COPPE
Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia
UFRJ

A DATA-DRIVEN APPROACH FOR ESTIMATING TEMPERATURE VARIATIONS BASED ON B-MODE ULTRASOUND IMAGES AND CHANGES IN BACKSCATTERED ENERGY

Luiz Fernando dos Reis de Oliveira

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Priscila Machado Vieira Lima
Wagner Coelho de Albuquerque Pereira
Felipe Maia Galvão França

Rio de Janeiro
Dezembro de 2023

A DATA-DRIVEN APPROACH FOR ESTIMATING TEMPERATURE VARIATIONS BASED ON B-MODE ULTRASOUND IMAGES AND CHANGES IN BACKSCATTERED ENERGY

Luiz Fernando dos Reis de Oliveira

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: Priscila Machado Vieira Lima
Wagner Coelho de Albuquerque Pereira
Felipe Maia Galvão França

Aprovada por: Profa. Priscila Machado Vieira Lima Ph.D.
Prof. Felipe Maia Galvão França Ph.D.
Prof. Wagner Coelho de Albuquerque Pereira D.Sc.
Prof. Claudio Miceli de Farias D.Sc.
Prof. César Alexandre Domingues Teixeira Ph.D.
Prof. Aluizio Fausto Ribeiro Araújo Ph.D.
Prof. Valmir Carneiro Barbosa Ph.D.

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2023

dos Reis de Oliveira, Luiz Fernando

A data-driven approach for estimating temperature variations based on B-mode Ultrasound Images and Changes in Backscattered Energy/Luiz Fernando dos Reis de Oliveira. – Rio de Janeiro: UFRJ/COPPE, 2023.

XXIV, 142 p.: il.; 29, 7cm.

Orientadores: Priscila Machado Vieira Lima
Wagner Coelho de Albuquerque Pereira
Felipe Maia Galvão França

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2023.

Referências Bibliográficas: p. 76 – 83.

1. Ultrasound imaging. 2. Temperature estimation. 3. Machine learning. I. Machado Vieira Lima, Priscila *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Ao vô Jorge.

Agradecimentos

Por todas as inúmeras situações que ocorreram durante esse período, estou certo de que foi uma das experiências mais difíceis de toda a minha vida. Confesso que ameacei desistir diversas vezes. Cheguei na iminência de largar de mão ao menos duas vezes. Mas tive ajuda e apoio de muitas pessoas e, com toda certeza, só escrevo este texto de agradecimento por conta delas.

Na primeira prateleira, à minha mãe e ao meu pai, Edilene e Jorge Luiz, pela dedicação durante toda a minha vida para que eu sempre tivesse tudo o que precisava para atingir qualquer objetivo ao qual me propusesse e por sempre acreditarem em mim de forma irrestrita. E, junto a eles, minha amada esposa Laís. Você suportou muita coisa durante todo esse tempo e me acompanhou nessa jornada caótica, sempre me dando todo o amor e suporte que eu precisava, tentando sempre me colocar pra cima, mesmo nos momentos mais difíceis. Que a partir de agora eu possa compensar e retribuir por todo seu esforço.

À minha família, pelo apoio e crédito durante todos esse anos e por fazerem parte em tantos aspectos do meu crescimento. Em especial aos meus afilhados, João Pedro e Isabela, que tanto me ajudaram a evoluir como pessoa.

Aos grandes amigos do Clube do Vacila (Hugo, Rafael e Vinícius), da Sociedade da RAM (Aluizio, Gabriel e Leopoldo) e do PESC (Alexsander, Bárbara e Christian) por toda a parceria durante esses anos. Cada um a seu modo foi fundamental para que eu chegasse até aqui, sejam nas conversas aleatórias ou discussões mais técnicas.

Meu também muito obrigado aos professores Felipe, Wagner e Priscila, acima de tudo pela paciência, pois sei que não foi fácil. O que começou como uma reunião sem muitas pretensões culminou neste trabalho que abriu diversas oportunidades. Espero que no futuro possamos voltar a colaborar.

Aos meus alunos de TCC da UFRRJ, à professora, colega e amiga Adria Lyra, e também aos meus alunos de Computação I e II da UFRJ, que muito me ensinaram e ajudaram a encontrar minha vocação em lecionar, sonho que buscarei a partir de agora.

E, por fim, um agradecimento especial ao meu avô, Jorge Luiz, que não mais está aqui. Que fez e faz muita falta todos os dias. Seguirei tentando levar a vida como você ensinou. Obrigado por tudo.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

A DATA-DRIVEN APPROACH FOR ESTIMATING TEMPERATURE VARIATIONS BASED ON B-MODE ULTRASOUND IMAGES AND CHANGES IN BACKSCATTERED ENERGY

Luiz Fernando dos Reis de Oliveira

Dezembro/2023

Orientadores: Priscila Machado Vieira Lima
Wagner Coelho de Albuquerque Pereira
Felipe Maia Galvão França

Programa: Engenharia de Sistemas e Computação

Terapias baseadas em aquecimento são aquelas onde uma fonte de calor é utilizada para auxiliar no tratamento de condições que variam deste inflamações até alguns tipos de câncer. Neste tipo de procedimento médico, o monitoramento da variação da temperatura é de vital importância pois, se a janela de temperatura desejada não for alcançada, a eficácia do tratamento por ser limitada ou é possível até mesmo agravar o problema existente. Nesse sentido, o presente trabalho propõe uma modelagem para tratar o problema de estimar a variação de temperatura em imagens de ultrassom em modo B possibilitando a utilização de modelos de aprendizado de máquina. Para validar a modelagem, foram explorados modelos baseados em árvores, topologias simples de redes neurais profundas e redes neurais sem peso modificadas para problemas de regressão. A modelagem proposta lida apenas com informações providas pelas imagens de ultrassom, incorporando informações obtidas a partir de uma imagem paramétrica construída com base nas mudanças de energia de retroespalhamento (CBEUS). Os resultados computacionais mostram que, em um cenário simulado, a modelagem proposta em conjunto com o modelo Gradient Boosting Decision Tree (GBDT) é capaz de estimar a variação de temperatura com um erro médio absoluto próximo de 0.5^{o}C, o que é considerado aceitável em ambientes práticos tanto em tratamentos fisioterapêuticos quando em ultrassom focalizado de alta intensidade (HIFU).

