques for Gene Expression 7
2.3.1. Unsupervised Methods 7
2.3.2. Supervised Methods 8
2.4. Machine Learning and Neural Networks Applied to Genomic Data 8
2.4.1. Classical Machine-Learning Algorithms 8
2.4.2. Deep Neural Networks Applied to Gene Expression 10
2.4.3. Transfer Learning and Recent Advances 11
2.5. Similar Studies 12
3. Data Preparation and Problem Definition 15
3.1. Data Sources and General Context 15
3.2. Datasets Description and Preparation 15
3.2.1. GSE3406 — Gene Expression in Yeast Species Under Stress 16
3.2.2. GSE1723 — Gene Expression of *S. cerevisiae* Under Nutrient Limitation 17
3.2.3. GSE6186 — Gene Expression During *Drosophila Melanogaster* Embryogenesis 19

3.2.4. Comparison Between Datasets 21

3.3. Common Preprocessing 21

3.4. Validation Strategy and Data Splitting 22
3.4.1. Hold-Out Validation 22
3.4.2. K-Fold Cross-Validation 22

3.5. Data Preparation for Modelling 23
3.5.1. Normalisation 23
3.5.2. Class Encoding 23
3.5.3. Input Structuring 23
3.5.4. Seeds and Reproducibility 24
3.5.5. Class Balancing 24

3.6. Problem Definition 25

4. Modelling Approaches 27

4.1. General Modelling Strategy 27
4.1.1. Justification for the Models Used 27
4.1.2. Hyperparameter Optimisation Process 28
4.1.3. Model Training Procedures 33
4.1.4. Repetition Strategy and Statistical Considerations 33
4.1.5. Evaluation Metrics 34

4.2. Modelling with GSE3406 35
4.2.1. Model Comparison on Full Dataset 35
4.2.2. Stimulus Combination Strategy 36
4.2.3. Gene-Level Consistency Analysis 36

4.3. Modelling with GSE1723 37
4.3.1. Model Evaluation on the Complete Dataset 37
4.3.2. Nutrient Combination Strategy 37

4.4. Modelling with GSE6186 38
4.4.1. Model Evaluation on Full Time Series 38
4.4.2. Reduced Input Strategy 38

4.5. Tools and Libraries Used 38

5. Results and Discussion 41

5.1. General Results Overview 41

5.2. Results with the GSE3406 Dataset 42
5.2.1. LSTM Input Strategy 42
5.2.2. Full Dataset (Five Stimuli) 42

5.2.3. Individual Stimuli 45
5.2.4. Pairwise Combinations 45
5.2.5. Triple Combinations 46
5.2.6. Quadruple Combinations 47
5.2.7. Gene-Level Consistency Analysis 48
5.2.8. Integrated Discussion of GSE3406 Results 49

5.3. Results with the GSE1723 Dataset 50
5.3.1. Complete Dataset (All Nutrient Conditions) 50
5.3.2. Individual Nutrients 51
5.3.3. Pairwise Combinations 52
5.3.4. Triple Combinations 52
5.3.5. Integrated Discussion of GSE1723 Results 53

5.4. Results with the GSE6186 Dataset 55
5.4.1. Complete Time Series (28 Points) 55
5.4.2. Reduced Input Strategies 56
5.4.3. Integrated Discussion of GSE6186 Results 56

5.5. Comparative Analysis with Previous Studies 58

6. Conclusions 61

References 63

Appendix A - Hyperparameter Optimisation Results 71

Appendix B - Execution Times for Optimization and Training 76

List of Figures

Figure 1 - The CRISP-DM methodology with its six iterative phases (IBM, 2025). 2

Figure 2 - Final structure of the GSE3406 dataset. 17

Figure 3 - Final structure of the GSE1723 dataset. 18

Figure 4 - Expression profiles of the three GSE6186 classes (maternal, transient, activated). Adapted from Tripto et al. (2020). 19

Figure 5 - Final structure of the GSE6186 dataset. 20

Figure 6 - Overview of the modelling workflow. 27

Figure 7 - Fixed architectures adopted for each dataset and model. 31

Figure 8 - Training and validation curves of the best model on the full GSE3406 dataset. 43

Figure 9 - Test accuracy across 60 repetitions with the CNN model on the full GSE3406 dataset. 44

Figure 10 - Confusion matrix of the best CNN model on the full GSE3406 dataset. 44

Figure 11 - Confusion matrix for the CNN model trained on the complete GSE1723 dataset. 54

Figure 12 - Confusion matrix for the CNN model trained on the best reduced subset (C + P + S). 54

Figure 13 - Training and validation curves for the alternating full-sequence configuration. 58

Figure 14 - Training and validation curves for the alternating half-sequence configuration. 58

List of Tables

Table 1 - Biological and structural characteristics of the GSE3406, GSE1723 and GSE6186 datasets 21

Table 2 - Optuna optimization intervals for the GSE3406 dataset 29

Table 3 - Optuna optimization intervals for the GSE1723 dataset 29

Table 4 - Optuna optimization intervals for the GSE6186 dataset 30

Table 5 - Summary of best-performing models for each dataset using the complete data. 41

Table 6 - Performance of alternative LSTM input strategies on the GSE3406. 42

Table 7 - Best performance of different models on the full GSE3406 dataset. 43

Table 8 - Performance of CNN models trained on individual stimuli (GSE3406). 45

Table 9 - Performance of CNN models trained on pairwise stimulus combinations (GSE3406). 46

Table 10 - Performance of CNN models trained on triple stimulus combinations (GSE3406). 47

Table 11 - Performance of CNN models trained on quadruple stimulus combinations (GSE3406). 48

Table 12 - Classification results with K-Fold cross-validation (GSE3406). 48

Table 13 - Genes consistently well classified across all four species (GSE3406). 49

Table 14 - Best performance of different models on the full GSE1723 dataset. 51

Table 15 - Performance of CNN models trained on individual nutrient limitations (GSE1723). 51

Table 16 – Performance of CNN models trained on pairwise nutrient limitations (GSE1723). 52

Table 17 - Performance of CNN models trained on triple nutrient limitations (GSE1723). 52

Table 18 - Best performance of different models on the full GSE6186 dataset. 55

Table 19 - LSTM results on the GSE6186 dataset with reduced temporal strategies. 56

Table 20 - CNN results on the GSE6186 dataset with reduced temporal strategies. 56

Table 21 - Comparative results between this study and Tripto et al. (2020). 59

List of Abbreviations and Acronyms

CLT	Central Limit Theorem
CNN	Convolutional Neural Network
CRISP-DM	Cross-Industry Standard Process for Data Mining
CV	Cross-Validation
DNA	Deoxyribonucleic Acid
DNN	Deep Neural Network
ESTG	Escola Superior de Tecnologia e Gestão
GEO	Gene Expression Omnibus
GPU	Graphics Processing Unit
HCIST	International Conference on Health and Social Care Information Systems and Technologies
HS	Heat Shock
IPLeiria	Instituto Politécnico de Leiria
k-NN	k-Nearest Neighbours
LLN	Law of Large Numbers
LSTM	Long Short-Term Memory
MMS	Methyl Methanesulfonate
NCBI	National Center for Biotechnology Information
NGS	Next-Generation Sequencing
NS	Nitrogen Starvation
ORF	Open Reading Frame
PCA	Principal Component Analysis
RNA	Ribonucleic Acid
SGD	Saccharomyces Genome Database
SHAP	SHapley Additive exPlanations
SOM	Self-Organizing Map
SVC	Support Vector Classifier
SVM	Support Vector Machine
TCGA	The Cancer Genome Atlas
TPE	Tree-structured Parzen Estimator

1. Introduction

Gene expression analysis is an essential tool for understanding the molecular mechanisms that regulate biological processes. In addition to enabling the identification of relevant genes in different contexts, it also makes it possible to compare species, to characterise environmental or stress conditions, and to investigate cellular responses to drugs or therapeutic interventions. Classification based on gene expression data is therefore of particular importance in computational biology, as it allows the exploration of complex patterns within large volumes of information. At the same time, the growing need to reduce costs underscores the relevance of developing approaches that maximise efficiency without compromising analytical performance.

1.1. Objectives of the Study

The general objective of this study is to develop and evaluate machine learning models capable of classifying gene expression data, while investigating whether comparable or at least satisfactory results can be achieved using reduced subsets of the datasets instead of the full data, thereby lowering experimental and computational costs and complexity.

The specific objectives are as follows:

- To analyse the impact of different stress stimuli, nutrient limitations, and temporal resolutions — individually or in combination — on the accuracy of classification models, as a means of reducing data requirements while maintaining comparable performance;
- To identify genes that are consistently well or incorrectly classified across experiments, providing evidence of their biological relevance and their potential use for feature selection;
- To compare the results obtained in this work with those reported in previous studies, thus situating the findings within the existing literature.

1.2. Methodology

This study was conducted following the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology, which provided a structured framework encompassing data

understanding, preparation, modelling, evaluation, and deployment considerations. Although the deployment phase was not within the scope of this work, the remaining stages were applied in line with the CRISP-DM cycle (Figure 1).

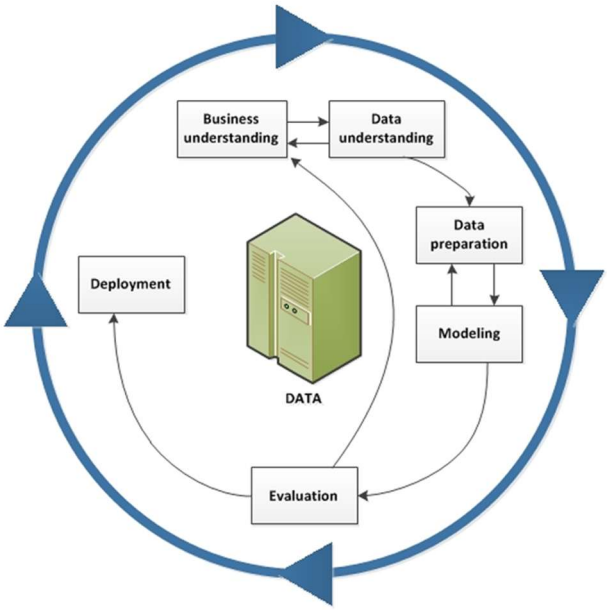

Figure 1 - The CRISP-DM methodology with its six iterative phases (IBM, 2025).

Within this framework, the study assessed the feasibility of reducing experimental and computational complexity in gene expression classification tasks. Three publicly available datasets were employed: GSE3406, containing stress-response profiles of different *Saccharomyces* species; GSE1723, comprising nutrient-limitation experiments with *S. cerevisiae* under distinct oxygen regimes; and GSE6186, describing gene expression during *Drosophila melanogaster* embryogenesis.

Data preprocessing included data cleaning, handling of missing values, normalisation, class encoding, structuring of inputs for model training, and class balancing where required. Depending on the objective, either hold-out or K-fold cross-validation strategies were applied to ensure a reliable evaluation.

Four modelling approaches were compared: convolutional neural networks (CNNs), long short-term memory networks (LSTMs), support vector machines (SVMs), and XGBoost classifiers. Hyperparameters were optimised using an optimization library, and model performance was assessed through multiple repetitions to capture variability.

Evaluation relied on accuracy, F1-score, and confusion matrices. In addition, a gene-level consistency analysis was conducted to identify genes consistently well or misclassified across repetitions, providing a basis for biological interpretation and feature-reduction strategies.

1.3. Structure of the Document

This dissertation is organised into six main chapters, followed by references and appendices. Chapter 2 provides a review of the literature, covering the fundamental concepts, technologies for measuring gene expression, traditional and modern analytical methods, and related studies that support the chosen approach. Chapter 3 describes the datasets, data preparation procedures, validation strategies, and the formulation of the classification problems. In Chapter 4, the modelling approaches are detailed, as well as the optimisation process and evaluation metrics. Chapter 5 presents the results and discussion, analysing the performance of the models across the three datasets and under reduced experimental conditions, with additional emphasis on gene-level consistency analysis. Finally, Chapter 6 concludes the dissertation, summarising the main contributions, assessing the achievement of the proposed objectives, and suggesting future research.

This page was intentionally left blank

2. Literature Review

This chapter brings together the key concepts, methodologies, and approaches related to the analysis and classification of gene expression data, with particular emphasis on the application of machine learning techniques and deep neural networks. It begins with an overview of the technologies used to measure gene expression, followed by a description of both traditional and advanced methods for analysing such data, and concludes with a summary of relevant studies that support the approach adopted in this dissertation.

2.1. Technologies for Measuring Gene Expression

Gene expression refers to the quantification of gene activity, which is measured by the amount of messenger RNA (mRNA) produced under specific experimental conditions. By determining how much mRNA is transcribed from each gene, it is possible to identify not only which genes are active but also their level of activity. This allows the study of cellular processes such as differentiation, stress response, disease progression, and adaptation to environmental stimuli (Alberts et al., 2014).

Two of the main technologies currently used to generate large-scale gene expression data are microarrays and RNA-Seq. Microarrays operate based on fixed probes attached to a surface, each designed to detect the expression of a known gene. The RNA from the sample binds to these probes through complementarity, and the intensity of the emitted signal provides an estimate of the relative expression level of each gene. This method is effective for comparing previously identified genes across different conditions (Quackenbush, 2001; Zhao et al., 2014).

RNA-Seq, in turn, uses next-generation sequencing (NGS) techniques to directly read the RNA present in the sample, allowing for more accurate quantification of gene expression, as well as the identification of novel genes, different variants of known genes (such as mutations or polymorphisms), and alternative transcripts not yet present in reference databases (Ozsolak & Milos, 2011; Wang et al., 2009). Classification based on gene expression is particularly important in detecting low-abundance transcripts and a broader dynamic range, making it particularly well suited for applications in biomedical research and supervised classification tasks (Hira & Gillies, 2015).

Once the expression levels have been quantified, these data can be used to address a wide range of biological and computational problems. One such task is classification, which is explored in the following section.

2.2. Classification Based on Gene Expression

Over the last decades, the increasing availability of gene expression data has enabled the development of computational models for a wide range of classification tasks. These tasks may focus on the identification of cell types (Kiselev et al., 2018), physiological states (e.g., cell cycle phases (Spellman et al., 1998)), or the distinction between organisms based on their responses to environmental stimuli (Tirosh et al., 2011).

Gene expression data are typically represented in the form of expression matrices, where each row corresponds to a gene and each column to a specific experimental condition, such as a type of stimulus, a time point, or a physiological state.

In this structure, genes are considered explanatory variables (features), while the different experimental samples represent the observations (Guyon et al., 2002; Tripto et al., 2020). These matrices may contain thousands of genes and only a few dozen samples, which constitutes a classic high-dimensional, low-sample-size problem — known as the *curse of dimensionality* — with direct implications for the effectiveness and generalisation of classification algorithms (Dougherty, 2001; Hira & Gillies, 2015). This imbalance hinders the application of many machine learning algorithms, which tend to overfit when the number of variables greatly exceeds the number of training instances. Furthermore, the number of labelled samples per class is often limited, compromising the generalisation ability of classification models and requiring the use of regularisation techniques, feature selection, or methods better suited to sparse data (Bar-Joseph et al., 2012; Guyon et al., 2002).

Classification based on gene expression is important in both biomedical and industrial contexts. In the healthcare domain, it is used for the diagnosis of cancers and other diseases (Fakoor et al., 2013), for predicting responses to therapies (Baranzini et al., 2004), and for identifying relevant biomarkers (Aliouane et al., 2025). In biotechnological and environmental settings, it enables the analysis of microbial responses to different cultivation conditions, such as nutrient availability, temperature, oxygen levels, or toxic agents (Tai et al., 2005).

The literature shows that the nature of expression data — characterised by high dimensionality, the presence of strongly correlated gene expression levels, and both technical and biological noise — poses a significant challenge for traditional classification algorithms (Hira & Gillies, 2015). For this reason, approaches based on machine learning methods and deep neural networks have been increasingly explored (Aliouane et al., 2025; Fakoor et al., 2013).

2.3. Traditional Data Analysis Techniques for Gene Expression

Traditional techniques for analysing gene expression data are generally divided into two main groups (Hira & Gillies, 2015; Tripto et al., 2020): unsupervised methods, such as principal component analysis (PCA) and clustering (e.g. k-means, Self-Organizing Maps - SOMs), and supervised techniques such as logistic regression and decision trees (Hastie et al., 2009).

Although suitable for preliminary analysis, these approaches face major challenges when applied to datasets with thousands of genes and only a few samples, as is typical of microarray or RNA-Seq studies (Hira & Gillies, 2015). They are commonly used for dimensionality reduction, pattern detection, or simple classifier construction, but their performance is constrained by the intrinsic characteristics of the data (Bar-Joseph et al., 2012).

2.3.1. Unsupervised Methods

In the case of unsupervised methods, such as PCA, the aim is to project the data into a lower-dimensional space, preserving the maximum variance. Although useful for visualization and noise filtering, PCA assumes linear relationships between variables and can lose biologically relevant information by condensing the data into a few principal components (Hastie et al., 2009; Hira & Gillies, 2015).

Similarly, clustering algorithms, such as k-means and self-organizing maps (SOMs), seek to group genes or samples with similar profiles, but are strongly influenced by the choice of the number of clusters, the scale of the variables and distance metrics that do not always capture subtle biological relationships (Tripto et al., 2020).

2.3.2. Supervised Methods

Among supervised methods, logistic regression has been successfully applied to binary classification tasks, such as distinguishing between healthy and tumour tissues (Guyon et al., 2002; Tabassum et al., 2024). Its effectiveness, however, is limited by the small sample sizes typical of gene expression studies, which increase the risk of overfitting. Decision trees, while offering good interpretability, are prone to instability and noise sensitivity (Bar-Joseph et al., 2012).

In general, traditional approaches struggle with high dimensionality, gene collinearity and experimental variability (Tripto et al., 2020), challenges that have led to the adoption of strategies such as L1/L2 regularisation and ensemble methods (Hira & Gillies, 2015; Tabassum et al., 2024).

Even so, these solutions are often insufficient to capture complex non-linear or temporal relationships — making room for the adoption of more advanced techniques, such as deep neural networks.

2.4. Machine Learning and Neural Networks Applied to Genomic Data

Although classical methods have laid the foundations for the initial analysis of genomic data, their effectiveness is often limited by the characteristics discussed previously. To address these issues, modern approaches based on machine learning and neural networks have been increasingly adopted, which offer significant advantages over traditional techniques (Montesinos-López et al., 2021)

2.4.1. Classical Machine-Learning Algorithms

This section presents some of the main machine learning algorithms used in genetic classification, along with their typical use in other domains.

Support Vector Machines

SVMs are widely used in gene expression classification due to their capacity to handle high-dimensional data and identify optimal separating hyperplanes. They are particularly suited for small sample sizes and are often combined with feature selection techniques to reduce overfitting (Guyon et al., 2002). Their performance has been validated in various genomic studies, especially in binary classification settings (Alharbi & Vakanski, 2023) although it may degrade when dealing with noisy or highly overlapping classes (Hira & Gillies, 2015).

Extreme Gradient Boosting (XGBoost)

It is a decision-tree-based ensemble algorithm known for its efficiency and scalability. It performs well on sparse and high-dimensional datasets, using both L1 and L2 regularisation to mitigate overfitting — a particularly useful feature when sample sizes are small (Deng et al., 2022). XGBoost[1] has been successfully applied to complex problems such as cancer classification based on gene expression data, demonstrating high predictive accuracy (Tabassum et al., 2024). However, the resulting models can be difficult to interpret due to their ensemble nature (Hastie et al., 2009).

Random Forests

Random Forests are ensemble learning algorithms that build multiple decision trees using random subsets of data and features, combining their predictions through majority voting, or averaging. This approach improves generalisation and reduces variance, making it suitable for high-dimensional datasets such as gene expression profiles (Hira & Gillies, 2015). Although individual trees are interpretable, the combined model is more complex. Nevertheless, Random Forests allow post hoc analysis through feature importance measures, including Gini importance, permutation-based methods, and SHAP values (Wang et al., 2009). These models have been effectively used in genomic studies, particularly for gene selection and classification tasks in oncology (Deng et al., 2022). Their performance may degrade when all features contribute equally or when strong feature interactions exist that are not captured by axis-aligned splits (Hastie et al., 2009).

Logistic Regression

Logistic regression remains a widely used algorithm for binary classification problems, owing to its simplicity, computational efficiency, and ability to provide explicit class probabilities. It is a linear model that estimates the probability of a sample belonging to a given class by applying a sigmoid function to a weighted sum of the input variables (Hastie et al., 2009). In addition, it offers direct interpretability of the model coefficients, making it particularly valuable in settings where understanding the relative importance of predictors is essential. While effective in many scenarios, logistic regression assumes a linear relationship between features and the log-odds of the outcome, which can limit its performance when dealing with complex, non-linear patterns in gene expression data (Hira & Gillies, 2015; Tripto et al., 2020).

[1] https://xgboost.readthedocs.io

k-Nearest Neighbors (k-NN)

Is a simple and intuitive method that assigns classes based on the proximity of a sample to its k most similar instances in the training set. It has been applied to gene expression data due to its non-parametric nature and capacity to model complex class boundaries without requiring an explicit training phase (Tripto et al., 2020). However, its performance strongly depends on the choice of distance metric and the scale of the variables, which can be problematic in high-dimensional gene expression datasets. Moreover, k-NN is sensitive to noise, affected by class imbalance, and prone to the *curse of dimensionality* when irrelevant or redundant features are present (Hira & Gillies, 2015).

Naive Bayes

Naive Bayes is a probabilistic classifier based on Bayes's Theorem, which assumes complete independence between the explanatory variables. It is recognised for its simplicity, computational efficiency, and ability to provide explicit class probabilities, particularly when fast classification is required. In gene expression analysis, it has been used as a baseline method or for comparative studies due to its speed and scalability, even with high-dimensional data (Hira & Gillies, 2015; Tripto et al., 2020). The algorithm estimates the posterior probability of a sample belonging to a given class based on the conditional probabilities of each attribute, assuming feature independence. While effective in various settings, the independence assumption often does not hold in gene expression data, where genes can be strongly correlated, potentially reducing the model's discriminative power (Hira & Gillies, 2015). Nonetheless, Naive Bayes remains relevant as a benchmark for evaluating more complex models, particularly in studies focused on expression pattern classification and the performance of supervised classifiers (Tripto et al., 2020).

2.4.2. Deep Neural Networks Applied to Gene Expression

CNNs and LSTMs architectures have been increasingly explored for the analysis of gene expression data. These models offer several advantages over classical machine learning approaches, particularly in handling high-dimensional and non-linear relationships (Mostavi et al., 2020; Tripto et al., 2020).

CNNs have proven effective in detecting local and spatial patterns in data, even when it is reorganised into two-dimensional matrices — as is done with gene expression data to simulate an image-like structure (Mostavi et al., 2020). This approach enables the automatic extraction of relevant features and is particularly useful in supervised classification tasks.

LSTM networks, in turn, were originally designed for time series analysis and have therefore been applied to modelling gene expression profiles over time — such as those recorded in the GSE6186 and GSE3406 datasets (Tripto et al., 2020). Their architecture allows them to capture temporal dependencies and expression trends across distinct stages of the experiment.

Several comparative studies have demonstrated that deep neural network models often outperform classical methods traditionally used as baselines in gene expression classification tasks. For instance, Tripto et al. (2020), reported that CNNs achieved the best accuracy and F1-score on the GSE6186 dataset, while LSTMs showed only intermediate performance. In contrast, clustering-based approaches produced clearly inferior results (Mostavi et al., 2020; Tripto et al., 2020).

Despite their advantages, deep neural networks have their own issues, such as the risk of overfitting in datasets with few labelled samples, the need for high computational resources and the difficulty of interpreting the model's internal weights (Hira & Gillies, 2015). Techniques such as dropout, batch normalisation and cross-validation have been used to mitigate these problems. The complexity of the networks also demands careful consideration in selecting the architecture and tuning the hyperparameters (Mostavi et al., 2020; Tripto et al., 2020).

2.4.3. Transfer Learning and Recent Advances

In addition to conventional supervised approaches, transfer learning techniques have been increasingly explored, in which a model previously trained in a given genomic context is adjusted for a new task, taking advantage of the representations and weights already learnt. This strategy has proved particularly promising in scenarios with a shortage of labelled data, reducing the need for large volumes of samples to achieve good performance (Montesinos-López et al., 2021; Yu et al., 2019).

Within transfer learning, a common approach is fine-tuning, where a pre-trained model is partially retrained on a new dataset, adapting its parameters to the specific task. Recently, Spolaôr et al. (2023) proposed and evaluated eight fine-tuning strategies for pre-trained convolutional neural networks (VGG16 and VGG19) on clinical datasets with limited images, showing that the use of differentiated learning rates and partial unfreezing of layers can substantially improve performance in domains with scarce data. The approach proved

effective for both dermoscopic images and computed tomography scans related to COVID-19, reinforcing the usefulness of transfer learning techniques in biomedical contexts with small sample sizes.

Meanwhile, more recent architectures have emerged, such as hybrid models combining CNNs and LSTMs, and attention-based architectures such as transformers — originally developed for natural language processing but now adapted to gene expression tasks (Jiang & Hassanpour, 2025; Zhang et al., 2022).

Furthermore, model explainability techniques such as Shapley values have gained relevance, as they make it possible to interpret the contribution of each gene or variable to a given prediction — a particularly important feature in sensitive biomedical contexts (Yap et al., 2021).

2.5. Similar Studies

Recently, several studies have applied machine learning models and deep neural networks to the classification of gene expression data, using widely available public datasets. One of the most comprehensive works is that by Tripto et al. (2020), which evaluated the performance of CNN, LSTM, SVM and other approaches on three different datasets: GSE6186, GSE3406 and GSE1723. These datasets are also used in the present work and differ in both biological context and structure: GSE3406 comprises time-series data from yeast species under environmental stress, GSE1723 contains static profiles under nutrient and oxygen limitation, and GSE6186 captures temporal expression during *D. melanogaster* embryogenesis. Together, they offer a diverse framework for testing classification approaches across different biological scenarios.

The results reported by Tripto et al. (2020) showed that for GSE6186, CNNs achieved the highest accuracy (96.15%), closely followed by SVM (95.75%), while LSTM performed slightly lower (92.19%). For GSE3406, CNN again led in accuracy (93.14%), with SVM reaching 88.83%. In GSE1723, where the response to nutrient limitation is more complex, all methods performed more modestly, with DNN achieving the top value (84.88%), followed by SVM (83.11%).

These results demonstrate that deep learning architectures can achieve high accuracy in most cases, but also reveal the limitations imposed by datasets with less distinctive characteristics or greater experimental variability.

The work of Mostavi et al. (2020) also represents a relevant contribution in the field of cancer classification based on gene expression profiles. The authors compared various CNNs architectures, including 1D-CNN, 2D-Vanilla-CNN, and 2D-Hybrid-CNN models, trained with data from The Cancer Genome Atlas (TCGA), covering thirty-three types of cancer and corresponding normal tissues. The performances achieved were between 93.9% and 95.7% accuracy, with the 1D-CNN model standing out for its simplicity, robustness to noise and lower risk of overfitting, making it particularly suitable for genomic data of high dimensionality and limited sampling.

It should be noted that, in addition to high accuracy achieved, this study addressed an aspect that is often overlooked: the impact of the tissue of origin on classification. Mostavi et al. (2020) demonstrated that failing to account for this factor leads models to misclassify normal tissues as tumours, due to similarities in their expression profiles. To mitigate this effect, they proposed an additional node in the model to distinguish normal samples, which improved generalisation and the identification of specific biomarkers for each type of cancer.

Another important contribution was the interpretation of the models using saliency maps, which made it possible to identify highly relevant marker genes for each tumour subtype — including well-established genes such as GATA3 and ESR1 in the case of breast cancer, but also novel candidates, which could have diagnostic or therapeutic potential. This ability to extract biologically significant information from neural networks represents a clear asset in their application to gene expression studies.

Another additional aspect addressed in the literature, concerns the reduction of variables or the simplification of experimental conditions. The study by Tripto et al. (2020) explicitly evaluated scenarios with partial removals of time points in the GSE6186 dataset, showing that model performance remained high even with less data. This approach reinforces the potential of reducing experimental complexity without significantly compromising the classification capacity of the models.

In addition to the study by Tripto et al. (2020) and Mostavi et al. (2020), other recent research reinforces the applicability of advanced models to gene expression-based classification.

Deng et al. (2022) proposed a hybrid approach for gene selection in cancer classification tasks, combining the XGBoost algorithm with a multi-objective genetic algorithm — an optimisation method that simultaneously balances multiple criteria, such as classification performance and the number of selected genes. The method demonstrated high accuracy in identifying subsets of relevant genes and building effective classifiers, revealing the potential of automatic feature selection as a critical step prior to modelling.

On the other hand, Fan et al. (2023) developed DeepASDPred, a CNN-LSTM model designed to identify RNA transcripts associated with the risk of autism spectrum disorders. The model integrated nucleotide sequence data (via K-mer coding) with expression profiles, following feature selection based on chi-squared tests and logistic regression. The results showed superior performance compared to traditional methods such as SVM, Random Forests and Logistic Regression, with significantly high AUC and F1-score metrics.

These examples show the diversity of strategies that have been explored — from well-designed pre-processing pipelines to hybrid deep learning architectures — and demonstrate their effectiveness in complex scenarios with high genetic heterogeneity. Overall, the literature reviewed consolidates the methodological foundations of this dissertation, highlighting the effectiveness of the models used and the benefit of experimental simplification through the strategic selection of conditions and variables.

3. Data Preparation and Problem Definition

The procedures for collecting, integrating, transforming, and preparing the data used throughout this work are described in this section. The operations conducted are presented, as well as how the final data was structured to enable the classification models to be built and evaluated. For each of the three datasets, a contextualisation is provided together with the processes of conversion and adaptation to a format compatible with the experiments in this study. Finally, the classification problems addressed with each dataset are defined, establishing the basis for the modelling approaches.

3.1. Data Sources and General Context

All datasets used in this dissertation were obtained from the public GEO repository maintained by the NCBI. The downloaded files included gene expression matrices (in *.txt* or *.soft* format), supplementary probe annotation tables (*GPL4455*, *GPL2910*), and metadata files describing experimental classes and conditions.

This dissertation focuses on three public gene expression datasets: *GSE3406* (Hooper et al., 2007), *GSE1723* (Tai et al., 2005), and *GSE6186* (Tirosh et al., 2006). These studies differ in biological context, experimental objectives, and specific characteristics such as the organism studied, the type of stimulus applied, the number of genes analysed, the temporal resolution, and the class structure. This diversity makes it possible to evaluate classification models in contexts with diverse levels of complexity and to test data reduction strategies aimed at preserving model performance while reducing computational or experimental costs.

3.2. Datasets Description and Preparation

Each dataset is presented individually, combining a description of its biological context, applied stimuli, and experimental conditions with the specific data preparation steps required to obtain the final datasets used for the modelling experiments. In this way, it is possible to understand both the original structure of the data and the transformations applied to produce a format suitable for subsequent modelling.

3.2.1. GSE3406 — Gene Expression in Yeast Species Under Stress

The GSE3406 dataset contains gene expression profiles collected from four closely related species of the *Saccharomyces* genus: *S. cerevisiae*, *S. paradoxus*, *S. mikatae* and *S. kudriavzevii*. These species were subjected to five types of environmental stimuli, each inducing specific transcriptional changes:

- H_2O_2 – oxidative stress;
- Heat shock – thermal stress;
- MMS (methyl methanesulfonate) – a DNA-damaging agent;
- Nitrogen starvation – nitrogen deprivation;
- Transfer from glucose to glycerol – change in carbon source (from glucose to glycerol).

A total of 13,056 genes were selected and measured across all experimental samples. The expression of each gene was recorded over six time points (10, 20, 30, 45, 60 and 90 minutes) for four environmental stimuli: H_2O_2, heat shock, MMS and transfer from glucose to glycerol, and in the case of nitrogen starvation, seven time points were considered: 10 minutes, 20 minutes, 45 minutes, 2 hours, 4 hours, 8 hours and 1 day. Each combination of species, stimulus and time corresponds to an individual sample. In total, the dataset includes 191 samples, including biological replicates in some conditions.

This dataset was originally published by Tirosh et al. (2006), in a study centred on identifying genetic signatures associated with inter-species variation in gene expression. Since then, it has been reused in several studies, including Yoneya & Mamitsuka, (2007), who applied a hidden Markov model to detect temporal differences in gene expression under different experimental conditions, and Cotton et al. (2015), who investigated pathway reconnection by analysing the heterogeneity of interaction between species. More recently, it was explored in Tripto et al. (2020), where deep learning architectures (CNN, LSTM, SVM, DNN) were applied to classify yeast species based on their expression patterns in response to environmental stimuli.

To build the final version of the dataset, the original expression file was first transposed to reorganise the data by gene. At the same time, a reference file was manually created from the information available on the GEO page, containing the associations between samples, species, stimuli, and time points.

After transposition, the two files were integrated, allowing each expression profile to be linked to the corresponding experimental conditions. Subsequently, all columns and rows composed exclusively of zeros were removed. The dataset was thus restructured so that each line showed a gene with its expression values under different stimuli, accompanied by the species to which it belongs.

The final structure of the dataset resulting from this process contains one row per gene and includes columns with the gene identifier, the species, and the expression values recorded at multiple time points — each corresponding to a specific combination of stimulus and time. Figure 2 provides a preview of the dataset's final form. Due to its size, not all columns can be displayed in the illustration.

Figure 2 - Final structure of the GSE3406 dataset.

Gene	species	H2O2 (0.3mM)(10min)	H2O2 (0.3mM)(20min)	H2O2 (0.3mM)(30min)	H2O2 (0.3mM)(45min)	H2O2 (0.3mM)(60min)	H2O2 (0.3mM)(90min)
2877	S. mikatae	0.788729312	1.005189946	0.996616937	0.734748383	0.431773811	2.084151018
7651	S. mikatae	1.140024442	1.339223273	-0.00139724	0.703583752	0.036388368	0.395985743
2219	S. paradoxus	0.600233688	0.473312058	0.913277219	1.219753836	1.478457881	0.554741299
7934	S. kudriavzevii	-0.370687562	-0.140188914	0.163630096	-0.077081604	0.034281572	0.257007924
1643	S. paradoxus	0.191176599	0.470299557	0.127398958	1.430896631	1.32888251	0.932664088
10921	S. paradoxus	0.525100456	0.618816261	-0.635658771	0.909489555	-0.185347093	-0.598276026
11509	S. mikatae	-0.144881424	-0.03249281	0.448700314	-0.172512794	0.285587312	-0.129856524
8772	S. mikatae	-0.453775038	-0.19353713	0.00047877	-0.686534268	-0.050792779	-0.105036985
9544	S. cerevisiae	0.017044927	-0.043224629	0.035109943	0.039815591	0.163964463	0.322875518

3.2.2. GSE1723 — Gene Expression of *S. cerevisiae* Under Nutrient Limitation

The GSE1723 dataset contains transcriptional profiles of *Saccharomyces cerevisiae* grown in chemostat cultures under four nutrient-limiting conditions: carbon, nitrogen, phosphorus, and sulphur. For each nutrient, cells were cultivated under aerobic and anaerobic conditions, resulting in a total of eight different experimental environments.

Gene expression was measured using Affymetrix microarrays, and the data were collected under steady-state conditions — meaning that the expression values reflect a stable physiological state obtained under fixed and independently controlled experimental conditions, rather than a temporal progression. For each of the eight conditions, three biological replicates were obtained, leading to 24 expression values per gene (8 conditions × 3 replicates). A total of 9,326 genes were profiled across all conditions.

Generated in the context of a study by Tai et al. (2005), GSE1723 captures the combined effects of oxygen levels and nutrient availability on yeast transcriptional responses. Since its publication, it has been reused in various contexts, including the inference of transcriptional regulatory modules (Knijnenburg et al., 2007) and the evaluation of classification methods

to distinguish oxygen availability based on gene expression patterns (Tripto et al., 2020) These works highlight both the biological diversity and the computational difficulty of the dataset: only around 2.6% of genes respond consistently to oxygen, while more than 40% remain unresponsive in all the conditions tested (Knijnenburg et al., 2007).

In this dissertation, GSE1723 was used to evaluate the ability of machine learning and deep learning models to classify oxygen availability (aerobic vs. anaerobic) based on gene expression data.

This dataset file included both probes and gene identifiers. All probes corresponding to technical controls used by the Affymetrix platform were removed from the dataset. Although they show expression values, these probes do not represent genes from the organism under study (*S. cerevisiae*), but are instead used for hybridisation quality control, detection, background noise, or technical standardisation. Consequently, they were not considered relevant to the classification task and were excluded from the final dataset (Allison et al., 2006; Bar-Joseph, 2004).

The data were then organised in a way that each row was associated with its respective experimental class, allowing the distinction between the different combinations of limited nutrient and oxygen availability. The *species* column was removed, since all the records belong to the same species.