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A DATA-DRIVEN APPROACH FOR ESTIMATING TEMPERATURE VARIATIONS BASED ON B-MODE ULTRASOUND IMAGES AND CHANGES IN BACKSCATTERED ENERGY

Luiz Fernando dos Reis de Oliveira

December/2023

Advisors: Priscila Machado Vieira Lima
Wagner Coelho de Albuquerque Pereira
Felipe Maia Galvão França

Department: Systems Engineering and Computer Science

Heating-based therapies are those where a heat source is used to assist in treatments of conditions that ranges from inflammation to forms of cancer. In this type of medical procedure, monitoring temperature variation is vitally important because, if the desired temperature range is not reached, treatment may be limited or even worsen the existing problem. In this sense, this work aims to propose a modeling to address the problem of estimating temperature variation in regions of B-mode ultrasound images that allows the use of machine learning models. To validate the modeling, tree-based models, simple topologies of deep neural networks and weightless neural networks adapted for regression problems were explored. The proposed modeling aims to deal only with information provided by the ultrasound images, incorporating data that can be obtained from a parametric image constructed based on changes in backscattered energy (CBEUS). The computational results showed that, in a simulated scenario, the proposed approach with the Gradient Boosting Decision Tree model would be able to estimate the temperature with an mean absolute error close to 0.5^{o}C, which is acceptable in practical environments both in physiotherapeutic treatments and high intensity focused ultrasound (HIFU).

Contents

List of Figures **x**

List of Tables **xxi**

1 Introduction **1**
1.1 Contributions . . . 3
1.2 Document Organization . . . 3

2 Background on Ultrasound **5**
2.1 Introduction . . . 5
2.2 Ultrasonography . . . 7
2.3 Therapeutic Ultrasound . . . 8
2.4 Temperature Estimation . . . 10
2.5 Related Work . . . 11

3 Proposed Data Modeling **14**
3.1 CBEUS . . . 14
3.2 Proposed Approaches . . . 16
3.3 Machine Learning Models . . . 20
3.3.1 Tree-based methods . . . 20
3.3.2 Deep Neural Networks . . . 22
3.3.3 RegressionWiSARD . . . 23
3.3.4 ClusRegressionWiSARD . . . 24

4 Computational Results **26**
4.1 Experimental Framework . . . 26
4.1.1 Details on Simulation Data . . . 26
4.1.2 Details on $In-Vitro$ Data . . . 28
4.1.3 Computational Environment . . . 29
4.2 Numerical Results . . . 30
4.3 Results on Simulation Data . . . 31
4.4 Results on *In-Vitro* Data . . . 38

5 Discussion **46**
5.1 Initial Considerations . . . 46
5.2 Error Analysis on Simulation Data . . . 47
5.3 Error Analysis on $in-vitro$ Data . . . 59
5.4 Error Analysis on Muscle Tissue . . . 71

6 Conclusion **73**
6.1 Future Work . . . 74

References **76**

A Publications **84**
A.1 Journal articles . . . 84
A.2 Book chapters . . . 84
A.3 Complete works published in proceedings of conferences . . . 84
A.4 Extended abstracts published in proceedings of conferences . . . 85

B Weightless Neural Networks **86**
B.1 WiSARD . . . 86
B.1.1 Bleaching . . . 88
B.2 Binary Encoding . . . 89
B.2.1 Thermometer Encoding . . . 89
B.2.2 Distributive Thermometer . . . 92
B.3 ClusWiSARD . . . 92
B.4 Applications of WiSARD-based Models . . . 93
B.5 Weightless Models for Regression . . . 94
B.5.1 n-Tuple Regression Network . . . 94
B.5.2 Interpolation Capability Comparison . . . 95

C Results on Hyperparameter Exploration of the ReW Model **98**
C.1 Models Evaluation . . . 98

D Heuristics for Selecting Thermometer sizes **113**
D.1 Experimental Framework . . . 113
D.1.1 Datasets Description . . . 113
D.1.2 Experiments Description . . . 115
D.2 Results and Discussion . . . 116

List of Figures

2.1 Description of a sound wave in terms of frequency its wavelength λ and amplitude A 6
2.2 Example of B-mode ultrasound. This figure was retrieved from the website: `https://commons.wikimedia.org/wiki/File:Dilated_cardiomyopathy_B-Mode.jpg` 9

3.1 Process for creating a CBEUS image. a) set of images ordered by temperature level in ascending order. b) small section of the images with grayscale values of the pixels. c) the resulting slope parameters for each pixel in the parametric image. d) the final CBEUS image obtained after pixel value normalization. 16
3.2 Illustration of the possible arrangements for generating CBEUS images from a set of B-mode images ordered by represented temperature. 17
3.3 Pictorial representation of the first data modeling approach used. By considering two B-mode temperature images, the same pixel of both images is selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images. 17
3.4 Pictorial representation of the second data modeling approach used. By considering a B-mode temperature image and the CBEUS image computed with a certain temperature variation, the same pixel of both images is selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images. 18