The final dataset contains one row per gene, and the columns include the gene identifier, a class label indicating oxygen availability (aerobic or anaerobic), and the expression values measured under four nutrient-limited conditions, each with three biological replicates , as partially illustrated in Figure 3.

Figure 3 - Final structure of the GSE1723 dataset.

gene	class	carbon_limited_1	carbon_limited_2	carbon_limited_3	nitrogen_limited_1	nitrogen_limited_2	nitrogen_limited_3	phosphorus_limited_1
11321_at	aerobic	51.5	39.5	49.9	81.7	64	89	83.8
2973_at	aerobic	0.3	2.5	0.2	0.4	1.4	0.1	0.8
9922_at	anaerobic	205.6	186.1	304.2	244.2	331.9	285.3	201
4411_at	aerobic	9.1	8.8	13.1	10.2	11.3	10.1	14.1
10428_at	anaerobic	6.8	3.2	0.5	7.3	10.9	3.3	2.2
2609_at	anaerobic	11.3	2.4	2.7	2.8	5.3	4.1	4.9
5880_at	aerobic	12	5	15.7	15.2	10.8	5.8	8.6
2981_at	aerobic	0.4	0.6	1.3	0.1	2	2.2	5.2
5231_at	anaerobic	44.5	21.3	21.2	22.1	30.5	28.4	31.2

3.2.3. GSE6186 — Gene Expression During *Drosophila Melanogaster* Embryogenesis

The GSE6186 dataset contains gene expression profiles recorded during the embryonic development of *D. melanogaster*. Gene expression was measured using microarrays at 28 different time points, covering a 24-hour window — the total duration of embryogenesis, from fertilisation to the larval stage. To better capture the dynamics of early development, the first 6.5 hours were sampled in 1-hour windows shifted every 0.5 hours, creating overlapping intervals, followed by hourly sampling for the rest of the time. The final expression values for each gene at each time point represent the average of three biological replicates, thus reducing experimental noise and ensuring greater measurement precision. Expression profiling was carried out for 12,868 genes, based on FlyBase version 4.0 annotation. All measurements were made against a common reference sample representing the complete life cycle of *D. melanogaster*.

The dataset stems from a study by Hooper et al. (2007), which aimed to identify groups of co-regulated genes involved in key transitions during embryogenesis. The authors applied local and global convolution techniques to detect sharp transitions in transcript levels, which enabled the classification of distinct expression classes (Figure 4):

- Class I (maternal): Genes with high initial expression that decreases over time;
- Class II (transient): Genes with increased and decreased transcript levels during embryogenesis;
- Class III (activated): Genes that are activated by transcription during embryogenesis without a subsequent decline.

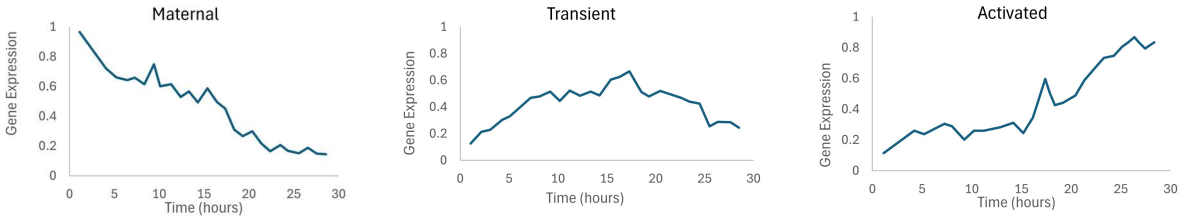

Figure 4 - Expression profiles of the three GSE6186 classes (maternal, transient, activated). Adapted from Tripto et al. (2020).

Out of the 12,868 genes measured in the dataset, 3,379 were assigned to the functional classes defined by Hooper et al. (2007): 1,534 maternal, 792 transient, and 1,053 activated. These labelled genes provide a structured framework for studying time-dependent regulation, offering a high-resolution time series format that supports the identification of

co-expression patterns. The remaining genes were not assigned to any class, as they did not exhibit sufficiently distinct temporal expression profiles.

The GSE6186 dataset has been used in a variety of studies exploring time-resolved gene expression in *D. melanogaster*. Costello et al. (2009) integrated GSE6186 into a large-scale analysis of functional gene networks, using temporal co-expression patterns to improve gene function prediction across the genome. More recently, Tripto et al. (2020) employed the dataset to benchmark deep learning and traditional machine learning models for the classification of expression patterns across time, highlighting its value as a structured time series resource.

The original GEO Series Matrix file (*.txt)* was converted to *.csv* after removal of metadata lines and extraction of the expression matrix. Column names were standardised to the "X–Yh" format, representing the time intervals. Gene identifiers, initially given as probe codes (*ID_REF*), were mapped to FlyBase biological identifiers (*FBgn*) using the GPL4455 platform annotation table.

Class labels were obtained from a supplementary file to the original article, provided in *.xls* format, which contained multiple sheets with subsets of genes. Based on the sheet names (e.g., Class I_a, II_b), genes were grouped into the three categories: maternal, transient, and activated. A consolidated file was then constructed with FBgn identifiers and corresponding functional classes. Genes without a class assignment were discarded, and records with inconsistent labels were removed — specifically, thirty-two duplicate rows with conflicting annotations were eliminated.

The final dataset contains one row per gene, with a total of 3,379 genes. The columns include the original probe identifier (*ID_REF*), the FlyBase gene ID (*FID*), the assigned functional class and expression values measured at 28 consecutive time intervals spanning the full 24-hour embryogenesis period, as shown in Figure 5 (partial representation).

Figure 5 - Final structure of the GSE6186 dataset.

ID_REF	FID	class	1-2h	1.5-2.5h	2-3h	2.5-3.5h	3-4h	3.5-4.5h	4-5h	4.5-5.5h
KP01833-411	FBgn0035059	maternal	0.9451	0.3542	0.4482	0.3231	0.315	0.224	0.195	0.439
KP01498-430	FBgn0034800	maternal	0.8857	0.8538	0.4215	0.7335	0.2216	-0.0796	-0.0714	-0.134
KP02040-750	FBgn0035205	maternal	0.9857	0.8716	1.4425	0.6581	1.0845	1.0206	1.3018	1.2302
KP04012-750	FBgn0036642	transient	1.0538	-0.0705	-0.0068	0.1444	0.4482	0.3896	0.4078	0.4383
KP06831-542	FBgn0031717	activated	-0.577	-0.3795	-0.4131	-0.2614	-0.1365	0.0667	0.181	0.2347
KP00291-554	FBgn0040395	transient	-0.1146	-0.2934	-0.3129	-0.2781	-0.3566	-0.1461	-0.1483	-0.0679
KP00242-258	FBgn0023515	maternal	0.7485	-0.0842	-0.1042	-0.9195	-0.9574	-0.6967	-0.6199	-0.4894
KP05672-750	FBgn0005616	activated	-0.4553	0.5044	0.2706	0.387	0.5561	0.6608	1.2362	0.359
KP07666-417	FBgn0003313	activated	-0.374	0.9889	0.7225	1.701	2.4109	2.3937	2.7848	2.7793

3.2.4. Comparison Between Datasets

The three datasets used in this work have different biological contexts, experimental structures, and characteristics of the classification task. Table 1 summarises their main characteristics, including the organism studied, the presence of temporal structure, the number of genes and samples and the nature of the classification task.

Table 1 - Biological and structural characteristics of the GSE3406, GSE1723 and GSE6186 datasets

Dataset	Organism	Biological Focus	Genes	Classes in original study	Temporal
GSE3406	*Saccharomyces* spp.	Response to stress stimuli	13,056	4 species	Yes
GSE1723	*S. cerevisiae*	Nutrient & oxygen limitation	9,326	8 conditions	No
GSE6186	*D. melanogaster embryos*	Embryonic development	12,868 (3,379 labelled)	3 expression-based classes	Yes

Each dataset supports a different classification scenario:

- GSE3406: multiclass, single label classification of species based on response to environmental stimuli;
- GSE1723: binary classification under different nutrient limitation conditions;
- GSE6186: multiclass, single label classification of temporal gene expression patterns during embryogenesis.

The selection of these datasets enables an evaluation of the models' performance in different biological systems, expression patterns, and data structures. Together, they cover static and time-dependent gene expression, binary and multiclass classification tasks, and varying levels of complexity in experimental design. This diversity provides a basis for evaluating the generalisability and adaptability of machine learning models to distinct types of transcriptomic data.

3.3. Common Preprocessing

Initially, data considered to be uninformative, columns and rows composed exclusively of zeros or null values were removed. The missing values were then dealt with and filled in by linear interpolation: in cases where there were valid values to the left and right of the null

position, interpolation was done on the basis of adjacent points; when the missing values were at the beginning or end of the series, the average of the two closest available points was used (Asyali et al., 2006; Troyanskaya et al., 2001).

After processing the missing values, the data was reorganised to integrate the expression tables with the experimental metadata, so that each sample was associated with its respective class (species, environmental condition, or expression profile) and structured in a format suitable for subsequent modelling. To avoid introducing bias in the distribution of the samples, the data was also shuffled to create a randomised layout before being split into training, validation, and testing. Finally, the processed datasets were generated in *.csv* format, containing only the data necessary for the modelling phase. All processing was carried out in Python (Python Software Foundation, 2024), with the help of the pandas[2] and NumPy[3] libraries.

3.4. Validation Strategy and Data Splitting

Model evaluation was conducted using two distinct validation strategies, adapted to the specific objective of each experiment. In all cases, measures were taken to ensure data representativeness and to support future reproducibility.

3.4.1. Hold-Out Validation

In the majority of experiments, model evaluation followed a fixed split strategy (hold-out), where the dataset was divided into 70% for training, 10% for validation, and 20% for testing.

This split ratio was chosen as it was the same used by Tripto et al. (2020), thereby ensuring consistency with previous work and allowing direct comparison between the results obtained in this dissertation and those reported in the literature.

3.4.2. K-Fold Cross-Validation

Cross-validation was applied specifically to the gene-level consistency analysis performed with the GSE3406 dataset. A stratified 5-fold strategy was used, with 20% of the training partition further set aside as an internal validation subset. Cross-validation was essential in this case not only to avoid bias from a single train/test split, but also to ensure that the gene-

[2] https://pandas.pydata.org
[3] https://numpy.org

level analysis covered the entire dataset, allowing all genes to be evaluated under test conditions across multiple partitions.

The adopted validation strategies ensured that the test set remained entirely isolated from the training process and was used exclusively to assess the models' generalisation capability. This procedure provides a reliable and unbiased evaluation for the modelling studies conducted.

3.5. Data Preparation for Modelling

Before the modelling phase, a set of preparation procedures common to all datasets and architectures was applied. These steps aimed to ensure data consistency, reproducibility of results, and alignment with the technical requirements of the models used.

3.5.1. Normalisation

Gene expression data were normalised using z-score transformation, where each feature was rescaled based on its mean and standard deviation. To prevent information leakage between partitions, normalisation parameters were computed exclusively from the training set and then applied to the validation and test sets, following standard recommendations (Hastie et al., 2009; Pedregosa et al., 2012).

3.5.2. Class Encoding

The categorical output labels were converted into integer indices using the *LabelEncoder* class from *scikit-learn*[4]. This ensured compatibility across all the supervised learning algorithms employed in this study. In CNN and LSTM models, this encoding was required by the *SparseCategoricalCrossentropy* loss function of Tensorflow[5] Keras[6] (Abadi et al., 2016; Chollet, 2015), which expects integer-encoded class labels rather than one-hot vectors. For SVM and XGBoost, the integer encoding was equally necessary, as both algorithms operate on discrete numerical class labels rather than string identifiers.

3.5.3. Input Structuring

For CNNs and LSTMs, input data were reshaped into 3D tensors (*samples, features, 1*) to comply with Keras requirements. In datasets with temporal structure, such as GSE3406 and

[4] https://scikit-learn.org/stable
[5] https://www.tensorflow.org
[6] https://github.com/fchollet/keras

GSE6186, the features represented time steps with a single feature per step (*n_samples, timesteps, 1*). In contrast, GSE1723 contained static measurements without sequential order, and was therefore reshaped as (*n_samples, n_features, 1*), where the features correspond to independent gene expression values rather than time steps.

The remaining models — SVM and XGBoost — were trained on standard two-dimensional input matrices, with each row representing a sample and each column a feature.

3.5.4. Seeds and Reproducibility

Reproducibility was ensured by consistently controlling all sources of randomness — including data splitting, sample shuffling, and neural network weight initialisation. Random seeds were explicitly set across the main pseudo-random number generators used in the project: *np.random.seed*, *tf.random.set_seed*, and *random.seed*, all with the value 42. This setup ensured consistent results across runs, provided that all other conditions remained unchanged.

Across all architectures, multiple independent training runs were performed (see Section 4.1.4), each with a distinct random seed. A user-defined base seed (*initial_seed = 1*) was employed and for each run, it was incremented based on the iteration index (*current_seed = initial_seed + run_index*). This introduced controlled variability across runs, mitigated overfitting to a single initialisation, and supported a more reliable estimation of average model performance.

3.5.5. Class Balancing

All datasets used in this study exhibited a balanced class distribution, except for GSE6186. In this case, a marked imbalance was observed among the three functional classes — maternal, transient, and activated — with the maternal class clearly overrepresented.

To reduce the risk of bias towards the majority class, an automatic class weighting strategy was applied, based on the relative frequency of each class in the training set. This was implemented using the *compute_class_weight* function from scikit-learn, which assigns weights inversely proportional to class frequencies. The resulting weights were approximately 1.5150 for the activated class, 0.7282 for maternal, and 1.0345 for transient.

Class weights were incorporated into the CNN training process for GSE6186 only. For the other datasets, class balancing was not required, as the class distributions were already equal.

3.6. Problem Definition

GSE3406

The GSE3406 dataset defines a multiclass, single label classification problem. The primary objective is to predict the species of each expression profile from its transcriptional response, and to assess whether comparable accuracy can be achieved when classification is performed using individual stimuli or reduced combinations of stimuli instead of the complete set.

In addition to this classification task, a secondary problem was formulated specifically for this dataset: the identification of genes that are consistently well classified or consistently misclassified across repetitions and cross-validation folds. This provides insight into which features (genes) carry stable discriminatory signals and which contribute to systematic misclassification, offering potential for dimensionality reduction and biological interpretation.

This secondary task was carried out only on the GSE3406 dataset, as it represents the largest and most complex dataset in this study, both in terms of gene count and experimental diversity. The analysis was performed independently for each stimulus, and the resulting sets of consistently well or poorly classified genes were subsequently compared to obtain their intersection across all five stimuli.

GSE1723

The GSE1723 dataset poses a binary classification task, in which the goal is to distinguish between aerobic and anaerobic conditions based on the gene expression profiles of *Saccharomyces cerevisiae* cultivated under different nutrient limitations. The analysis investigates whether classification performance can be maintained when only subsets of nutrient conditions are used, thereby evaluating the feasibility of reducing experimental complexity without compromising accuracy.

GSE6183

The GSE6186 dataset represents a multiclass, single label classification task, where genes are assigned to one of three expression classes based on their temporal expression profiles during embryogenesis. The analysis aims not only to classify genes from the full 28-point series but also to determine whether comparable accuracy can be achieved when using shortened or subsampled versions of the time series, thereby evaluating the impact of temporal reduction on classification performance.

This page was intentionally left blank

4. Modelling Approaches

This chapter details the strategies and evaluation procedures adopted for gene expression classification, focusing on the training of different machine learning and deep learning models.

4.1. General Modelling Strategy

The modelling process was designed to be modular, reproducible, and comparable across all datasets. Each architecture — CNN, LSTM, SVM, and XGBoost — was trained using a structured workflow that included hyperparameter optimisation, validation strategies, statistical repetition, and evaluation. This section details the motivation for selecting each model type, the adopted tuning and evaluation procedures, and the settings that ensured consistency across experiments. An overview of the modelling workflow is presented in Figure 6.

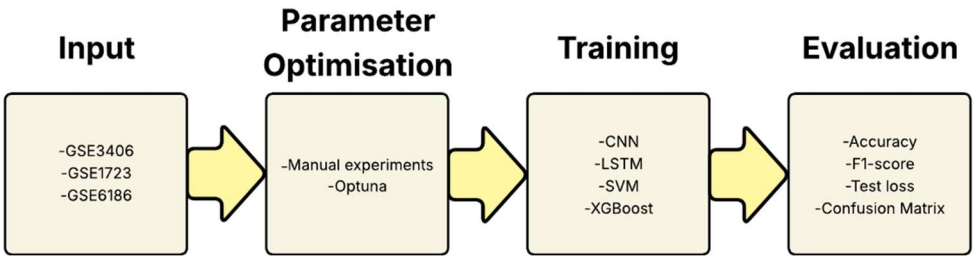

Figure 6 - Overview of the modelling workflow.

4.1.1. Justification for the Models Used

The selection of models in this dissertation was based on the nature of the gene expression data, as well as the need to evaluate approaches with different characteristics. The four models explored enable a comparison between deep neural networks, classical supervised methods, and tree-based ensemble algorithms, and this combination also mirrors the experimental design adopted by Tripto et al. (2020) which serves as a methodological reference for this dissertation.

Convolutional Neural Networks: CNNs were chosen due to their strong performance in previous gene expression classification studies (Mostavi et al., 2020; Zeng et al., 2016), and particularly as demonstrated in the work of Tripto et al. (2020). They are especially effective

at detecting spatial patterns and local trends, even in temporal data that have been reorganised into a matrix format, as is the case with the datasets used here.

Long Short-Term Memory: LSTMs were included due to their ability to model sequences and capture long-term temporal dependencies. This type of neural network is particularly well suited to time series data, such as those in the GSE6186 and GSE3406 datasets (Karim et al., 2018; Tripto et al., 2020).

Support Vector Machines: SVMs are widely used in classification tasks involving high-dimensional datasets with limited sample sizes, which is a typical scenario in gene expression analysis (Guyon et al., 2002; Hira & Gillies, 2015). In this dissertation, SVMs are employed as a classical baseline for comparison with deep learning models.

XGBoost: XGBoost was selected as it is one of the most effective ensemble algorithms for classification tasks on tabular data, particularly in high-dimensional contexts (Chen & Guestrin, 2016) similarly to SVMs. Its use allows for the evaluation of decision tree–based methods with built-in regularisation and high computational efficiency (Deng et al., 2022).

Each of these models was tested on the three datasets using the complete data, with the aim of comparing their performance and identifying the most suitable architecture for each case. The model selected for the subsequent data reduction experiments (involving combinations of stimuli, nutrients, or time points) was determined based on the results obtained with the full datasets.

4.1.2. Hyperparameter Optimisation Process

Before applying automated optimisation methods, preliminary manual experiments were conducted to narrow the search ranges to relevant regions and avoid excessively broad or uninformative spaces. These ranges were defined through exploratory testing and adapted to each architecture, drawing inspiration from previous studies in related contexts (Mostavi et al., 2020; Tripto et al., 2020; Zeng et al., 2016), although these did not use automated methods such as Optuna[7] (Akiba et al., 2019). The refined search ranges for each model and dataset, determined after preliminary experiments to narrow down the initial values, are presented in Table 2, Table 3, and Table 4. The complete set of hyperparameters obtained is provided in Appendix A - Hyperparameter Optimisation Results.

[7]https://optuna.org

Table 2 - Optuna optimization intervals for the GSE3406 dataset

Model	Parameter	Optimization Interval	Sampling Distribution
CNN	*learning_rate*	1e-7 — 1e-2	log-uniform
	dropout_rate	0.1 — 0.3	uniform
	L2_regularization	1e-6 — 1e-2	log-uniform
	batch_size	[64, 128, 256]	categorical
LSTM	*learning_rate*	1e-4 — 1e-1	log-uniform
	dropout_rate	0.25 — 0.4	uniform
	L2_regularization	1e-6 — 1e-3	log-uniform
	batch_size	[256, 512]	categorical
SVM	*C*	1e-4 — 1e2	log-uniform
	kernel	[linear, rbf, poly, sigmoid]	categorical
	gamma	1e-4 — 1	log-uniform
	degree	2 — 5	integer
XGBoost	*learning_rate*	2e-1 — 3.3e-1	log-uniform
	max_depth	5 — 7	integer
	n_estimators	430 — 500	integer
	subsample	0.8 — 0.9	uniform
	colsample_bytree	0.7 — 0.8	uniform
	reg_alpha	0.5 — 2.5	uniform
	reg_lambda	0.5 — 1.5	uniform
	gamma	0.0 — 0.2	uniform

Table 3 - Optuna optimization intervals for the GSE1723 dataset

Model	Parameter	Optimization Interval	Sampling Distribution
CNN	*learning_rate*	1e-6 — 3e-4	log-uniform
	dropout_rate	0.15 — 0.35	uniform
	L2_regularization	1e-6 — 1e-4	log-uniform
	batch_size	[64, 128, 256]	categorical
LSTM	*learning_rate*	1e-7 — 1e-2	log-uniform
	dropout_rate	0.1 — 0.3	uniform
	L2_regularization	1e-6 — 1e-2	log-uniform
	batch_size	[64, 128, 256]	categorical
SVM	*C*	1e-3 — 1e3	log-uniform
	kernel	[linear, rbf, poly, sigmoid]	categorical
	gamma	1e-3 — 5e0	log-uniform
	degree	2 — 5	integer
XGBoost	*learning_rate*	1e-3 — 3e-1	log-uniform
	max_depth	3 — 12	integer
	n_estimators	50 — 400	integer
	subsample	0.5 — 1	uniform
	colsample_bytree	0.5 — 1	uniform
	reg_alpha	0 — 10	uniform
	reg_lambda	0 — 10	uniform
	gamma	0 — 5	uniform

Table 4 - Optuna optimization intervals for the GSE6186 dataset

Model	Parameter	Optimization Interval	Sampling Distribution
CNN	*learning_rate*	1e-7 — 1e-2	log-uniform
	dropout_rate	0.4 — 0.9	uniform
	L2_regularization	1e-9 — 1e-4	log-uniform
	batch_size	[16, 32, 64, 128, 256]	categorical
LSTM	*learning_rate*	1e-7 — 1e-2	log-uniform
	dropout_rate	0.3 — 0.9	uniform
	L2_regularization	1e-9 — 1e-4	log-uniform
	batch_size	[16, 32, 64, 128, 256]	categorical
SVM	*C*	1e-4 — 1e2	log-uniform
	kernel	[linear, rbf, poly, sigmoid]	categorical
	gamma	1e-4 — 1e0	log-uniform
	degree	2 — 5	integer
XGBoost	*learning_rate*	3e-2 — 1e-1	log-uniform
	max_depth	6 — 11	integer
	n_estimators	200 — 400	integer
	subsample	0.85 — 1	uniform
	colsample_bytree	0.5 — 0.7	uniform
	reg_alpha	5 — 8	uniform
	reg_lambda	2.5 — 5 (uniform)	uniform
	gamma	0 — 0.5 (uniform)	uniform

Within this process, certain architectural parameters — such as the number of convolutional and dense layers in CNNs or the number of units in LSTM models — were fixed according to empirical results from initial testing. The adopted structures are shown in Figure 7, which shows the sequential arrangement of layers, activation functions, and neurons defined for each CNN and LSTM architecture.

Based on preliminary testing, batch normalization was applied in the CNNs immediately after each Conv1D layer and before dropout to improve gradient stability and convergence. By contrast, in LSTM models it was used selectively—applied only when training remained stable—in line with reports that batch normalization in sequential models can introduce instability or offer limited gains (Laurent et al., 2015; Santurkar et al., 2019).

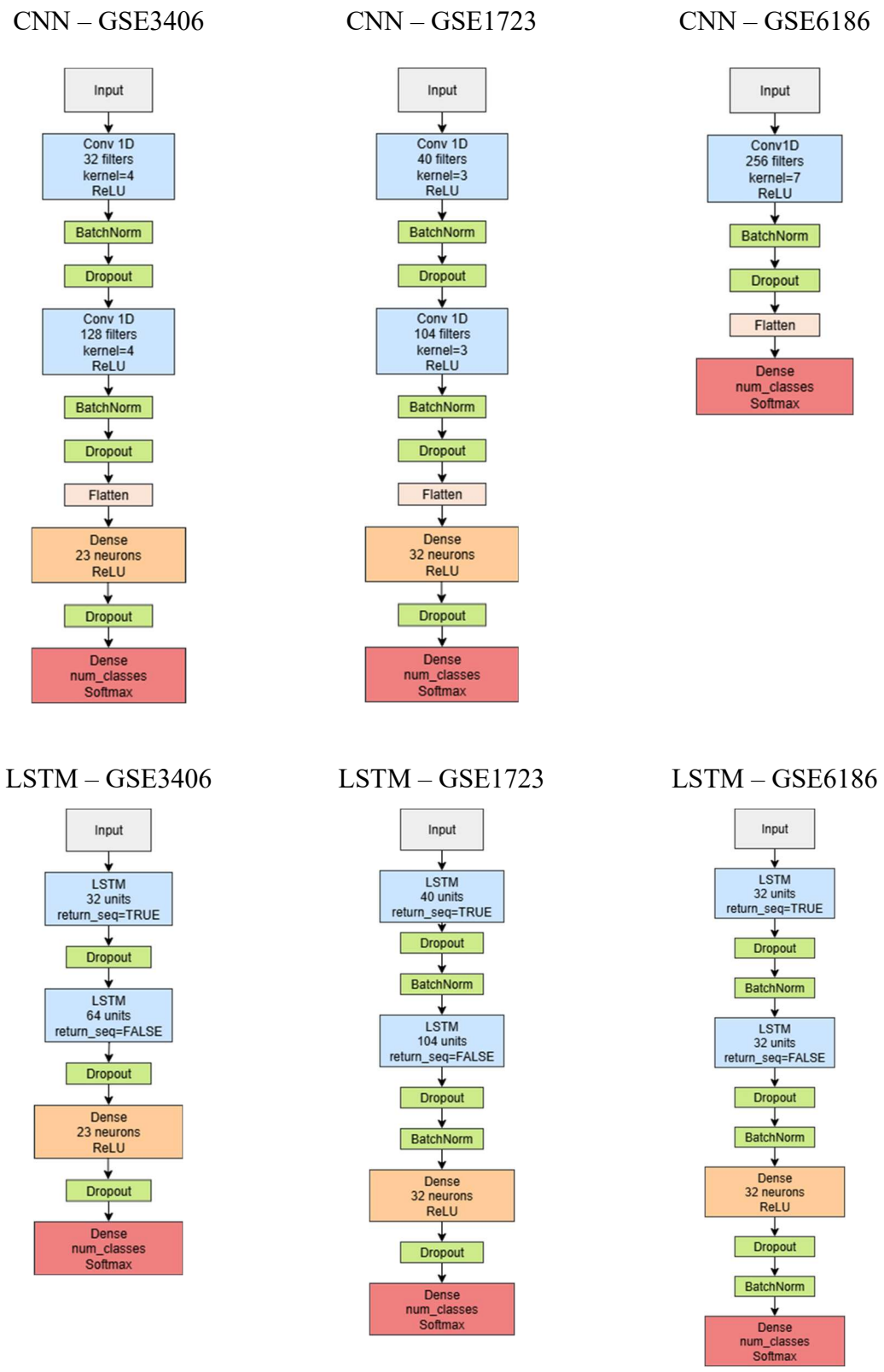

Figure 7 - Fixed architectures adopted for each dataset and model.

For the SVM and XGBoost models, since their structure is not layer-based, no fixed architecture was defined. Instead, only their main hyperparameters were subject to optimisation.

Hyperparameter tuning was then performed separately for each architecture. Within each architecture, individual searches were conducted for each combination using the Optuna library, which employs the Bayesian optimisation algorithm TPE (Tree-structured Parzen Estimator).

For neural networks, the objective of the evaluation function was to minimise the validation loss. Both *Adam* and *RMSProp* optimisers were tested during preliminary experiments, and *Adam* consistently achieved the best performance across all datasets. The hyperparameters optimised in these models were the *learning rate*, *batch size*, *dropout rate*, and *L2 regularisation* factor.

For the SVM model, validation accuracy was used as the evaluation metric. This was chosen instead of validation loss, as SVMs optimise a margin-based objective internally (hinge loss), but do not expose a direct loss function for validation purposes. The optimised parameters were the kernel type, the penalty parameter *C*, the *gamma* value (for non-linear kernels), and, when applicable, the *polynomial degree*.

For XGBoost, the objective function aimed to minimise the validation log loss. The optimised hyperparameters were the *maximum tree depth*, *learning rate*, *number of estimators*, sampling fractions (*subsample*, *colsample_bytree*), as well as regularisation coefficients (*reg_alpha*, *reg_lambda*, *gamma*).

In addition to the standard optimisation procedure, stratified 5-fold cross-validation was implemented inside each Optuna trial to perform hyperparameter optimisation for the CNN on the GSE3406 dataset. With this setup, each sample (gene) appeared in the validation set exactly once across the five folds, ensuring that all genes were used under validation conditions. At the end of each trial, the mean validation loss across the five folds was returned as the objective value, thereby providing a more stable and generalisable estimate of performance (Kohavi, 1995). A hold-out split would have been less appropriate in this context, as it would have left out part of the genes during validation.

Each optimisation study consisted of sixty trials per model and data configuration. The best sets of hyperparameters were selected based on validation performance and subsequently used in the final model training.

4.1.3. Model Training Procedures

Each training run for the CNN and LSTM models was limited to a maximum of 1000 epochs. However, training was always interrupted earlier due to the use of two callbacks: *early_stopping* with a patience of 36 epochs and learning rate reduction on plateau (*ReduceLROnPlateau*) with a patience of 20 epochs. These values were defined empirically based on preliminary tests. The 20-epoch patience allowed the optimiser to gradually reduce the learning rate before stopping, while the 36-epoch threshold ensured tolerance to temporary fluctuations in validation loss, avoiding premature termination. In all cases, the model's best weights (based on validation loss) were automatically restored at the end of training.

4.1.4. Repetition Strategy and Statistical Considerations

To ensure that the results obtained were representative, stable, and comparable across configurations, all experiments were conducted using 60 independent training repetitions per model and configuration. Each model was trained from scratch in every repetition.

The results of all repetitions were recorded, allowing for the computation of descriptive statistics, namely the mean and standard deviation of key performance metrics (*accuracy* and *F1-score*). This statistical characterisation provides insight into the variability and expected performance of each model under repeated training conditions.

This strategy is grounded in the Law of Large Numbers (LLN), which guarantees that, as the number of independent trials increases, the sample mean converges to the true expected value of the underlying distribution (Casella & Berger, 2002). Furthermore, the Central Limit Theorem (CLT) supports the assumption that, given a sufficient number of repetitions, the distribution of the sample means approximates a normal distribution, regardless of the shape of the underlying distribution (Wasserman, 2004).

In this study, sixty repetitions were empirically chosen as a trade-off between statistical reliability and computational feasibility. Preliminary experiments indicated that performance metrics exhibited limited variability beyond this number, suggesting convergence of the sample mean and stabilisation of the standard deviation. This choice is

also consistent with repetition strategies observed in empirical studies addressing stochastic training behaviour and generalisation variability in deep learning models (Bouthillier et al., 2021; Jordan, 2023).

In the specific case of the GSE3406 dataset, where gene-level classification consistency was studied, each configuration underwent 5-fold cross-validation repeated 60 times, resulting in a total of 300 trained models per configuration.

This methodological framework was designed to reduce the influence of individual training anomalies and to ensure that the performance reported reflects general trends rather than isolated outcomes.

4.1.5. Evaluation Metrics

Several evaluation metrics were applied to quantify model performance across different architectures and datasets, balancing global accuracy with class-wise discrimination and probabilistic error.

Model performance was primarily assessed using accuracy, weighted F1-score, and the confusion matrix, all standard tools for evaluating multiclass classification tasks. These metrics were computed for each trained model across all repetitions.

The accuracy metric quantifies the proportion of correct predictions among all evaluated samples but does not account for potential class imbalance. To address this limitation, the weighted F1-score was also reported, as it incorporates both precision and recall for each class, weighted according to class frequency. This is particularly relevant for datasets with class imbalance, such as GSE6186, where accuracy alone could lead to misleading conclusions.

Additionally, the confusion matrix was generated for all models. This provides a detailed view of prediction distributions across classes, allowing for the identification of systematic misclassifications or class confusion.

The validation loss was used during training to monitor model improvement and was the criterion for both early stopping and learning rate reduction. In all cases, the model weights corresponding to the epoch with the lowest validation loss were restored at the end of training.

For SVM models, predictions were also evaluated using accuracy, weighted F1-score, and confusion matrix. To obtain probabilistic outputs, the option *probability=True* was set in the SVC classifier, which applies Platt's method for probability calibration (Platt, 1999).

For XGBoost models, in addition to accuracy, weighted F1-score, and the confusion matrix, the multiclass logarithmic loss (*mlogloss*) was computed and used as the training objective. The *multi:softprob* objective function returns a full probability vector over all classes for each prediction, allowing *mlogloss* to penalise overconfident incorrect predictions more severely and thereby provide a more informative performance indicator.

4.2. Modelling with GSE3406

The modelling experiments with GSE3406 focused on comparing alternative machine learning and deep learning architectures and assessing how different stimulus combinations influenced classification accuracy. A series of systematic experiments was conducted to provide an in-depth analysis of how each condition affected model performance.

For LSTM, in this specific dataset, four alternative input strategies were evaluated to investigate how temporal alignment affects sequential modelling. This adjustment was particularly relevant since nitrogen starvation presented seven measurements but only three time points coincided with the other four stimuli. The strategies were:

- Flat — a flat representation where all columns were concatenated and treated as a single sequence;
- T6-Interpolated — a fixed six-point temporal grid, with interpolation applied only to nitrogen starvation at the time points that did not match the other stimuli;
- Intersect — a strict intersection of time points across all stimuli, resulting in only three common points while discarding the others;
- Index-Based — an index-based selection of six points per stimulus, without interpolation and excluding the last nitrogen starvation measurement (1440 minutes).

4.2.1. Model Comparison on Full Dataset

Initially, the four models under consideration — CNN, LSTM, SVM, and XGBoost — were trained using the full dataset, i.e., with all five stimuli simultaneously. Each model was independently trained 60 times with distinct seeds, and the final performance metrics were computed as the mean and standard deviation across these repetitions. This initial

comparison established the baseline performance of each architecture under the complete dataset.

4.2.2. Stimulus Combination Strategy

For each subsequent experimental configuration, a dedicated hyperparameter optimisation was performed using Optuna, and the resulting best parameters were employed to train the sixty repetitions for that configuration. To assess the influence of environmental stimuli on classification performance, the following configurations were tested:

- Individual stimuli: One stimulus at a time (5 configurations);
- Pairs of stimuli (2×2): All 10 possible pairwise combinations;
- Triplets of stimuli (3×3): All 10 possible combinations of three stimuli;
- Quadruplets of stimuli (4×4): All 5 combinations of four stimuli;
- All five stimuli: Full dataset.

Each configuration was treated as an independent training scenario, and the dataset was filtered to include only the selected stimuli.

4.2.3. Gene-Level Consistency Analysis

In addition to evaluating classification performance at the sample level, a gene-level consistency analysis was conducted focusing exclusively on the five individual stimulus configurations. The full dataset with all five stimuli combined was not used for this purpose. Instead, for each stimulus, classification performance was evaluated using 5-fold cross-validation, ensuring that each gene–species sample was used exactly once for testing. The results from these individual analyses were then intersected to identify genes that remained consistently correctly or incorrectly classified across all stimuli.

For each stimulus, a stratified 5-fold cross-validation procedure was applied, with each fold repeated sixty times — resulting in three hundred trained models per stimulus. For each gene, the number of times it was correctly classified or misclassified across all folds and repetitions was recorded. This enabled the identification of:

- Always correctly classified genes, which may exhibit strong species-specific expression patterns under that stimulus;
- Consistently misclassified genes, which may be ambiguous or non-informative in that context.

Cross-stimulus comparison allowed the identification of genes that consistently exhibited stable discriminatory power or persistent ambiguity, thus revealing candidates for simplified input representations.

4.3. Modelling with GSE1723

This section presents the modelling experiments conducted with the GSE1723 dataset, focusing on model comparison and on evaluating whether classification performance could be preserved when restricting the analysis to subsets of nutrient environments.

4.3.1. Model Evaluation on the Complete Dataset

The four candidate models were first trained on the complete dataset, covering all four nutrient limitations under both aerobic and anaerobic regimes. Each architecture was evaluated over 60 independent runs, and the resulting mean and standard deviation of the metrics provided an initial benchmark for subsequent analysis.

4.3.2. Nutrient Combination Strategy

Dedicated training runs were performed using subsets of the data corresponding to individual nutrients, as well as all possible pairwise and triplet combinations.