3.5 Pictorial representation of the third data modeling approach used. By considering a B-mode temperature image and the CBEUS image computed with a certain temperature variation, for the same pixel of both images, a neighborhood of size N for the temperature image and the single pixel are selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images. A valid pixel implies that the neighborhood of size N can be retrieved, i.e. it is within the image bounds. 19
3.6 Pictorial representation of the fourth data modeling approach used. By considering a B-mode temperature image and the CBEUS image computed with a certain temperature variation, for the same pixel of both images, a neighborhood of size N is selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images. A valid pixel implies that the neighborhood of size N can be retrieved, i.e. it is within the image bounds. 19
3.7 Classic tree data structure. 20
3.8 Examplification of the boosting technique. 21
3.9 Description of the training process of the RegressionWiSARD model. A binary pattern is presented alongside its expected output. The pattern is divided into n tuples and each tuple is addressed to a RAM node. The address formed by the tuple increments the counter by 1 and the *sum* by the expected output. 24
3.10 Example representation of the CReW model process to select a ReW model for training. 25

4.1 Digital phantom used for B-mode image generation. White pixels simulate muscle tissue, black pixels simulate lipid tissue (fat), and the gray region simulates water. This figure is based on the one presented in [38]. 27
4.2 Collection of the resulting B-mode images representing different temperatures obtained using the numeric phantom. In each figure, the related temperature is the same across the whole image. 27

4.3 Resulting B-mode images representing temperatures of 36ºC and 45ºC obtained using *in vitro* porcine skeletal striated muscle. In each figure, the related temperature is the same across the whole image. We use different images for all integer temperature values between the ones displayed in the figure, but they do not present significant differences visually and were omitted for simplification purposes. . . 29
4.4 Comparison of the experimental results considering how the MAE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MAE for the 30% reserved data. 35
4.5 Comparison of the experimental results considering how the MSE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MSE for the 30% reserved data. 36
4.6 Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data. 36
4.7 Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data. In this figure, the RF model was suppressed to evaluate the other models better. 37
4.8 Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data. 37
4.9 Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data. In this figure, the DNN and CReW models were suppressed to evaluate the other models better. . 38

4.10 Comparison of the experimental results considering how the MAE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MAE for the 30% reserved data. 43
4.11 Comparison of the experimental results considering how the MSE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MSE for the 30% reserved data. 43
4.12 Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data. 44
4.13 Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data. In this figure, the RF model was suppressed to evaluate the other models better. 44
4.14 Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data. 45
4.15 Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data. In this figure, the DNN and CReW models were suppressed to evaluate the other models better. . 45

5.1 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 1$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 48

5.2 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 2$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 48
5.3 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 3$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 49
5.4 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 4$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 49
5.5 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 5$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 50
5.6 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 6$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 50
5.7 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 7$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 51

5.8 KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 8$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. 51
5.9 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 8ºC. 53
5.10 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 7ºC. 53
5.11 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 6ºC. 54
5.12 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 5ºC. 54
5.13 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 4ºC. 55
5.14 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 3ºC. 56
5.15 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 2ºC. 57
5.16 Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 1ºC. 58
5.17 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 1$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 59

5.18 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 2$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 60
5.19 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 3$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 60
5.20 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 4$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 61
5.21 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 5$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 61
5.22 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 6$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 62
5.23 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 7$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 62

5.24 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 8$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature. 63
5.25 KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 9$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. 63
5.26 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 9ºC. 64
5.27 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 8ºC. 64
5.28 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 7ºC. 65
5.29 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 6ºC. 65
5.30 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 5ºC. 66
5.31 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 4ºC. 67
5.32 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 3ºC. 68
5.33 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 2ºC. 69
5.34 Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 1ºC. 70

5.35 Error visualization of muscle-related region in the phantom figure. a) the standard heatmap considering the base image at 37ºC and temperature variation $\Delta = 1$, as presented in Figure 5.9. b) The same heatmap but displaying only the regions related to muscle tissue in the phantom. 72

B.1 Example of discriminators in a digits classification scenario. A total of 10 discriminators, each one responsible for recognizing a single digit. In the training phase, the input is addressed to the corresponding discriminator. 87
B.2 Training phase of the WiSARD model. The pseudo-random mapping creates a new order of the input bits and the new input is divided according to the defined address size. The counter value c of the corresponding address in each RAM node registers the activation. . . 87
B.3 Classification phase of the WiSARD model. The same pseudo-random mapping used in training creates a new order of the input bits and the new input is divided according to the defined address size. For each RAM node, the corresponding address is verified and, if the memory address was accessed, the node is considered activated and the discriminator reports the number of active nodes. 88
B.4 To complete the classification of the presented pattern, the WiSARD model compares the response of all discriminators. The one with the highest response defines the class. 88
B.5 Illustrative example of the linear thermometer. 90
B.6 Correlation matrix for the Iris dataset. 91
B.7 Comparison between the thresholds generated by a) the conventional linear thermometer and b) the distributive thermometer. 92
B.8 ReW interpolation capability. The blue line indicates the real function u(k), while the orange line indicates the predicted values from ReW. From the results obtained, it is possible to observe limitations regarding three of the analysed averages as they are unable to generate a continuous line. 96

C.1 MAE measures - Hyperparameter exploration on the Abalone Dataset. 101
C.2 Training time measures - Hyperparameter exploration on the Abalone Dataset. 102
C.3 Prediction time measures - Hyperparameter exploration on the Abalone Dataset. 103
C.4 MAE measures - Hyperparameter exploration on the Auto Imports Dataset. 104

C.5 Training time measures - Hyperparameter exploration on the Auto Imports Dataset. 105
C.6 Prediction time measures - Hyperparameter exploration on the Auto Imports Dataset. 106
C.7 MAE measures - Hyperparameter exploration on the Auto Insurance Dataset. 107
C.8 Training time measures - Hyperparameter exploration on the Auto Insurance Dataset. 108
C.9 Prediction time measures - Hyperparameter exploration on the Auto Insurance Dataset. 109
C.10 MAE measures - Hyperparameter exploration on the House Prices Dataset. 110
C.11 Training time measures - Hyperparameter exploration on the House Prices Dataset. 111
C.12 Prediction time measures - Hyperparameter exploration on the House Prices Dataset. 112