To investigate how individual nutrients contributed to distinguishing oxygen regimes, the following configurations were assessed:

- Individual nutrients: Carbon, nitrogen, phosphorus, sulphur (4 configurations);
- All possible pairs (2×2): 6 combinations;
- Triplets (3×3): 4 combinations;
- All four nutrients: Full dataset.

This approach enabled the analysis of whether classification accuracy could be preserved under reduced experimental conditions and highlighted which nutrient environments provided the strongest oxygen-related transcriptional signals.

4.4. Modelling with GSE6186

The final set of experiments was conducted using the GSE6186 dataset, focusing on model comparison and on assessing whether predictive performance could be preserved when using shortened or subsampled time series.

4.4.1. Model Evaluation on Full Time Series

The experimental design mirrored that of the previous datasets: first, comparing CNN, LSTM, SVM, and XGBoost. In this case the input consisted of a 28-point time series. Second, evaluating whether comparable results could be obtained with reduced temporal inputs.

4.4.2. Reduced Input Strategy

To assess whether classification performance could be preserved with fewer time points, a second set of experiments was conducted using truncated or subsampled versions of the time series. The following configurations were evaluated:

- First quarter of the sequence (7 time points);
- First third (9 time points);
- First half (14 time points);
- Alternating time points up to halfway (14 time points: column yes/column no to mid);
- Alternating time points across the full sequence (14 time points: column yes/column no to end).

This experimental design enabled the assessment of how temporal compression affected predictive performance and whether a shorter or sparser measurement schedule yield performance comparable to the full 28-point series.

4.5. Tools and Libraries Used

The implementation and training of the models were carried out in Python (version 3.12.5), using libraries in the fields of data science and machine learning.

From the scikit-learn library (Pedregosa et al., 2012), the *train_test_split* function was used to divide the data into training, validation, and test sets and the *SVC* class was employed to implement the support vector machine models, with the *probability=True* option enabled to provide probabilistic outputs. The *compute_class_weight* utility was also used in the

GSE6186 dataset to address class imbalance, with the resulting weights passed to the training processes via the *class_weight* parameter in Keras.

From the XGBoost library, the *XGBClassifier* class was used to implement gradient-boosted decision trees.

The CNN and LSTM models were implemented using TensorFlow Keras with the Sequential API. Regularisation and training control techniques — including batch normalisation, dropout, early stopping, and learning rate reduction on plateau — were incorporated using standard Keras layers and callbacks.

Hyperparameter tuning was conducted using the *Optuna* library, relying on the TPE (Tree-structured Parzen Estimator) algorithm.

Additionally, the *NumPy* and *Pandas* libraries were used for data manipulation and preparation of input structures. For result visualisation, including learning curves and confusion matrices, *Matplotlib*[8] (Hunter, 2007) and *Seaborn*[9] (Waskom, 2021) were employed.

All experiments were conducted in a local environment equipped with an Intel Core i7-12700H (12th generation) processor, 32 GB of RAM, and an NVIDIA RTX 3060 Laptop GPU, running Windows 11 Pro (64-bit). GPU acceleration was enabled using the TensorFlow backend with CUDA support.

[8] https://matplotlib.org
[9] https://seaborn.pydata.org

This page was intentionally left blank

5. Results and Discussion

This chapter presents and analyses the results obtained across the different classification scenarios studied, organised in a dataset-based structure to highlight their biological and computational particularities. Each subsection discusses the average performance and variability of the tested architectures, the impact of reducing stimuli, nutrients, or time points on classification, and critically examines consistent patterns, limitations, and practical implications — with particular attention to strategies for reducing experimental and computational costs. Finally, a global comparison of the three datasets is provided in the context of the existing literature.

5.1. General Results Overview

The global performance of the models across the three datasets is summarised before presenting the detailed analyses. All experiments were repeated 60 times per configuration to ensure statistical reliability, with results reported as mean ± standard deviation. The main evaluation metrics considered were accuracy and weighted F1-score, complemented by confusion matrices for qualitative error analysis. Execution times for both optimisation and training were also recorded, and the detailed values are provided in Appendix B - Execution Times for Optimization and Training.

Table 5 provides an overview of the best-performing architecture in each dataset when trained with the complete set of data — all stimuli for GSE3406, all nutrients for GSE1723, and all time points for GSE6186. The results indicate that CNN achieved the highest performance in GSE3406. In GSE1723, CNN and XGBoost produced comparable results, with a slight advantage for CNN, while in GSE6186 LSTM marginally outperformed the other models. These differences highlight how dataset characteristics — such as temporal structure or class complexity — influence the relative effectiveness of each architecture.

Table 5 - Summary of best-performing models for each dataset using the complete data.

Dataset	Best Model	Accuracy (mean ± std)	F1-score (mean ± std)
GSE3406	CNN	0.9510 ± 0.0014	0.9509 ± 0.0014
GSE1723	CNN	0.8730 ± 0.0036	0.8730 ± 0.0037
GSE6186	LSTM	0.9461 ± 0.0053	0.9460 ± 0.0051

These results identify the best-performing model per dataset, and the following sections detail how experimental reductions affected classification.

5.2. Results with the GSE3406 Dataset

The experiments with GSE3406 aimed to classify four *Saccharomyces* species under five stimuli and their combinations. This section begins with the full dataset, continues with the reduced combinations, and concludes with a gene-level consistency analysis.

5.2.1. LSTM Input Strategy

Before comparing the different model architectures, it was necessary to determine the most suitable input structure for the LSTM. Four strategies were tested to handle the temporal alignment of the five stimuli: Flat, T6-Interpolated, Intersect, and Index-Based. The results of these experiments are summarised in Table 6.

Table 6 - Performance of alternative LSTM input strategies on the GSE3406.

Strategy	Accuracy (mean ± std)	F1-Score (mean ± std)
Flat	0.9307 ± 0.0066	0.9304 ± 0.0067
T6-Interpolated	0.8571 ± 0.0030	0.8566 ± 0.0030
Intersect (T=3)	0.6924 ± 0.0020	0.6904 ± 0.0021
Index-Based	0.8714 ± 0.0023	0.8710 ± 0.0024

Among them, the Flat strategy achieved the best performance, with an accuracy of 93.07%, clearly surpassing the other approaches. Both T6-Interpolated and Index-Based yielded lower accuracies, while the Intersect strategy, restricted to only three common time points, led to a marked loss of performance. Overall, these results indicate that the Flat representation, despite not preserving explicit synchrony between stimuli, provided the most discriminative input structure for this dataset.

5.2.2. Full Dataset (Five Stimuli)

Once the most effective input format for LSTM had been established (Flat), the next step was to compare its performance against other models. Table 7 summarises the highest results obtained for each architecture after hyperparameter optimisation. CNN achieved the best performance, with both accuracy and F1-score above 95%, while LSTM (Flat) and XGBoost

performed slightly lower, and SVM obtained the weakest results among the tested models. Given its superior results, CNN was selected as the reference architecture for the subsequent experiments.

Table 7 - Best performance of different models on the full GSE3406 dataset.

Model	Accuracy (mean ± std)	F1-score (mean ± std)
CNN	0.9510 ± 0.0014	0.9509 ± 0.0014
LSTM (Flat)	0.9307 ± 0.0066	0.9304 ± 0.0067
SVM	0.8976 ± 0.0000	0.8978 ± 0.0000
XGBoost	0.9220 ± 0.0013	0.9221 ± 0.0013

In addition to the aggregated results, the learning dynamics of the best-performing model (CNN) are illustrated in Figure 8. The training and validation curves show a steady improvement in accuracy and a corresponding reduction in loss, with no signs of overfitting, confirming the stability of the model.

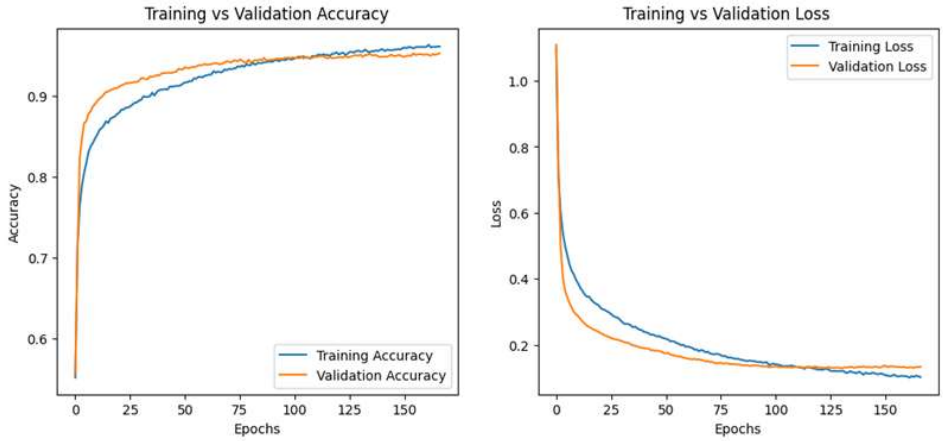

Figure 8 - Training and validation curves of the best model on the full GSE3406 dataset.

Figure 9 shows the evolution of test accuracy across 60 repetitions with the CNN model on the full GSE3406 dataset. Despite natural fluctuations between runs, the running average quickly stabilised around 95.10%, demonstrating the consistency of the training strategy.

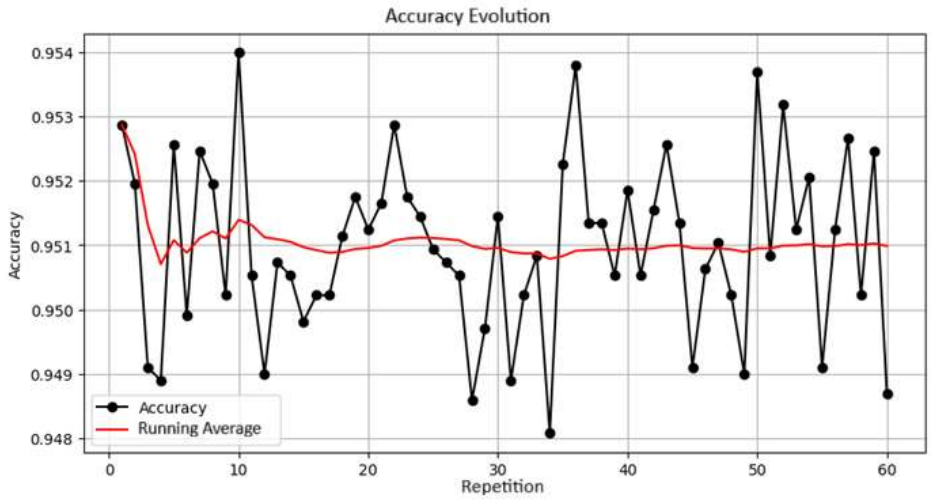

Figure 9 - Test accuracy across 60 repetitions with the CNN model on the full GSE3406 dataset.

Finally, Figure 10 presents the confusion matrix of the best model, highlighting both the overall high classification performance and the specific misclassifications observed. Most errors occurred between *S. cerevisiae* and *S. paradoxus*, two closely related species that exhibit highly conserved transcriptional responses under stress conditions (Swamy et al., 2014), which partially explains this misclassification.

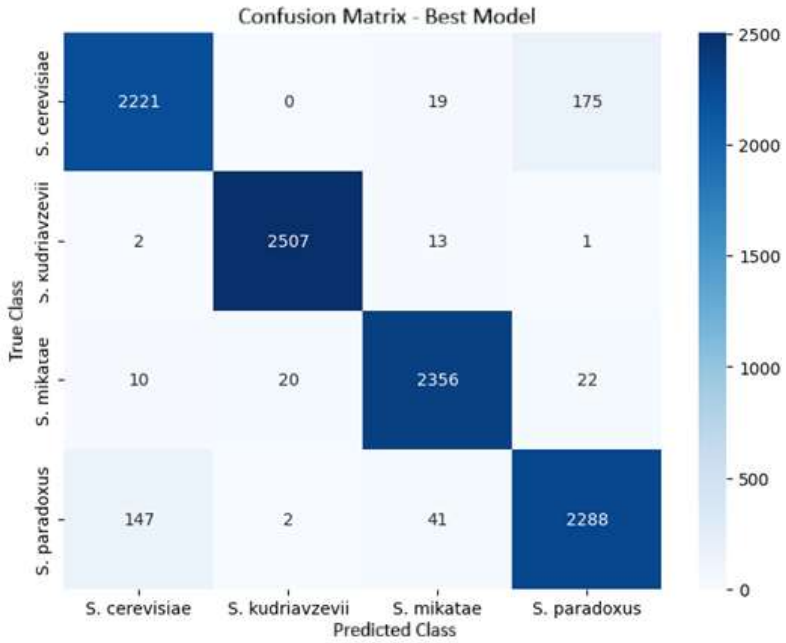

Figure 10 - Confusion matrix of the best CNN model on the full GSE3406 dataset.

5.2.3. Individual Stimuli

To assess the discriminative power of each condition separately, the CNN model was trained with individual stimuli only. The results are presented in Table 8.

Table 8 - Performance of CNN models trained on individual stimuli (GSE3406).

Stimulus	Accuracy (mean ± std)	F1-score (mean ± std)
Heat shock (HS)	0.4694 ± 0.0021	0.4654 ± 0.0023
H_2O_2	0.7065 ± 0.0021	0.7047 ± 0.0022
MMS	0.5996 ± 0.0024	0.5990 ± 0.0025
Nitrogen starvation (NS)	0.7356 ± 0.0019	0.7346 ± 0.0019
Transfer from glucose (TfG)	0.5574 ± 0.0023	0.5557 ± 0.0023

Among the individual stimuli, nitrogen starvation produced the highest accuracy (73.56%), followed by H_2O_2 (70.65%). In contrast, heat shock yielded the weakest results, with accuracy below 47%, this inferior performance may be explained by the divergence of *Saccharomyces* species in their thermal tolerance. Recent work has shown that thermophilic species such as *S. cerevisiae* and *S. paradoxus* exhibit a weaker and more rapidly resolved heat shock response, whereas cryophilic species display a stronger but more general stress response (Fay et al., 2023). By contrast, nitrogen deprivation triggers broad and well-defined transcriptional reprogramming, particularly in genes involved in protein synthesis and central metabolism, which provides clearer discriminative signals between species (Wu et al., 2004). Overall, the low scores across all single-stimulus experiments suggest that species discrimination could improve when different stimuli are combined.

5.2.4. Pairwise Combinations

To evaluate whether combining two stimuli could enhance species discrimination, the CNN model was trained on all pairwise subsets. The results are presented in Table 9.

This analysis not only allowed the identification of effects between stimuli but also provided insight into whether weaker conditions could be compensated when combined with stronger ones. In this way, pairwise evaluation served as an intermediate step between single-stimulus models and higher-order combinations.

Table 9 - Performance of CNN models trained on pairwise stimulus combinations (GSE3406).

Stimuli combination	Accuracy (mean ± std)	F1-score (mean ± std)
H_2O_2 + HS	0.7689 ± 0.0028	0.7682 ± 0.0028
H_2O_2 + MMS	0.8203 ± 0.0019	0.8199 ± 0.0018
H_2O_2 + NS	0.9003 ± 0.0022	0.8997 ± 0.0022
H_2O_2 + TfG	0.7776 ± 0.0020	0.7774 ± 0.0021
HS + MMS	0.6937 ± 0.0036	0.6928 ± 0.0037
HS + NS	0.7999 ± 0.0020	0.7996 ± 0.0021
HS + TfG	0.6444 ± 0.0028	0.6445 ± 0.0028
MMS + NS	0.8309 ± 0.0021	0.8305 ± 0.0021
MMS + TfG	0.7283 ± 0.0025	0.7273 ± 0.0025
NS + TfG	0.8411 ± 0.0025	0.8410 ± 0.0025

The best results were obtained for *H_2O_2 + nitrogen starvation*, which reached an accuracy of 90.03%, followed by *nitrogen starvation + transfer from glucose* (84.11%) and *MMS + nitrogen starvation* (83.09%). In contrast, combinations involving heat shock consistently underperformed, with the lowest result observed for *heat shock + transfer from glucose* (64.44%).

These findings confirm that pairing stimuli provides stronger discriminatory signals than individual conditions, particularly when nitrogen starvation is included. Conversely, the persistent weakness of heat shock combinations reinforces its limited utility as a standalone or complementary discriminator. Notably, the best pairwise results (*H_2O_2 + nitrogen starvation*, 90.03% accuracy) was only about 5% lower than using all five stimuli, highlighting the strong discriminatory power of this reduced subset.

5.2.5. Triple Combinations

Triple-stimulus experiments further improved classification performance compared with individual or pairwise conditions (Table 10). Most combinations that included nitrogen starvation achieved accuracies above 0.88, with two subsets — *H_2O_2 + MMS + nitrogen starvation* and *H_2O_2 + nitrogen starvation + transfer from glucose* — surpassing 92%.

In contrast, subsets involving heat shock remained weaker, rarely exceeding 87% and dropping below 78% when combined with MMS and transfer from glucose. This reinforces the limited discriminatory value of heat shock, even when used alongside other conditions.

Taken together, the results show that adding a third stimulus substantially strengthens classification and that nitrogen starvation consistently provides a decisive contribution to species separation.

Table 10 - Performance of CNN models trained on triple stimulus combinations (GSE3406).

Stimuli combination	Accuracy (mean ± std)	F1-score (mean ± std)
H_2O_2 + HS + MMS	0.8524 ± 0.0024	0.8520 ± 0.0024
H_2O_2 + HS+ NS	0.9134 ± 0.0016	0.9128 ± 0.0016
H_2O_2 + HS + TfG	0.8237 ± 0.0025	0.8236 ± 0.0025
H_2O_2 + MMS + NS	0.9260 ± 0.0016	0.9256 ± 0.0016
H_2O_2 + MMS +TfG	0.8598 ± 0.0024	0.8596 ± 0.0024
H_2O_2 + NS+ TfG	0.9277 ± 0.0020	0.9274 ± 0.0020
HS + MMS + NS	0.8619 ± 0.0024	0.8617 ± 0.0024
HS+ MMS + TfG	0.7790 ± 0.0038	0.7786 ± 0.0039
HS + NS + TfG	0.8722 ± 0.0023	0.8721 ± 0.0023
MMS + NS + TfG	0.8883 ± 0.0023	0.8882 ± 0.0023

5.2.6. Quadruple Combinations

Table 11 illustrates that using four stimuli is sufficient to reach classification scores that approach the maximum achieved with all five. Most subsets that contained nitrogen starvation consistently reached values above 93%. In practical terms, this means that nearly optimal classification can be achieved without requiring all five stimuli. Combinations without nitrogen starvation did not achieve comparable performance. The weakest case, *H_2O_2 + heat shock + MMS + transfer from glucose*, remained below 89%.

This confirms the importance of nitrogen starvation and suggests that its inclusion is more decisive for performance than the specific choice of the remaining stimuli.

Table 11 - Performance of CNN models trained on quadruple stimulus combinations (GSE3406).

Stimuli combination	Accuracy (mean ± std)	F1-score (mean ± std)
H_2O_2 + HS + MMS + NS	0.9344 ± 0.0016	0.9341 ± 0.0016
H_2O_2 + HS + MMS + TfG	0.8786 ± 0.0049	0.8784 ± 0.0050
H_2O_2 + HS + NS + TfG	0.9358 ± 0.0016	0.9355 ± 0.0016
H_2O_2 + MMS + NS + TfG	0.9430 ± 0.0014	0.9428 ± 0.0014
HS + MMS + NS + TfG	0.9078 ± 0.0024	0.9077 ± 0.0024

5.2.7. Gene-Level Consistency Analysis

In addition to dataset-level classification, a gene-level analysis was performed to identify genes that were either correctly or incorrectly classified consistently across repetitions and experimental conditions. This analysis, performed with K-Folds, focused on experiments conducted with a single stimulus at a time, rather than with combined stimuli, thereby ensuring that classification performance did not benefit from stronger stimuli compensating for weaker ones. The classifications obtained under individual stimuli were then cross-analysed to identify genes that showed consistent behaviour across all conditions.

For completeness, the results obtained with all stimuli using K-Folds are reported in Table 12, alongside those for the individual stimuli, allowing a direct comparison with the hold-out procedure (Table 7 and Table 8). Performance was remarkably similar to that reported with hold-out, indicating that the models were not overly dependent on a specific data split. This similarity likely reflects the relatively large and balanced dataset, together with the models' capacity to generalise.

Table 12 - Classification results with K-Fold cross-validation (GSE3406).

Stimulus	Accuracy (mean ± std)	F1-score (mean ± std)
Heat shock	0.4728 ± 0.0043	0.4672 ± 0.0045
H_2O_2	0.7025 ± 0.0042	0.7002 ± 0.0043
MMS	0.5774 ± 0.0045	0.5756 ± 0.0046
Nitrogen starvation	0.7348 ± 0.0040	0.7333 ± 0.0041
Transfer from glucose	0.5519 ± 0.0045	0.5501 ± 0.0046
All stimuli	0.9530 ± 0.0022	0.9529 ± 0.0022

Beyond overall performance, the analysis identified genes that were consistently classified correctly or incorrectly across all stimuli. A total of 5,820 gene–species pairs were always correctly classified, including 11 genes that were consistently recognised across all four species, suggesting strong discriminative potential. These genes, together with their identifiers, ORFs, gene symbols, and gene titles as provided in the GPL2910 platform annotation file, are shown in Table 13. In cases where no gene symbol was available in the annotation, this is indicated by "—." Conversely, 657 gene–species pairs were always misclassified, although no single gene was misclassified across all species simultaneously.

Table 13 - Genes consistently well classified across all four species (GSE3406).

ID	Platform ORF	Gene Symbol	Gene Title
2656	YMR194W	RPL36A	Ribosomal 60S subunit protein L36A
3245	YPL086C	ELP3	Elongator subunit ELP3
3481	YOR004W	UTP23	rRNA-binding ribosome biosynthesis protein UTP23
4040	YOR340C	RPA43	DNA-directed RNA polymerase I subunit RPA43
5209	YDL086W	—	Carboxymethylenebutenolidase
5807	YGL076C	RPL7A	Ribosomal 60S subunit protein L7A
6821	YBR230C	OM14	Om14p
7866	YPR110C	RPC40	DNA-directed RNA polymerase core subunit RPC40
8115	YOR168W	GLN4	Glutamine--tRNA ligase
8956	YPL257W	—	Hypothetical protein
9486	YOR312C	RPL20B	Ribosomal 60S subunit protein L20B

These results highlight two complementary aspects: the stability of the classification procedure under different validation strategies (hold-out vs K-Folds), and the presence of subsets of genes with highly consistent behaviour, either as reliable discriminators or as persistent sources of error.

5.2.8. Integrated Discussion of GSE3406 Results

The analysis of the GSE3406 dataset across various levels of stimulus combinations revealed consistent patterns that reflect the biological impact of different stimuli on classification. Using all five stimuli provided the highest performance, with accuracy and F1-scores exceeding 95%. However, several reduced configurations achieved results that were only marginally lower, indicating that near-optimal classification can be maintained with fewer experimental conditions.

Among single stimuli, nitrogen starvation, and H_2O_2 were the most informative, while heat shock consistently produced the weakest results. This trend persisted across combinations: subsets that included nitrogen starvation were systematically among the best-performing, often achieving accuracies above 90% in pairwise conditions and above 92% in triple combinations. By contrast, combinations involving heat shock frequently underperformed, reinforcing the limited discriminatory value of this condition, even when combined with others.

The analysis of quadruple combinations showed that accuracies above 94% could be achieved without all five stimuli, provided nitrogen starvation was included. This demonstrates that a strategically reduced set of conditions can closely approximate the performance of the full dataset while potentially lowering experimental costs.

The gene-level consistency analysis further identified subsets of genes that were reliably classified across all stimuli and species, as well as others that were consistently misclassified. These findings indicate the presence of stable discriminators and ambiguous signals, which may inform future feature selection strategies.

Overall, these results indicate that nitrogen starvation acts as a pivotal stimulus for distinguishing *Saccharomyces* species, while heat shock adds little value for classification purposes. More broadly, the experiments confirm that carefully chosen subsets of stimuli can reduce experimental complexity without significantly compromising accuracy.

5.3. Results with the GSE1723 Dataset

This section presents the results obtained with the GSE1723 dataset, whose purpose was to classify oxygen availability (aerobic vs. anaerobic) based on gene expression profiles under different nutrient limitations.

5.3.1. Complete Dataset (All Nutrient Conditions)

The initial experiments with the GSE1723 dataset considered all four nutrient limitations simultaneously under aerobic and anaerobic regimes.

Table 14 summarises the best results obtained for each architecture after hyperparameter optimisation, with the best-performing model selected for the subsequent experiments. As with the GSE3406 dataset, the training process for this dataset showed stable convergence,

with consistent improvements in accuracy and loss reduction. Despite minor fluctuations between runs, the repeated training remained statistically consistent across all repetitions.

Table 14 - Best performance of different models on the full GSE1723 dataset.

Model	Accuracy (mean ± std)	F1-score (mean ± std)
CNN	0.8730 ± 0.0036	0.8730 ± 0.0037
LSTM	0.8218 ± 0.0852	0.8207 ± 0.0889
SVM	0.8402 ± 0.0000	0.8402 ± 0.0000
XGBoost	0.8655 ± 0.0018	0.8655 ± 0.0018

The CNN architecture recorded the best overall results, with an accuracy of 87.30% and an F1-score of 87.30%. XGBoost followed closely (86.55%), showing itself as a strong competitor in this dataset and reinforcing its adequacy for complex, high-dimensional profiles. The SVM achieved slightly lower values, while the LSTM was less consistent, with broader fluctuations across runs and a higher error rate. Taken together, these outcomes suggest that CNN is the most reliable option for modelling GSE1723.

5.3.2. Individual Nutrients

Table 15 reports the classification results obtained when each nutrient limitation was considered independently.

Table 15 - Performance of CNN models trained on individual nutrient limitations (GSE1723).

Nutrient	Accuracy (mean ± std)	F1-score (mean ± std)
Carbon (C)	0.7507 ± 0.0021	0.7506 ± 0.0021
Nitrogen (N)	0.6288 ± 0.0035	0.6207 ± 0.0039
Phosphorus (P)	0.7036 ± 0.0042	0.7030 ± 0.0043
Sulphur (S)	0.6315 ± 0.0033	0.6303 ± 0.0029

Among the four nutrients, carbon produced the most consistent signal, with results approaching 75% accuracy. Phosphorus also contributed moderately, while nitrogen and sulphur were clearly less informative, both stabilising near 63%. The overall picture suggests that, on their own, none of the nutrient environments provided enough discriminative strength to robustly separate aerobic from anaerobic states.

5.3.3. Pairwise Combinations

The next step considered pairs of nutrient limitations. Table 16 shows the performance of CNN models trained on all possible two-nutrient subsets.

Table 16 – Performance of CNN models trained on pairwise nutrient limitations (GSE1723).

Nutrients	Accuracy (mean ± std)	F1-score (mean ± std)
C + N	0.7863 ± 0.0028	0.7862 ± 0.0028
C + P	0.8366 ± 0.0040	0.8366 ± 0.0041
C + S	0.7944 ± 0.0024	0.7944 ± 0.0024
N + P	0.7579 ± 0.0037	0.7578 ± 0.0038
N + S	0.7080 ± 0.0029	0.7077 ± 0.0030
P + S	0.7579 ± 0.0032	0.7578 ± 0.0032

Results highlight that combinations involving carbon achieved better scores, with *carbon + phosphorus* standing out at 83.66% accuracy. By contrast, subsets dominated by nitrogen or sulphur, especially *nitrogen + sulphur* (70.80%), performed more modestly. In general, considering two nutrients improved classification compared with single conditions, but the gain depended strongly on the specific pairing.

5.3.4. Triple Combinations

To further explore the contribution of nutrient environments, CNN models were trained on all possible triplets. The results are presented in Table 17.

Table 17 - Performance of CNN models trained on triple nutrient limitations (GSE1723).

Nutrients	Accuracy (mean ± std)	F1-score (mean ± std)
C + N + P	0.8472 ± 0.0038	0.8471 ± 0.0039
C+ N + S	0.8181 ± 0.0027	0.8180 ± 0.0028
C + P + S	0.8603 ± 0.0033	0.8602 ± 0.0033
N + P + S	0.8023 ± 0.0033	0.8023 ± 0.0033

When three nutrients were combined, classification accuracy improved compared with pairwise conditions. The best results were obtained for *carbon + phosphorus + sulphur*

(86.03% accuracy), followed closely by *carbon* + *nitrogen* + *phosphorus* (84.72%). These two subsets reached values close to those observed with the complete dataset.

Combinations without carbon achieved weaker results, particularly *nitrogen* + *phosphorus* + *sulphur*, which stabilised near 80% accuracy. This reinforces the idea that not all nutrient environments contribute equally to oxygen discrimination, and that specific subsets are more informative than others.

5.3.5. Integrated Discussion of GSE1723 Results

The experiments conducted with the GSE1723 dataset reveal several consistent trends regarding the ability of nutrient environments to support the discrimination between aerobic and anaerobic regimes. When all four nutrients were considered simultaneously, CNN models reached the highest accuracy (87.30%), setting a baseline for subsequent reductions.

Individually, no nutrient proved sufficient to reach high performance, with carbon being the most informative (75.07% accuracy) and nitrogen and sulphur clearly weaker (~63%). This pattern is consistent with transcriptome studies showing that carbon limitation induces broad reprogramming of metabolic and energy-related genes, whereas nitrogen and sulphur limitations are associated with more restricted transcriptional responses (Wu et al., 2004). Pairwise analyses confirmed this imbalance: subsets including carbon consistently produced higher scores, while *nitrogen* + *sulphur* emerged as the least effective combination.

Introducing a third nutrient improved classification in most cases, with *carbon* + *phosphorus* + *sulphur* (86.03%) and *carbon* + *nitrogen* + *phosphorus* (84.72%) standing out. These results approached the performance of the full dataset while excluding one nutrient, indicating that nearly optimal discrimination can be achieved without considering all conditions. The combination of *nitrogen* + *phosphorus* + *sulphur* lagged, again pointing to the limited contribution of nitrogen and sulphur when carbon is absent.

To illustrate classification performance beyond summary metrics, confusion matrices were generated for each experimental setup. To avoid excessive redundancy, only two representative cases are shown here: the CNN model trained on the complete dataset (Figure 11), and the CNN model trained on the best reduced subset, *carbon* + *phosphorus* + *sulphur* (Figure 12).

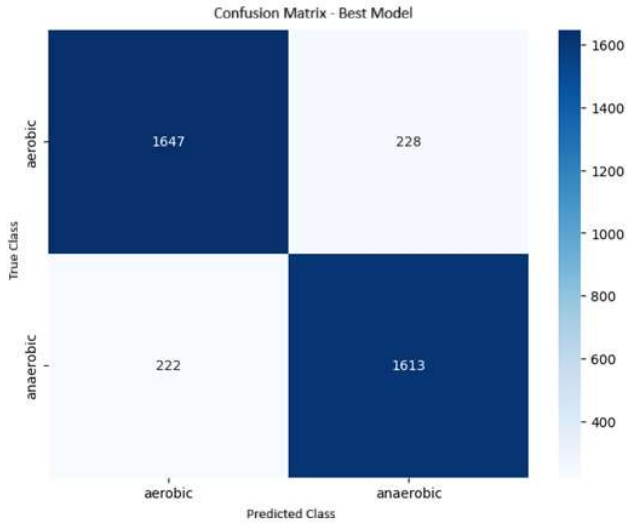

Figure 11 - Confusion matrix for the CNN model trained on the complete GSE1723 dataset.

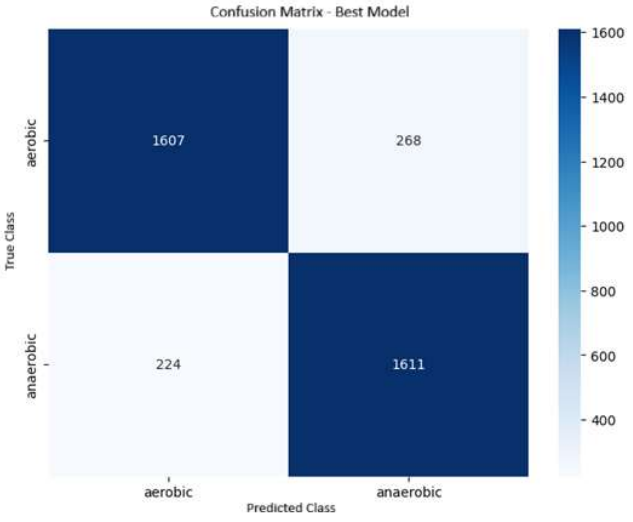

Figure 12 - Confusion matrix for the CNN model trained on the best reduced subset (C + P + S).

Overall, the integrated analysis highlights carbon as the key factor for effective oxygen classification in this dataset, with phosphorus providing additional complementary signal, and weaker results for nitrogen and sulphur. These findings suggest that a strategically reduced experimental design — prioritising carbon and phosphorus environments — could capture most of the discriminative information while reducing the number of conditions required.

5.4. Results with the GSE6186 Dataset

The GSE6186 dataset, which captures gene expression during *Drosophila melanogaster* embryogenesis, was evaluated to classify genes into maternal, transient, and activated categories. Results are presented for the complete dataset with all 28 time points, as well as for reduced input strategies based on truncated or alternating subsets of the series. These experiments evaluate how temporal compression affects classification performance and the feasibility of reducing measurement requirements, thereby testing whether experimental simplification can be achieved without compromising predictive accuracy.

5.4.1. Complete Time Series (28 Points)

In the first stage, all 28 time points were used to establish a baseline performance across the four architectures. The results obtained after hyperparameter optimisation are presented in Table 18

Table 18 - Best performance of different models on the full GSE6186 dataset.

Model	Accuracy (mean ± std)	F1-score (mean ± std)
CNN	0.9399 ± 0.0040	0.9396 ± 0.0040
LSTM	0.9461 ± 0.0053	0.9460 ± 0.0051
SVM	0.9380 ± 0.0000	0.9384 ± 0.0000
XGBoost	0.9302 ± 0.0025	0.9296 ± 0.0026

Among the tested architectures, the LSTM achieved the highest overall performance, slightly outperforming CNN, with both models recording accuracies above 94%. The SVM model followed closely, reaching competitive values despite its simpler nature. XGBoost, while slightly lower, still achieved a stable performance above 93% accuracy. As with the other two datasets, the training showed stable convergence, and the mean performance across repetitions behaved consistently with the trends observed previously.

Given the temporal structure of the dataset and the natural suitability of LSTM to capture long-term dependencies in sequential data, this model was selected as the reference for the reduced input experiments. Nonetheless, CNN-based experiments were also conducted in parallel to provide a comparative baseline, given the close similarity in performance.

5.4.2. Reduced Input Strategies

To evaluate the impact of temporal compression on classification performance, both LSTM and CNN models were trained with truncated and subsampled versions of the time series. The results are reported separately in Table 19 and Table 20, which summarise the performance of each architecture across all configurations.

Table 19 - LSTM results on the GSE6186 dataset with reduced temporal strategies.

Strategy	Accuracy (mean ± std)	F1-score (mean ± std)
Half	0.7925 ± 0.0112	0.7855 ± 0.0116
Third	0.7682 ± 0.0045	0.7573 ± 0.0055
Quarter	0.7446 ± 0.0055	0.7345 ± 0.0080
Alternating (half)	0.7771 ± 0.0073	0.7709 ± 0.0072
Alternating (full)	0.9201 ± 0.0067	0.9198 ± 0.0062

Table 20 - CNN results on the GSE6186 dataset with reduced temporal strategies.

Strategy	Accuracy (mean ± std)	F1-score (mean ± std)
Half	0.7931 ± 0.0085	0.7898 ± 0.0080
Third	0.7607 ± 0.0132	0.7528 ± 0.0103
Quarter	0.7444 ± 0.0052	0.7403 ± 0.0056
Alternating (half)	0.7790 ± 0.0082	0.7756 ± 0.0082
Alternating (full)	0.9297 ± 0.0046	0.9292 ± 0.0045

Overall, truncated strategies produced moderate performance, with accuracy ranging from approximately ~74% to ~79% for both models. Alternating points restricted to the first half of the series achieved comparable results. In contrast, alternating time points across the full sequence delivered substantially higher scores, with accuracy exceeding 92% for LSTM and for CNN, approaching the levels observed with the complete dataset.

5.4.3. Integrated Discussion of GSE6186 Results

When using the complete 28-point series, the LSTM model achieved the best overall results, marginally outperforming CNN and confirming the suitability of recurrent architectures for time-dependent data. Both models, however, recorded accuracies above ~94%, indicating

that the full dataset provides a strong temporal signal for discriminating between gene classes.

In other hand, truncated strategies based on the first quarter, third, or half of the series yielded considerably weaker results, with accuracies stabilising around ~74% – ~79% for both CNN and LSTM. This suggests that early time points alone are insufficient to capture the full dynamics of embryogenesis, where class-specific transitions often occur later in development. Indeed, Hooper et al. (2007) reported that approximately 38% of Class II (transient) genes fall into three dominant temporal windows (2.5–12 h, 11–20 h, and 15–20 h), highlighting the importance of sampling across the full developmental period. The alternating strategy restricted to the first half of the sequence also failed to improve, remaining within the same performance range.