D.1 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Breast Cancer dataset. 118
D.2 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Breast Cancer dataset. 119
D.3 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Breast Cancer dataset. 120
D.4 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Diabetes dataset. . 121
D.5 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Diabetes dataset. . 122
D.6 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Diabetes dataset. . 123
D.7 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Ionosphere dataset. 124
D.8 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Ionosphere dataset. 125
D.9 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Ionosphere dataset. 126

D.10 Mean Absolute Error score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the House Prices dataset. 127
D.11 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the House Prices dataset. 128
D.12 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the House Prices dataset. 129
D.13 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Iris dataset. 130
D.14 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Iris dataset. 131
D.15 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Iris dataset. 132
D.16 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Sonar dataset. . . 133
D.17 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Sonar dataset. . . 134
D.18 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Sonar dataset. . . 135
D.19 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wheat Seeds dataset. 136
D.20 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wheat Seeds dataset. 137
D.21 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wheat Seeds dataset. 138
D.22 Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wine dataset. . . 139
D.23 Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wine dataset. . . 140
D.24 Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wine dataset. . . 141

List of Tables

4.1 Information regarding the generation of datasets using each of the proposed approaches based on simulation data. The table shows, for each approach, the number of input features, the total size of the dataset, and the size of each partition after dividing the training set and the prediction set. 28

4.2 Information regarding the generation of datasets using each of the proposed approaches based on real data. The table shows, for each approach, the number of input features, the total size of the dataset, and the size of each partition after dividing the training set and the prediction set. 29

4.3 Parameters of the models Random Forest (RF), Gradient-boosting Decision Tree (GBDT), Deep Neural Network (DNN), Regression-WiSARD (ReW) and ClusRegressionWiSARD (CReW) that returned the best results across the computational experiments. The values were obtained empirically after exploring a finite set of different configurations. 30

4.4 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #1. 31

4.5 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #1. 31

4.6 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #2. 31

4.7 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #2. 32

4.8 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 1. 32

4.9 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 1. 32
4.10 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 3. 32
4.11 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 3. 33
4.12 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 5. 33
4.13 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 5. 33
4.14 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 1. 33
4.15 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 1. 34
4.16 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 3. 34
4.17 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 3. 34
4.18 Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 5. 34
4.19 Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 5. 35
4.20 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #1. 38
4.21 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #1. . . . 39
4.22 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #2. 39

4.23 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #2. . . . 39
4.24 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #3 and neighborhood size N = 1. . . . 39
4.25 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #3 and neighborhood size N = 1. . . . 40
4.26 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #3 and neighborhood size N = 3. . . . 40
4.27 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #3 and neighborhood size N = 3. . . . 40
4.28 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #3 and neighborhood size N = 5. . . . 40
4.29 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #3 and neighborhood size N = 5. . . . 41
4.30 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #4 and neighborhood size N = 1. . . . 41
4.31 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #4 and neighborhood size N = 1. . . . 41
4.32 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #4 and neighborhood size N = 3. . . . 41
4.33 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #4 and neighborhood size N = 3. . . . 42
4.34 Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #4 and neighborhood size N = 5. . . . 42
4.35 Computational results regarding training and prediction time for estimating temperature variation on real data using approach #4 and neighborhood size N = 5. . . . 42

5.1 Average degree of imbalance between the different temperature variations existing in the datasets based on the simulation and real data. 71
5.2 Behavior of the angular coefficients of the CBEUS parametric image when splitting the image between the different types of tissue in the digital phantom. 72

B.1 Comparison of MAE values from ReW models while varying the type of average. It is possible to observe that Median appears as a better type of computing the output in contrast to the Simple Mean as suggested by the original n-tuple regressor for this particular task. . . 97

C.1 Regression models configuration 99
C.2 Results on the Abalone dataset. 99
C.3 Results on the Auto Imports dataset. 99
C.4 Results on the Auto Insurance dataset. 100
C.5 Results on the House Prices dataset. 100

D.1 Statistics regarding the datasets used in the experiments. 142

Chapter 1

Introduction

Thermotherapy is a type of medical treatment that is based on the variation of tissue temperature, where this variation can be considered both as increasing or decreasing direction [1]. In general, thermotherapy applications act superficially, so that other types of procedures are required to reach deeper tissues. When considering temperature increase, heat can be obtained from the use of infrared radiation or ultrasound.

Ultrasound is characterized as any mechanical vibration with a frequency greater than 20 kHz [2]. In medicine, ultrasound devices are commonly associated with image diagnosis, but the use of therapeutic ultrasound for treating injuries or certain types of cancer has wide application due to its low cost and because it is a harmless and non-invasive procedure [3].

Among the existing thermal therapies that use ultrasound as a heat source, hyperthermia is particularly prominent. It is a thermal treatment for some kinds of cancer, where the region is heated to temperatures close to 45^{o}C, which improves the effectiveness of procedures like radiotherapy [4]. Ultrasound is also used in physiotherapeutic treatments. It is applied, for example, to treat inflammatory conditions by increasing the blood flow and accelerating healing processes. [5]. Being a treatment that requires a specific temperature window to work and since heat does not spread equally, online temperature tracking is an essential procedure. The treatment would be ineffective if the temperature is below the required range. On the other hand, reaching higher temperatures could be harmful, causing burns and undesired effects.

However, tracking the temperature in deep tissues is a challenging task. The gold standard for measuring the temperature is through invasive techniques using sensing probes, specifically sensors like thermocouples that need to be in direct contact with the region [6]. This type of procedure is not always a viable option, which leads to an investigation of non-invasive techniques for temperature monitoring, and Magnetic Resonance Thermography (MRT) is presented as an alternative. It is a non-ionizing

procedure that can retrieve information that is clinically accepted but has major drawbacks, like high cost and the difficulty of designing systems that are not affected by the scanner's magnetic field [7].