A notable shift emerged when alternating time points were distributed across the entire sequence. In this configuration, CNN surpassed LSTM, reaching 92.97% accuracy compared with 92.01% for LSTM. This inversion indicates that CNN benefits more from sparse but globally distributed temporal information, whereas LSTM relies on continuous sequences to fully exploit its memory capacity. Importantly, this reduced-input strategy produced results comparable to the complete dataset, demonstrating that a shorter but evenly distributed measurement schedule can preserve predictive accuracy.

To visualise the differences between the alternating configurations, Figure 13 and Figure 14 present the training and validation curves for the models trained with alternating points up to half of the sequence and across the full sequence, respectively. In the half-sequence configuration, validation accuracy plateaued around ~77% – ~78% and validation loss remained high, confirming the limited predictive value of this strategy. In contrast, the alternating full-sequence configuration reached validation accuracies above 92%, close to the results obtained with the complete dataset. Nevertheless, the validation loss revealed fluctuations and a late upward trend, suggesting some overfitting despite the overall strong predictive performance.

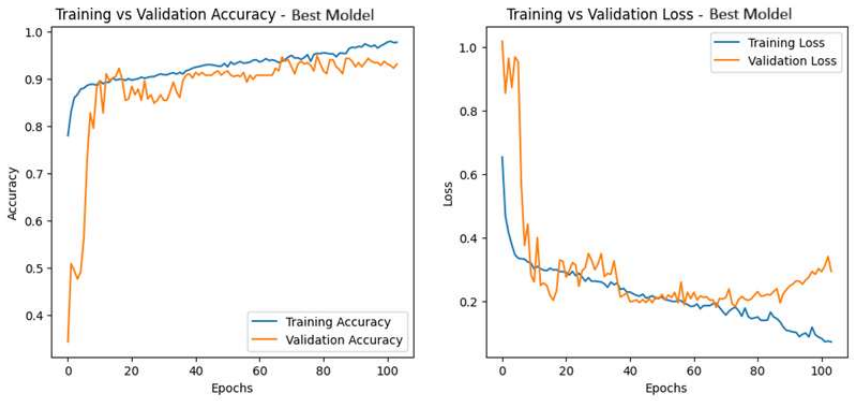

Figure 13 - Training and validation curves for the alternating full-sequence configuration.

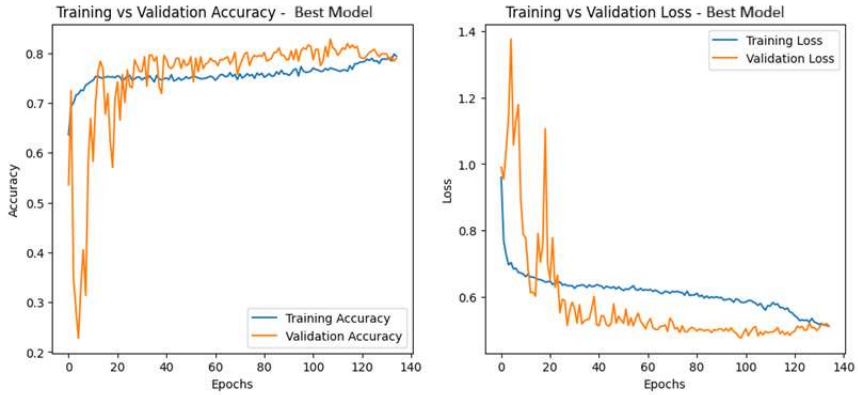

Figure 14 - Training and validation curves for the alternating half-sequence configuration.

5.5. Comparative Analysis with Previous Studies

To contextualise the results obtained in this work, a direct comparison was made with the study of Tripto et al. (2020), who also applied machine learning and deep learning models to the GSE3406, GSE1723, and GSE6186 datasets. Table 21 summarises the best-performing models reported in that study, in terms of accuracy, alongside the corresponding results achieved here using the complete datasets. It is important to note that, the results reported by Tripto et al. do not specify the number of repetitions performed. The values presented in this dissertation are the mean accuracies, providing a more robust estimate of model performance.

Table 21 - Comparative results between this study and Tripto et al. (2020).

Dataset	CNN (TP)	CNN (TS)	LSTM (TP)	LSTM (TS)	SVM (TP)	SVM (TS)
GSE3406	93.14%	95.10%	86.38%	93.07%	88.83%	89.76%
GSE1723	80.25%	87.30%	76.15%	85.72%	83.11%	84.02%
GSE6186	96.15%	93.99%	92.19%	94.61%	95.75%	93.80%

TP = Tripto; TS = This study.

The comparative results reveal distinct trends across the three datasets.

For GSE3406, the results of this study surpassed those reported by Tripto et al. across all architectures. CNN achieved 95.10% compared with 93.14%, LSTM reached 93.07% versus 86.38%, and SVM achieved 89.76% versus 88.83%. These gains suggest that the optimised training procedure and validation strategy may have contributed to improved generalisation.

For GSE1723, the present results also outperformed those of Tripto et al. across all architectures. CNN, LSTM, and SVM all achieved higher performance in this study, further confirming the effectiveness of the proposed methodology and showing that the improvements are consistent across datasets with different stress conditions.

For GSE6186, Tripto et al. reported the highest accuracy with CNN (96.15%), whereas in the present work LSTM slightly outperformed CNN (94.61% vs 93.99%). This inversion suggests that recurrent architectures may provide advantages in capturing sequential dependencies within this dataset, although both models achieved similar results overall. Although the absolute values are slightly lower than those of Tripto, the results indicate that both architectures perform competitively when the full temporal resolution is preserved.

Overall, the comparison indicates that, while Tripto et al. identified CNN as the top-performing model across datasets, our results show that LSTM can surpass CNN in GSE6186 and that substantial improvements can be achieved for GSE3406 and GSE1723 across all architectures. Unlike Tripto et al., whose study does not specify the number of repetitions performed or provide detailed information on hyperparameter tuning, this dissertation incorporated repeated training runs, systematic hyperparameter optimisation, and additional regularisation, with architectures slightly adapted to the specific characteristics of each dataset. As a result, the models, although not deeper in terms of layers, are potentially more stable and consistently reliable.

This page was intentionally left blank

6. Conclusions

This dissertation demonstrated that both experimental and computational complexity in gene expression classification can be simplified without substantially compromising predictive performance. Systematic comparison of models and evaluation of data simplification strategies showed that deep learning approaches, particularly CNNs and LSTMs, retain strong predictive capacity even under limited input conditions.

In the GSE3406 dataset, CNNs proved especially effective for discriminating between *Saccharomyces* species under stress conditions, with nitrogen starvation emerging as the most informative condition and heat shock contributing the least to classification accuracy. Gene-level consistency analysis further revealed subsets of genes that were consistently accurately or poorly classified, providing evidence for future feature selection and biological interpretation.

In the GSE1723 dataset, performance depended strongly on nutrient environment, with carbon-based conditions providing the strongest discriminative signal for oxygen availability. Combinations involving carbon and phosphorus nearly matched the full dataset, showing that well-chosen subsets can deliver strong classification with fewer requirements.

In GSE6186, full temporal resolution favoured LSTM models, yet alternating time-point strategies showed that near-optimal results can be achieved with fewer measurements, offering a compromise between accuracy and experimental cost.

Overall, the research objectives were achieved:

- It was shown that reduced sets of stimuli, nutrient conditions, or time points can sustain high levels of predictive performance.
- Gene-level consistency analysis highlighted features with stable discriminative power, reinforcing their potential for feature selection.
- The complementary strengths of CNNs and LSTMs were demonstrated, emphasising that model choice should be aligned with dataset structure and the degree of simplification required.
- The findings were compared with results reported in previous studies, showing consistent trends and reinforcing the validity of the proposed approach.

Together, these results support the development of more efficient and cost-effective strategies for gene expression classification.

The contributions presented here also suggest directions for future research. Future studies could apply the proposed approaches to additional gene expression datasets to assess their generalisability, while also incorporating interpretability methods such as SHAP to better understand the contribution of individual features. Exploring hybrid architectures such as CNN–LSTM or attention-based models may further enhance predictive capacity and open new perspectives for future studies. It will also be important to evaluate the real costs of complete versus reduced experimental designs, thereby confirming or refuting its advantages. Finally, the consistently well-classified genes identified in this work could be validated as biomarkers or used to guide feature selection in future models.

Finally, it is worth highlighting that part of the work presented in this dissertation was accepted for oral presentation and peer-reviewed publication at the International Conference on Health and Social Care Information Systems and Technologies (HCIST 2025), further validating its scientific relevance.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., … Zheng, X. (2016). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems* (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1603.04467

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631. https://doi.org/10.1145/3292500.3330701

Alberts, B., Johnson, A., Lewis, J., Morgan, D., Raff, M., Roberts, K., & Walter, P. (2014). *Molecular biology of the cell* (6th ed). Garland science, Taylor and Francis group.

Alharbi, F., & Vakanski, A. (2023). Machine Learning Methods for Cancer Classification Using Gene Expression Data: A Review. *Bioengineering*, *10*(2), 173. https://doi.org/10.3390/bioengineering10020173

Aliouane, S. E., Chehili, H., Boulahrouf, K., Abdelaziz, A., Khlifa, N., & Hamidechi, M. A. (2025). Integrating Deep Learning and SHAP for Breast Cancer Classification and Biomarker Discovery Using Gene Expression Data. *IEEE Access*, *13*, 49693–49709. https://doi.org/10.1109/ACCESS.2025.3552280

Allison, D. B., Cui, X., Page, G. P., & Sabripour, M. (2006). Microarray data analysis: From disarray to consolidation and consensus. *Nature Reviews Genetics*, *7*(1), 55–65. https://doi.org/10.1038/nrg1749

Asyali, M., Colak, D., Demirkaya, O., & Inan, M. (2006). Gene Expression Profile Classification: A Review. *Current Bioinformatics*, *1*(1), 55–73. https://doi.org/10.2174/157489306775330615

Bar-Joseph, Z. (2004). Analyzing time series gene expression data. *Bioinformatics*, *20*(16), 2493–2503. https://doi.org/10.1093/bioinformatics/bth283

Bar-Joseph, Z., Gitter, A., & Simon, I. (2012). Studying and modelling dynamic biological processes using time-series gene expression data. *Nature Reviews Genetics*, *13*(8), 552–564. https://doi.org/10.1038/nrg3244

Bouthillier, X., Delaunay, P., Bronzi, M., Trofimov, A., Nichyporuk, B., Szeto, J., Sepah, N., Raff, E., Madan, K., Voleti, V., Kahou, S. E., Michalski, V., Serdyuk, D., Arbel, T., Pal, C., Varoquaux, G., & Vincent, P. (2021). *Accounting for Variance in Machine Learning Benchmarks* (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2103.03098

Casella, G., & Berger, R. L. (2002). *Statistical inference* (2nd edn). Duxbury.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Chollet, F. (2015). *Keras* [Computer software].

Costello, J. C., Dalkilic, M. M., Beason, S. M., Gehlhausen, J. R., Patwardhan, R., Middha, S., Eads, B. D., & Andrews, J. R. (2009). Gene networks in *Drosophila melanogaster*: Integrating experimental data to predict gene function. *Genome Biology*, *10*(9). https://doi.org/10.1186/gb-2009-10-9-r97

Cotton, T. B., Nguyen, H. H., Said, J. I., Ouyang, Z., Zhang, J., & Song, M. (2015). Discerning mechanistically rewired biological pathways by cumulative interaction heterogeneity statistics. *Scientific Reports*, *5*(1). https://doi.org/10.1038/srep09634

Deng, X., Li, M., Deng, S., & Wang, L. (2022). Hybrid gene selection approach using XGBoost and multi-objective genetic algorithm for cancer classification. *Medical & Biological Engineering & Computing*, *60*(3), 663–681. https://doi.org/10.1007/s11517-021-02476-x

Dougherty, E. R. (2001). Small sample issues for microarray-based classification. *Comparative and Functional Genomics*, *2*(1), 28–34. https://doi.org/10.1002/cfg.62

Fakoor, R., Ladhak, F., Nazi, A., & Huber, M. (2013). *Using deep learning to enhance cancer diagnosis and classification*. The 30th International Conference on Machine Learning (ICML 2013),WHEALTH workshop.

Fan, Y., Xiong, H., & Sun, G. (2023). DeepASDPred: A CNN-LSTM-based deep learning method for Autism spectrum disorders risk RNA identification. *BMC Bioinformatics*, *24*(1). https://doi.org/10.1186/s12859-023-05378-x

Fay, J. C., Alonso-del-Real, J., Miller, J. H., & Querol, A. (2023). Divergence in the *Saccharomyces* Species' Heat Shock Response Is Indicative of Their Thermal Tolerance. *Genome Biology and Evolution*, *15*(11), evad207. https://doi.org/10.1093/gbe/evad207

GPL2910. (n.d.). Retrieved 19 September 2025, from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GPL2910

GPL4455. (n.d.). Retrieved 15 July 2025, from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GPL4455

GSE1723. (2005). https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE1723

GSE3406. (2006). https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE3406

GSE6186. (2007). https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE6186

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, *46*(1/3), 389–422. https://doi.org/10.1023/A:1012487302797

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer New York. https://doi.org/10.1007/978-0-387-84858-7

Hira, Z. M., & Gillies, D. F. (2015). A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data. *Advances in Bioinformatics*, *2015*, 1–13. https://doi.org/10.1155/2015/198363

Hooper, S. D., Boué, S., Krause, R., Jensen, L. J., Mason, C. E., Ghanim, M., White, K. P., Furlong, E. E., & Bork, P. (2007). Identification of tightly regulated groups of genes during *Drosophila melanogaster* embryogenesis. *Molecular Systems Biology*, *3*(1). https://doi.org/10.1038/msb4100112

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

IBM. (2025). *CRISP-DM Help Overview—IBM Documentation*. IBM Documentation. https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview&mhsrc=ibmsearch_a&mhq=crisp-dm

Jiang, S., & Hassanpour, S. (2025). *Transformer-Based Representation Learning for Robust Gene Expression Modeling and Cancer Prognosis* (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2504.09704

Jordan, K. (2023). *On the Variance of Neural Network Training with respect to Test Sets and Distributions* (Version 4). arXiv. https://doi.org/10.48550/ARXIV.2304.01910

Karim, F., Majumdar, S., Darabi, H., & Chen, S. (2018). LSTM Fully Convolutional Networks for Time Series Classification. *IEEE Access*, *6*, 1662–1669. https://doi.org/10.1109/access.2017.2779939

Kiselev, V. Y., Yiu, A., & Hemberg, M. (2018). scmap: Projection of single-cell RNA-seq data across data sets. *Nature Methods*, *15*(5), 359–362. https://doi.org/10.1038/nmeth.4644

Knijnenburg, T. A., De Winde, J. H., Daran, J.-M., Daran-Lapujade, P., Pronk, J. T., Reinders, M. J., & Wessels, L. F. (2007). Exploiting combinatorial cultivation conditions to infer transcriptional regulation. *BMC Genomics*, *8*(1). https://doi.org/10.1186/1471-2164-8-25

Kohavi, R. (1995). *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection*. *2*, 1137–1143. https://www.ijcai.org/Proceedings/95-2/Papers/016.pdf

Laurent, C., Pereyra, G., Brakel, P., Zhang, Y., & Bengio, Y. (2015). *Batch Normalized Recurrent Neural Networks* (No. arXiv:1510.01378). arXiv. https://doi.org/10.48550/arXiv.1510.01378

Montesinos-López, O. A., Montesinos-López, A., Pérez-Rodríguez, P., Barrón-López, J. A., Martini, J. W. R., Fajardo-Flores, S. B., Gaytan-Lugo, L. S., Santana-Mancilla, P. C., & Crossa, J. (2021). A review of deep learning applications for genomic selection. *BMC Genomics*, *22*(1), 19. https://doi.org/10.1186/s12864-020-07319-x

Mostavi, M., Chiu, Y.-C., Huang, Y., & Chen, Y. (2020). Convolutional neural network models for cancer type prediction based on gene expression. *BMC Medical Genomics*, *13*(S5). https://doi.org/10.1186/s12920-020-0677-2

Ozsolak, F., & Milos, P. M. (2011). RNA sequencing: Advances, challenges and opportunities. *Nature Reviews Genetics*, *12*(2), 87–98. https://doi.org/10.1038/nrg2934

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Müller, A., Nothman, J., Louppe, G., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2012). *Scikit-learn: Machine Learning in Python*. https://doi.org/10.48550/ARXIV.1201.0490

Platt, J. C. (1999). Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers* (Alexander J.

Smola, Peter Bartlett, Bernhard Schölkopf, Dale Schuurmans, Vol. 10, pp. 61–74). MIT Press.

Python Software Foundation. (2024). *Python: A programming language* (Version 3.12.5) [Computer software]. https://www.python.org/

Quackenbush, J. (2001). Computational analysis of microarray data. *Nature Reviews Genetics*, *2*(6), 418–427. https://doi.org/10.1038/35076576

Santurkar, S., Tsipras, D., Ilyas, A., & Madry, A. (2019). *How Does Batch Normalization Help Optimization?* (No. arXiv:1805.11604). arXiv. https://doi.org/10.48550/arXiv.1805.11604

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., & Futcher, B. (1998). Comprehensive Identification of Cell Cycle–regulated Genes of the Yeast *Saccharomyces cerevisiae* by Microarray Hybridization. *Molecular Biology of the Cell*, *9*(12), 3273–3297. https://doi.org/10.1091/mbc.9.12.3273

Spolaôr, N., Lee, H. D., Mendes, A. I., Nogueira, C. V., Parmezan, A. R. S., Takaki, W. S. R., Coy, C. S. R., Wu, F. C., & Fonseca-Pinto, R. (2023). Fine-tuning pre-trained neural networks for medical image classification in small clinical datasets. *Multimedia Tools and Applications*, *83*(9), 27305–27329. https://doi.org/10.1007/s11042-023-16529-w

Swamy, K. B. S., Lin, C.-H., Yen, M.-R., Wang, C.-Y., & Wang, D. (2014). Examining the condition-specific antisense transcription in *S. cerevisiae* and *S. paradoxus*. *BMC Genomics*, *15*(1), 521. https://doi.org/10.1186/1471-2164-15-521

Tabassum, N., Kamal, M. A. S., Akhand, M. A. H., & Yamada, K. (2024). Cancer Classification from Gene Expression Using Ensemble Learning with an Influential Feature Selection Technique. *BioMedInformatics*, *4*(2), 1275–1288. https://doi.org/10.3390/biomedinformatics4020070

Tai, S. L., Boer, V. M., Daran-Lapujade, P., Walsh, M. C., De Winde, J. H., Daran, J.-M., & Pronk, J. T. (2005). Two-dimensional Transcriptome Analysis in Chemostat Cultures. *Journal of Biological Chemistry*, *280*(1), 437–447. https://doi.org/10.1074/jbc.M410573200

Tirosh, I., Weinberger, A., Carmi, M., & Barkai, N. (2006). A genetic signature of interspecies variations in gene expression. *Nature Genetics*, *38*(7), 830–834. https://doi.org/10.1038/ng1819

Tirosh, I., Wong, K. H., Barkai, N., & Struhl, K. (2011). Extensive divergence of yeast stress responses through transitions between induced and constitutive activation. *Proceedings of the National Academy of Sciences*, *108*(40), 16693–16698. https://doi.org/10.1073/pnas.1113718108

Tripto, N. I., Kabir, M., Bayzid, Md. S., & Rahman, A. (2020). Evaluation of classification and forecasting methods on time series gene expression data. *PLOS ONE*, *15*(11), e0241686. https://doi.org/10.1371/journal.pone.0241686

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., & Altman, R. B. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics*, *17*(6), 520–525. https://doi.org/10.1093/bioinformatics/17.6.520

Wang, Z., Gerstein, M., & Snyder, M. (2009). RNA-Seq: A revolutionary tool for transcriptomics. *Nature Reviews Genetics*, *10*(1), 57–63. https://doi.org/10.1038/nrg2484

Waskom, M. (2021). seaborn: Statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Wasserman, L. (2004). *All of Statistics: A Concise Course in Statistical Inference*. Springer New York. https://doi.org/10.1007/978-0-387-21736-9

Wu, J., Zhang, N., Hayes, A., Panoutsopoulou, K., & Oliver, S. G. (2004). Global analysis of nutrient control of gene expression in *Saccharomyces cerevisiae* during growth and

starvation. *Proceedings of the National Academy of Sciences*, *101*(9), 3148–3153. https://doi.org/10.1073/pnas.0308321100

Yap, M., Johnston, R. L., Foley, H., MacDonald, S., Kondrashova, O., Tran, K. A., Nones, K., Koufariotis, L. T., Bean, C., Pearson, J. V., Trzaskowski, M., & Waddell, N. (2021). Verifying explainability of a deep learning tissue classifier trained on RNA-seq data. *Scientific Reports*, *11*(1). https://doi.org/10.1038/s41598-021-81773-9

Yoneya, T., & Mamitsuka, H. (2007). A hidden Markov model-based approach for identifying timing differences in gene expression under different experimental factors. *Bioinformatics*, *23*(7), 842–849. https://doi.org/10.1093/bioinformatics/btl667

Yu, H., Samuels, D. C., Zhao, Y., & Guo, Y. (2019). Architectures and accuracy of artificial neural network for disease classification from omics data. *BMC Genomics*, *20*(1). https://doi.org/10.1186/s12864-019-5546-z

Zeng, H., Edwards, M. D., Liu, G., & Gifford, D. K. (2016). Convolutional neural network architectures for predicting DNA–protein binding. *Bioinformatics*, *32*(12), i121–i127. https://doi.org/10.1093/bioinformatics/btw255

Zhang, T.-H., Hasib, M. M., Chiu, Y.-C., Han, Z.-F., Jin, Y.-F., Flores, M., Chen, Y., & Huang, Y. (2022). Transformer for Gene Expression Modeling (T-GEM): An Interpretable Deep Learning Model for Gene Expression-Based Phenotype Predictions. *Cancers*, *14*(19), 4763. https://doi.org/10.3390/cancers14194763

Zhao, S., Zhao, X., Zou, H., Fu, J., Du, G., Zhou, J., & Chen, J. (2014). Comparative proteomic analysis of *Saccharomyces cerevisiae* under different nitrogen sources. *Journal of Proteomics*, *101*, 102–112. https://doi.org/10.1016/j.jprot.2014.01.031

Appendix A - Hyperparameter Optimisation Results

This appendix presents all the hyperparameters obtained through the optimisation process carried out with the Optuna library. It includes the complete results for all datasets considered and for all tested data combinations.

GSE3406 — CNN Hyperparameters Optimisation Results (Hold-Out)

Stimuli	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
All stimuli	0.0001231719	8.70E-06	0.2150	256
H^2O^2	0.0002943151	6.66E-06	0.1979	256
Heat Shock (HS)	0.0005067572	1.64E-06	0.2700	128
Nitrogen Starvation (NS)	0.0002751649	7.40E-06	0.2548	256
Transferred from Glucose (TfG)	0.0004225770	2.53E-06	0.2566	128
H^2O^2+HS	0.0008834366	1.65E-06	0.2697	256
H^2O^2+MMS	0.0003312434	1.30E-06	0.2781	64
H^2O^2+NS	0.0001479935	1.80E-06	0.2577	128
H^2O^2+TfG	0.0001179644	8.01E-06	0.1979	256
HS+MMS	0.0003583409	2.28E-06	0.2845	256
HS+NS	0.0000629132	3.08E-06	0.2694	64
HS+TfG	0.0000935891	3.66E-06	0.2399	128
MMS+NS	0.0001060901	1.04E-06	0.2167	256
MMS+TfG	0.0004414568	2.64E-06	0.2897	256
NS+TfG	0.0001155506	2.93E-06	0.2492	128
H^2O^2+HS+MMS	0.0002544451	3.20E-06	0.2623	256
H^2O^2+HS+NS	0.0000709417	4.15E-06	0.2662	256
H^2O^2+HS+TfG	0.0002159013	2.39E-06	0.2459	256
H^2O^2+MMS+NS	0.0001263755	5.04E-06	0.2786	256
H^2O^2+MMS+TfG	0.0002758583	1.86E-06	0.2635	64
H^2O^2+NS+TfG	0.0001333365	4.02E-06	0.2798	64
HS+MMS+NS	0.0003694783	1.14E-05	0.2894	256

Stimuli	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
HS+MMS+TfG	0.0001647915	3.76E-06	0.1873	256
HS+NS+TfG	0.0000899202	4.53E-06	0.2678	256
MMS+NS+TfG	0.0000355608	6.97E-06	0.2177	256
H^2O^2+HS+MMS+NS	0.0000246987	3.29E-06	0.2641	256
H^2O^2+HS+MMS+TfG	0.0000263920	9.56E-06	0.2721	128
H^2O^2+HS+NS+TfG	0.0000564465	3.43E-06	0.2890	256
H^2O^2+MMS+NS+TfG	0.0001445505	1.50E-06	0.2766	256
HS+MMS+NS+TfG	0.0000684094	1.53E-05	0.2904	128

GSE3406 — LSTM Hyperparameters Optimisation Results (Hold-Out)

Stimuli	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
All stimuli (Flat)	0.0070299507	3.24E-06	0.3924	256
All stimuli (T6-Interpolated)	0.0005772839	2.15E-06	0.2874	512
All stimuli (Intersect)	0.0004850366	1.67E-06	0.3377	256
All stimuli (Index-Based)	0.0001667435	2.23E-06	0.3639	256

GSE3406 — SVM Hyperparameters Optimisation Results (Hold-Out)

Stimuli	*C*	*kernel*	*gamma*
All stimuli	86.7239	rbf	0.0155

GSE3406 — XGBoost Hyperparameters Optimisation Results (Hold-Out)

Stimuli	*max_depth*	*learning_rate*	*n_estimators*	*subsample*	*colsample_bytree*	*reg_alpha*	*reg_lambda*	*gamma*
All stimuli	5	0.3056	490	0.8847	0.7089	0.6261	0.9471	0.00920

GSE3406 — CNN Hyperparameters Optimisation Results (K-Folds)

Stimuli	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
All stimuli	0.0000640895	2.34E-06	0.2355	256
H^2O^2	0.0003106605	1.27E-06	0.2149	256
Heat Shock	0.0007623637	1.30E-06	0.2673	256
MMS	0.0002763171	2.29E-06	0.2145	256
Nitrogen Starvation	0.0002935296	1.35E-06	0.2805	256
Transferred from Glucose	0.0002468275	1.12E-06	0.2077	128

GSE1723 — CNN Hyperparameters Optimisation Results (Hold-Out)

Nutrients	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
All nutrients	0.0001438303	3.03E-06	0.2429	64
Carbon (C)	0.0001397480	2.44E-06	0.1601	128
Nitrogen (N)	0.0002983408	1.45E-06	0.2497	128
Phosphorus	0.0002526022	1.05E-06	0.1557	256
Sulphur (S)	0.0002801640	5.55E-06	0.2636	128
C+N	0.0001274503	8.58E-06	0.2393	128
C+P	0.0001587449	3.43E-06	0.1598	256
C+S	0.0002062448	1.38E-06	0.1620	256
N+P	0.0001159805	1.59E-06	0.1696	128
N+S	0.0002435140	1.63E-06	0.2841	64
P+S	0.0001534802	1.84E-06	0.2180	256
C+N+P	0.0001072539	2.98E-06	0.2116	64
C+N+S	0.0002376626	1.39E-06	0.2143	256
C+P+S	0.0002568367	1.60E-06	0.2451	256
N+P+S	0.0000617960	8.20E-06	0.1930	128

GSE1723 — LSTM Hyperparameters Optimisation Results (Hold-Out)

Nutrients	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
All Nutrients	0.0011368209	1.51E-05	0.2874	256

GSE1723 — SVM Hyperparameters Optimisation Results (Hold-Out)

Nutrients	*C*	*kernel*	*gamma*
All Nutrients	946.8991	rbf	0.5960

GSE1723 — XGBoost Hyperparameters Optimisation Results (Hold-Out)

Nutrients	*max_depth*	*learning_rate*	*n_estimators*	*subsample*	*colsample_bytree*	*reg_alpha*	*reg_lambda*	*gamma*
All Nutrients	7	0.1783	317	0.6017	0.8670	6.4738	5.3474	0.0182

GSE6186 — CNN Hyperparameters Optimisation Results (Hold-Out)

Time Points	learning_rate	L2_regularisation	dropout	batch_size
All Time Points	4.28E-05	9.86E-08	0.5531	256
Third	5.74E-03	1.98E-06	0.4346	16
Alternating (half)	9.51E-04	3.15E-09	0.5388	16
Alternating (all)	1.03E-03	1.37E-06	0.6296	32
Quarter	7.09E-04	5.76E-07	0.5755	16
Half	1.43E-03	7.78E-07	0.5658	16

GSE6186 — LSTM Hyperparameters Optimisation Results (Hold-Out)

Time Points	*learning_rate*	*L2_regularisation*	*dropout*	*batch_size*
All Time Points	0.002416	3.16E-09	0.7581	64
Third	0.002785	2.46E-06	0.6135	64
Alternating (half)	0.004523	3.28E-06	0.4745	64
Alternating (all)	0.004808	7.22E-05	0.4013	32
Quarter	0.007308	1.14E-09	0.5362	128
Half	0.001846	4.04E-06	0.4777	64

GSE6186 — SVM Hyperparameters Optimisation Results (Hold-Out)

Time Points	*C*	*kernel*
All Time Points	0.0548	linear

GSE6186 — XGBoost Hyperparameters Optimisation Results (Hold-Out)

Time Points	*max_depth*	*learning_rate*	*n_estimators*	*subsample*	*colsample_bytree*	*reg_alpha*	*reg_lambda*	*gamma*
All Time P.	6	0.0829	275	0.6017	0.6276	5.0861	2.6795	0.0033

Appendix B - Execution Times for Optimization and Training

This appendix compiles the execution times recorded during the optimisation and training phases. Detailed results are provided for all models, datasets, and data combinations evaluated. Preliminary test times were not included.

Execution Times for GSE3406 Dataset

Stimuli	Model	Optuna Time	Training Time	Total (hh:mm:ss)
Heat Shock	CNN	04:46:21	03:03:01	07:49:22
H^2O^2	CNN	04:24:21	02:34:49	06:59:10
MMS	CNN	04:29:24	03:00:17	07:29:41
Nitrogen Starvation	CNN	04:42:45	02:26:50	07:09:35
Transferred from Glucose	CNN	04:26:57	03:09:47	07:36:44
H^2O^2+ HS	CNN	05:27:35	02:18:16	07:45:51
H^2O^2 + MMS	CNN	07:47:21	07:07:21	14:54:42
H^2O^2 + NS	CNN	06:00:56	03:59:11	10:00:07
H^2O^2 + TfG	CNN	04:26:09	02:37:05	07:03:14
HS + MMS	CNN	04:10:38	01:57:06	06:07:44
HS + NS	CNN	08:10:57	08:59:44	17:10:41
HS + TfG	CNN	06:05:45	04:51:07	10:56:52
MMS + NS	CNN	05:10:51	02:12:27	07:23:18
MMS + TfG	CNN	04:00:46	04:31:19	09:32:05
NS + TfG	CNN	06:19:43	04:58:07	11:17:50
H^2O^2 + HS + MMS	CNN	05:27:27	02:00:23	07:27:50
H^2O^2+ HS + NS	CNN	04:23:38	02:52:34	07:16:12
H^2O^2+ HS + TfG	CNN	05:39:49	02:02:02	07:41:51
H^2O^2 + MMS + NS	CNN	04:16:36	02:00:01	06:16:37
H^2O^2 + MMS + TfG	CNN	04:39:11	05:11:45	09:50:56
H^2O^2 + NS + TfG	CNN	06:30:31	07:03:05	13:33:36
HS + MMS + NS	CNN	04:25:00	01:20:49	05:45:49

Stimuli	Model	Optuna Time	Training Time	Total (hh:mm:ss)
HS + MMS + TfG	CNN	03:51:30	01:36:20	05:27:50
HS + NS + TfG	CNN	07:28:11	03:07:55	10:36:06
MMS + NS + TfG	CNN	08:40:43	04:42:42	13:23:25
H^2O^2 + HS + MMS + NS	CNN	05:39:01	04:44:57	10:23:58
H^2O^2 + HS + MMS + TfG	CNN	07:04:06	09:12:17	16:16:23
H^2O^2 + HS + NS + TfG	CNN	04:21:37	03:09:08	07:30:45
H^2O^2 + MMS + NS + TfG	CNN	04:06:53	01:38:48	05:45:41
HS + MMS + NS + TfG	CNN	04:13:46	04:50:41	09:04:27
All Stimuli	CNN	04:52:03	01:36:52	06:28:55
All Stimuli	LSTM (Flat)	05:30:45	04:24:24	09:55:09
All Stimuli	LSTM (T6-Int)	08:22:44	06:12:39	14:35:23
All Stimuli	LSTM (Intersect)	05:00:24	02:43:27	07:43:51
All Stimuli	LSTM (Index)	08:41:17	14:11:57	22:53:14
All Stimuli	SVM	03:10:39	01:41:04	04:51:43
All Stimuli	XGBoost	00:04:35	00:09:59	00:14:34
Heat Shock	CNN (K-Fold)	34:11:37	07:34:11	41:45:48
H^2O^2	CNN (K-Fold)	22:42:23	13:46:45	36:29:08
MMS	CNN (K-Fold)	26:37:32	32:49:31	59:27:03
Nitrogen Starvation	CNN (K-Fold)	42:24:09	13:43:53	56:08:02
Transferred from Glucose	CNN (K-Fold)	50:48:34	16:57:24	67:45:58
All Stimuli	CNN (K-Fold)	37:14:26	13:30:48	50:45:14
TOTALS	—	**410:59:36**	**242:42:48**	**631:49:10**

Execution Times for GS1723 Dataset

Nutrients	Model	Optuna Time	Training Time	Total (hh:mm:ss)
Carbon	CNN	05:01:44	02:17:24	07:19:08
Nitrogen	CNN	02:43:20	01:20:06	04:03:26
Phosphorus	CNN	02:33:26	01:14:29	03:47:55
Sulphur	CNN	03:35:16	02:14:21	05:49:37
C+N	CNN	04:26:41	02:36:55	07:03:36

Nutrients	Model	Optuna Time	Training Time	Total (hh:mm:ss)
C+P	CNN	03:22:19	01:31:03	04:53:22
C+S	CNN	03:28:20	01:21:36	04:49:56
N+P	CNN	03:30:19	02:13:11	05:43:30
N+S	CNN	04:19:25	03:18:26	07:37:51
P+S	CNN	03:49:06	01:48:09	05:37:15
C+N+P	CNN	05:00:48	04:11:22	09:12:10
C+N+S	CNN	03:24:30	01:24:17	04:48:47
C+P+S	CNN	03:10:31	01:29:29	04:40:00
N+P+S	CNN	04:45:45	03:20:01	08:05:46
All Nutrients	CNN	05:03:56	03:39:20	08:43:16
All Nutrients	LSTM	09:38:21	02:35:43	12:14:04
All Nutrients	SVM	00:47:07	00:47:28	01:34:35
All Nutrients	XGBoost	00:01:04	00:04:12	00:05:16
TOTALS	**—**	**68:41:58**	**37:27:32**	**106:09:30**

Execution Times for GSE6186 Dataset

Time Points	Model	Optuna Time	Training Time	Total (hh:mm:ss)
All Time Points	CNN	00:56:24	00:27:54	01:24:18
Third	CNN	01:01:19	01:13:23	02:14:42
Alternating (half)	CNN	01:07:37	01:13:30	02:21:07
Alternating (all)	CNN	00:49:03	00:36:30	01:25:33
Quarter	CNN	01:07:22	01:02:51	02:10:13
Half	CNN	01:15:58	01:06:46	02:22:44
All Time Points	LSTM	03:21:45	01:58:15	05:20:00
Third	LSTM	01:42:43	00:57:08	02:39:51
Alternating (half)	LSTM	02:30:42	01:16:42	03:47:24
Alternating (all)	LSTM	02:56:47	01:13:09	04:09:56
Quarter	LSTM	01:43:15	00:36:15	02:19:30
Half	LSTM	02:35:46	01:07:49	03:43:35
All Time Points	SVM	00:01:05	00:00:14	00:01:19

Time Points	Model	Optuna Time	Training Time	Total (hh:mm:ss)
All Time Points	XGBoost	00:01:05	00:01:46	00:02:51
TOTALS	**—**	**21:10:51**	**12:52:12**	**34:03:03**

Total Execution Times across all datasets

Execution Type	Total Time (hh:mm:ss)
Optuna	500:52:25
Training	284:06:26
Optuna + Training	784:58:51