Backscattered energy is an acoustic property that is a topic of interest in research related to temperature estimation. It is the portion of the energy that travels back to the transducer after it interacts with a medium or an object. Other works aimed at studying the impact of changes in backscattered energy (CBE). In [8] the authors demonstrate how the power of backscattered ultrasound is dependent on temperature. [9] present analysis where CBE varies monotonically depending on the medium type, and [10] produced 2D temperature maps based on echo shifts and CBE of acoustic harmonic in the temperature range of hyperthermia.

Machine learning (ML) rises as a potential candidate in many domains - including healthcare - when talking about automatic methods to process both structured and unstructured data. From supporting prognosis [11] to highlighting hidden patterns in medical imaging generation [12], the application of artificial intelligence algorithms and usage of intelligent agents has reportedly enhanced both productivity and quality not only in the medical domain but in various sectors of the modern society.

Machine learning methods are usually applied to image processing tasks when dealing with ultrasound. In [13] the authors present a literature review showing applications on image classification (the system must decide whether the image belongs to a class or not), image detection (discover if and where the desired element is present in the image), image segmentation (isolate parts of the image that correspond to distinguished elements) and 3D ultrasound analysis. [14] describes an application on obstetrics in which a classification system based on support vector machines is used to detect fetal presentation by analyzing features extracted from images. [15] shows a deep neural network to reconstruct ultrasound images from radio-frequency data.

The goal of this work is to propose a data modeling for the problem of estimating the temperature variation of B-mode images as a supervised learning problem by proposing the transformation of the image information into a structured tabular format that enables the use of machine learning models to estimate the variation in a non-invasive way. In particular, the proposed modeling resort to the use of a parametric image built based on CBE.

As stated earlier, one of the benefits of ultrasound is that it is a cheaper solution compared to other applications, and it would be desirable to include machine learning-based solutions that cause minimal impacts in cost while retaining good performance. Weightless neural models (WNN) have reportedly presented advantages like their simplicity to be implemented as hardware, low training time, good

convergence with few examples, and a steep learning curve [16].

One of the most famous WNN is the WiSARD (Wilkie, Stonham, and Aleksander Recognition Device) [17], a classifier that is formed by a group of multiple discriminators built using RAM-like structures that operates as the network neurons. Since the WiSARD model was developed for classification tasks, and the problem of estimating the temperature variation requires the output to be a continuous value, it was necessary to perform a modification on the model structure to enable it to work in regression tasks. In this sense, this work also proposes the RegressionWiSARD (ReW) and ClusRegressionWiSARD (CReW) models, which are extensions of the WiSARD models for regression. Both models were initially applied to classic machine learning datasets to explore parameter tuning.

1.1 Contributions

During the development of this work, the following contributions were presented:

- proposal of novel modeling for the problem of estimating temperature variation in ultrasound images that enables the use of machine learning models;
- exploration of three groups of machine learning models to evaluate the proposed modeling;
- presentation of two architecture extensions of the WiSARD models for performing regression tasks;
- comparison of the proposed extensions with classic regression models in different scenarios;

1.2 Document Organization

The remainder of this thesis is organized as follows:

- Chapter 2 presents background on medical ultrasound, introducing the basic aspects that enable understanding the application scenario.
- Chapter 3 describes the proposed data modeling for estimating temperature variation as a supervised learning problem.
- Chapter 4 presents the structure of the experiments and the main results obtained from the experiments,
- Chapter 5 shows an exploratory data analysis to understand the results better.

- Chapter 6 summarizes the general conclusions and points to the next steps for this research.
- Appendix A list all published works developed during this period.
- Appendix B presents background on weightless neural models the two proposed weightless neural models for regression tasks, RegressionWiSARD and ClusRegressionWiSARD.
- Appendix C shows graphical results on hyperparameter exploration of the proposed regression models when applied to a set of classic regression datasets and discusses these results.
- Appendix D presents a heuristic proposed for selecting the number of bits addressed to a feature when applying a thermometer encoding.

Chapter 2

Background on Ultrasound

This chapter starts with a background on ultrasound, contextualizing fundamental principles and characteristics, followed by the subsequent exploration of therapeutic ultrasound and its applications, which is the focal domain of this work. Particularly, the main problem that is addressed in this work is the non-invasive temperature measurement in hyperthermia, and a literature review on different approaches for this problem is performed.

2.1 Introduction

Sound is a mechanical wave that is created by an object's vibration, such as a speaker or musical instruments like guitars and drums. This vibration causes the air molecules around the object to compress and expand, creating a pressure wave that travels through the air. When this wave reaches an ear structure, it causes the eardrum to vibrate, which turns the signals to the brain that are interpreted as sound. Sound can also travel through other mediums such as solids and liquids. Sound is measured in decibels (dB) and the loudness of sound is known as sound intensity. As it propagates, it can lose intensity along the wave path, as it interacts with medium properties.

Being a wave, sound can be described by components like frequency (how often a wave oscillates), wavelength (the distance between two peaks), and amplitude (the height of the wave), as illustrated in figure 2.1. Typically, the human ear can detect sounds that present a frequency between 20 Hz and 20,000 Hz.

When a sound wave reaches a frequency higher than 20,000 Hz it is defined as ultrasound [2]. At this frequency, these waves are not audible to the human ear. Although it cannot be heard, ultrasound waves have a wide range of applications in various fields, such as:

- Sonar: used in localization systems for tasks like navigation and object detec-

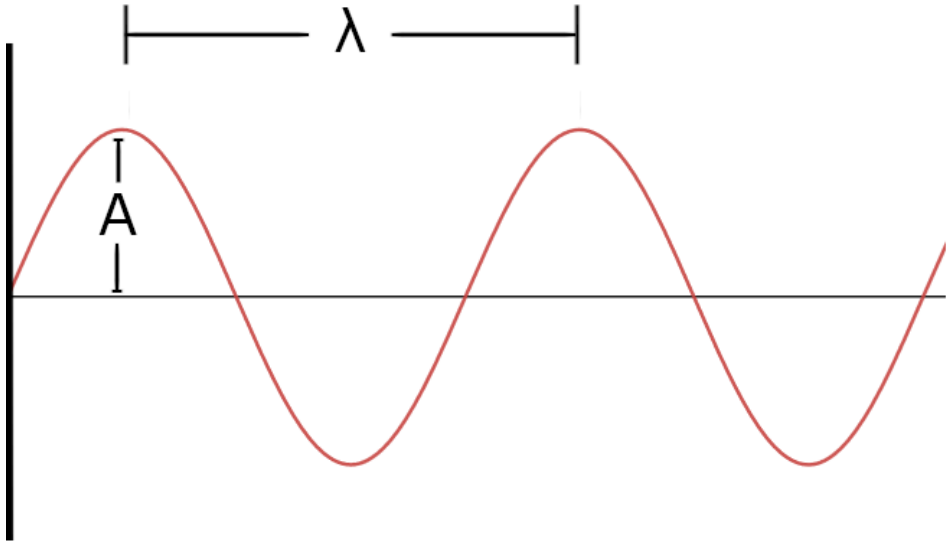

Figure 2.1: Description of a sound wave in terms of frequency its wavelength λ and amplitude A

tion [18].

- Industrial Inspection: ultrasound testing is a type of non-destructive testing (NDT). Ultrasound can be used for integrity inspection of components such as pipes or pressure vessels. It can detect internal defects like cracks and corrosion that would not be detected by the human eye [19].
- Industrial Cleaning: to remove dirt, oil, and other substances from surfaces by creating microscopic bubbles that implode and create tiny shockwaves that remove the desired material [19].
- Pest control: to repel pests such as rats, bats, and other animals. The high-frequency sound waves are used to create an environment that is uncomfortable for pests, turning them to leave [20].
- Cosmetics: used alongside cosmetics and personal care procedures to help remove dirt from the skin and dead skin cells [21].

Besides the above-mentioned applications, ultrasound is also commonly associated with the medical domain. The Austrian neurologist Karl Theo Dussik is referred to as the first to apply ultrasound as a medical diagnostic tool, using it to generate brain images [22]. In general terms, there are two major uses of ultrasound in medicine: ultrasonography and therapeutic ultrasound.

2.2 Ultrasonography

Ultrasonography, also known as diagnostic sonography, is an imaging procedure that uses ultrasound waves to create images of internal organs, blood vessels, and other inner structures of the human body. Ultrasound images are created by sending sound waves into the body and then measuring the echoes that are returned. These echoes are then used to create a detailed image of the structure or organ being examined. Ultrasonography is a non-invasive and safe procedure, as it does not use ionizing radiation, and it is relatively inexpensive compared to other imaging techniques such as computed tomography (CT) or magnetic resonance imaging (MRI).

It can be used to monitor and diagnose a wide range of medical conditions, including pregnancy, tumors, organ damage, blood clots, and more. In obstetrics and gynecology, ultrasound images are used to monitor the growth and development of a fetus during pregnancy. Also, it can help detect anomalies in the abdominal region and vascular system, for example, [23].

For generating images, a device known as a transducer is used [24]. It both emits the ultrasound waves and also captures the waves that reflect on body structures and return. All transducers have piezoelectric elements, usually made of ceramic or polyvinylidene fluoride that convert electrical energy into mechanical energy and vice versa. This crystal is responsible for creating and receiving the ultrasound waves. There are several types of ultrasound transducers, each with a specific purpose and design. Some of the most common types include:

- Linear transducers: have a linear array of elements that emit and receive sound waves. They are used for applications such as abdominal imaging, and musculoskeletal imaging.
- Convex transducers: have a curved surface that allows them to scan a larger area of the body. They are used for applications such as obstetrics and gynecology, as well as abdominal imaging.
- Electronically focalized transducers: have a matrix of elements that emit and receive sound waves. They can be electronically focused, allowing them to image deeper structures and improve image resolution. They are used in applications such as cardiac imaging, and vascular imaging.

The choice of the transducer depends on the type of exam to be performed and the type of tissue or organ to be imaged [25]. Based on that, the chosen transducer is then moved over the area of the body being examined, and the images are displayed on a monitor for the doctor to review. Different modes of ultrasound images can provide different information and can be used to evaluate a wide range of conditions and diseases. Some of these modes are described as follows:

- B-mode (Brightness mode): typically used for diagnostic purposes, such as detecting tumors, cysts, and other abnormalities. Also used to measure the size and shape of organs, and to monitor the growth of a fetus during pregnancy.
- M-mode (Motion mode): provides one and bi-dimensional views of internal structures. Usually used to evaluate cardiac function and to measure the size and movement of internal organs.
- Doppler mode: used to evaluate blood flow in the body. It can be used to evaluate blood flow in the uterus, placenta, and fetal heart during pregnancy.
- Power Doppler mode: similar to Doppler mode, but it is more sensitive and can detect even small amounts of blood flow.
- Color Doppler mode: combines Doppler information with B-mode images to create a color-coded map of blood flow. It can be used to evaluate blood flow in the uterus, placenta, and fetal heart during pregnancy.
- Spectral Doppler mode: displays the speed and direction of blood flow in a line graph format, it is used to evaluate blood flow in the vessels, such as the carotid artery, and it can also be used to detect any blockages or problems with blood flow.
- 3D images: provide a more detailed, three-dimensional view of the body and are used to visualize the fetus during pregnancy, as well as to examine the shape and structure of organs such as the heart and liver.

In this work, the focus is on the analysis of B-mode ultrasound images, which are gray-level images created based on the measurement of the amplitude of ultrasound waves' echoes that reflect after interacting with internal organs and tissues. The strength of the echoes is represented by different shades of gray in the image, with brighter areas indicating stronger reflectors while dark areas indicate weaker ones. An example of a B-mode ultrasound image can be seen in figure 2.2. The choice of this type of image was because it is the ultrasound image variation used in most practical situations.

2.3 Therapeutic Ultrasound

Therapeutic ultrasound, also known as therapeutic ultrasonography, is a medical treatment that uses ultrasound waves to promote healing and reduce pain. The ultrasound waves are delivered to the body and penetrate deep into the tissue, creating heat and stimulating blood flow and tissue repair, which helps reduce inflammation

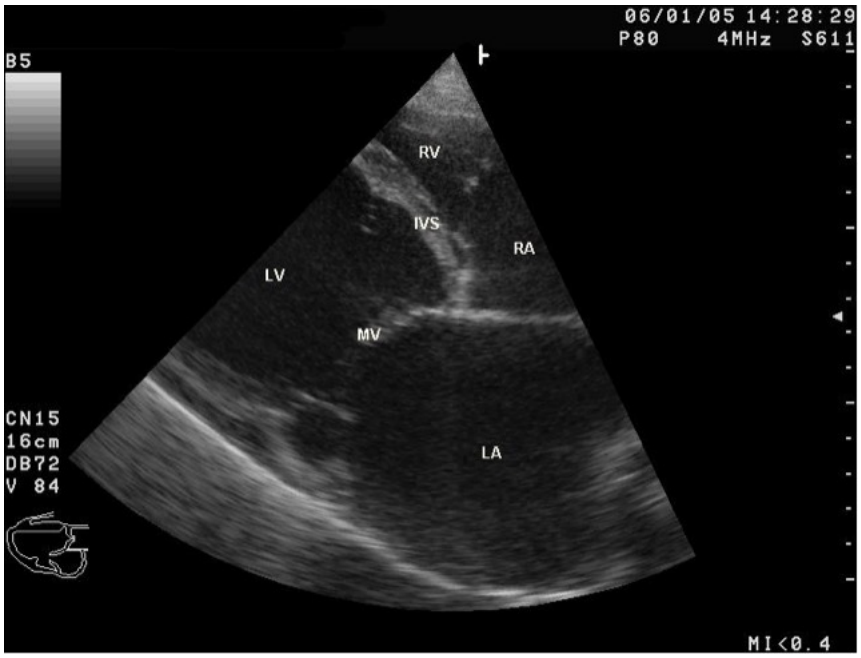

Figure 2.2: Example of B-mode ultrasound. This figure was retrieved from the website: `https://commons.wikimedia.org/wiki/File:Dilated_cardiomyopathy_B-Mode.jpg`

and promote healing. Additionally, the heat generated by the ultrasound waves can help reduce pain and muscle spasms. Therapeutic ultrasound is used to treat a variety of conditions, including:

- Musculoskeletal conditions such as tendinitis, bursitis, and muscle strains
- Soft tissue injuries such as sprains and strains
- Wounds and ulcers
- Bone healing and fractures

Therapeutic ultrasound is a non-invasive treatment with minimal side effects, and it can be used in combination with other treatments such as exercise, physical therapy, and medication. It is encapsulated in a group of procedures known as thermotherapy, which uses heat to relieve pain and muscle spasms, increase circulation, and promote healing. And, besides ultrasound waves, other forms of electromagnetic energy can also be used. Thermotherapy is often used to treat conditions such as arthritis, menstrual cramps, and muscle strains. One important application of thermotherapy is cancer treatment, in the form of procedures like thermal ablation and hyperthermia.

Thermal ablation uses heat energy to destroy or remove tumors. The energy is delivered to the tissue through a probe guided to the target area using imaging

techniques such as ultrasonography. Thermal ablation is often used to treat benign or malignant tumors in the liver, kidney, and lung, among others.

Hyperthermia is a form of thermotherapy that uses heat to raise the temperature of cancerous tissue to kill the cancer cells. [26] The idea behind hyperthermia is that cancer cells become more sensitive to drug therapy than normal cells. So, by raising the temperature of the cancerous tissue, the cancer cells will be killed while leaving the normal cells unharmed. Among different types of hyperthermia treatments, it could be cited:

- Whole-body: raises the entire body's temperature. It is used to treat advanced-stage cancers such as metastatic melanoma.
- Regional: raises the temperature of a specific area of the body, such as an organ. It is used to treat localized cancers such as prostate or breast cancer.
- Focal: raises the temperature of a specific area of the body. It is used to treat small tumors and cancer that have not spread to other parts of the body.

In hyperthermia, ultrasound is used in a technique called High-Intensity Focused Ultrasound (HIFU) [27]. HIFU therapy works by focusing the ultrasound energy at a certain point. The high energy at the focal point causes the temperature to rise rapidly, leading to cell death in the targeted area. The sound waves are precisely focused and the energy is delivered at a high intensity to the target area, while the surrounding healthy tissue is spared. [28]

2.4 Temperature Estimation

Since ultrasound images do not directly provide information regarding temperature. It is possible, however, to estimate temperature changes in tissue using techniques such as thermal imaging. Thermal imaging utilizes the differences in ultrasound backscatter between regions of different temperatures to create a thermal map of the tissue. [29].

It is important to notice that these techniques are not as accurate as direct temperature measurements and are affected by several factors such as tissue type and composition, power settings, and probe frequency. Thus, the main objective is to estimate temperature within a certain margin error, which, according to the literature, is desired to be placed equal to or lower than 0.5 degree Celsius [30].

In thermotherapy procedures, the control of temperature variation is of great interest, as the correct level of heating/cooling is directly associated with the procedure's success. In a heating scenario, if the temperature is not high enough, the

procedure may have little to no effect. On the other hand, if the temperature is too high, it could cause burns in tissue and aggravate the problem.

2.5 Related Work

Non-invasive temperature measurement has been a subject of interest in several works over the past decades. [31] presents a handful of examples where invasive monitoring methods can be questioned, like patients' low acceptance and the possibility of undesired side effects. The authors, then, developed a study comparing the effectiveness of thermal monitoring using less-invasive (monitor application in shallow organs) and non-invasive techniques (magnetic resonance tomography). It was concluded that, even though there are limitations and specific application scenarios, both approaches should be considered and further explored.

In one of the first works to approach ultrasound for non-invasive thermometry, the authors in [32] analyze the modification in the backscattered signal generated by a given tissue. This was motivated by the fact that the signal has a direct dependence on the tissue properties and these properties change based on the temperature of the tissue. In particular, it is stated that the backscattered power on the considered temperature range can be retrieved by the following equation:

$$P_r(T) = \frac{2H^2\delta}{8R^4\alpha(T)}\eta(T)S(1 - e^{-2\alpha(T)c(T)\tau}) \tag{2.1}$$

where $\frac{H}{R}$ and δ are the amplitude and duration of the insonifying sinusoidal burst, respectively; R is the distance from the transducer to the scattering volume of tissue; α(T) is the attenuation within the tissue volume as a function of temperature; c(T) is the speed of sound in the tissue volume as a function of temperature; and η(T) is the backscatter coefficient of the tissue volume as a function of temperature.

In [33], the authors question the results of previous works that proposed temperature estimation methodologies without considering a realistic temperature rise using body temperature baseline. Besides, it also questions the phantom material of the works (rubber), which presents different properties comparing to muscle or lipid tissue.

The work evaluates three liver samples, each one containing normal, intermediate, and high-fat percentages, and it aims to promote a method for visualizing the heated region based on analysis of the ultrasound echo strain.

A different method used for the non-invasive temperature estimation is the *average gray-level* (AVGL) [34]. Given a defined rectangular region with dimensions MxN, the absolute average gray-level is computed as

$$AVGL_{ab} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} p_{ij} \tag{2.2}$$

where p_{ij} is the gray-level of the pixel in the position $i \times j$. The authors use a sample of the porcine kidney (with both muscular and lipid tissue) and measure the temperature continuously in a range of 36^oC - 45^oC in two different regions of the sample while generating corresponding images at each given temperature, which leads to the development of a relation mapping (e.g. a function) between the AVGLs and the temperatures. The temperature variation was performed using a water bath-controlled system.

The measurements indicated a high correlation between the two variables (AVGL and temperature), with a minimum correlation of 0.98. By displaying scatter plots and fitting a function to approximate the distributions, a maximum error of 0.25^oC. It must be stated, though, that this error was obtained through an interpolation method. In other words, it is a function that describes the distribution of measured points, and no evaluation of unseen data was performed.

Next, the authors perform an extended exploration of how to use the AVGLs for temperature estimation [35]. A similar temperature-controlled system was used, but a bovine muscle sample was chosen for the experiments. Also, a different approach was taken, where the heating source was a therapeutic ultrasound. By using these different approaches, it was found that while in a water bath, the AVGLs presented a linear relationship concerning temperature, the therapeutic ultrasound promoted a quadratic relationship.

In [36] the authors further extend the research by starting an evaluation study on a metric known as *uncertainty*, which is a description of the difference between the measured value and the actual value in real life. The experiments followed the previous work's design using a plastic phantom that was placed on a reservoir with controlled temperature changes. Based on the measurements, they computed a regression model to describe the variation between metrics.

The experiments were performed 10 independent times, and the regression model was evaluated through the usage of the leave-one-out method, where, at each given time, one test was removed and the regression model was performed using the other nine tests. Then, the one remaining test's actual result was compared to the predicted value of the model. It was observed that in a worst-case scenario, the prediction error was around 1^oC.

In a later work [37], the authors expand the studies regarding uncertainty measurement by dealing with a tissue-mimicking material using an ultrasound-based heating mechanism. The analysis was performed both in heating and cooling procedures. As previously mentioned, the type of heating source impacts the relationship

between the AVGLs and temperature variation. The interference of the therapeutic ultrasound in the uncertainty computation was noticed since the value obtained during the heating procedure (2.5^{o}C, and where the device was active) was higher than the one obtained during the cooling procedure (1.9^{o}C).

Chapter 3

Proposed Data Modeling

This chapter presents the proposed data modeling, which is a novel methodology that enables the application of machine learning methods to the temperature variation estimation problem. It starts by explaining CBEUS, a parametric image generated based on changes in backscattered energy that was proposed in [38] and is used as a resource for the proposed modeling. Then, it follows up by explaining the proposed data modeling, showing the different approaches that will be evaluated in the following chapter. It concludes by presenting a summary of some of the machine learning models that will be used in the experiments. It is worth noticing that the examples illustrated in this chapter use B-mode ultrasound images that will be better described in the Experimental Framework section in Chapter 4.

3.1 CBEUS

In [38] the authors explore a new imaging modality in the analysis of intensity changes of pixels related to temperature changes, referred to as CBE-based Ultrasound (CBEUS) Imaging. The novel method proved to be efficient not only by maintaining the ability to distinguish structures but also by providing the identification of new ones.

During the analysis, the authors noticed that the behavior during heating of a single pixel location could be fit by a linear model and that the angular coefficient (slope parameter) of the linear model was different according to the type of tissue. So, by mapping the angular coefficient, it would be possible to describe the different tissue compositions.

The general process for creating a CBEUS image uses a set of B-mode ultrasound images ordered by temperature level in ascending order. For each pixel location (i, j), the linear model is used to fit the pixel values across all images:

$$\Delta I_{i\times j}(\Delta T) = a_{i\times j}\Delta T + b_{i\times j} \tag{3.1}$$

where $\Delta I_{i\times j}(\Delta T)$ is the temperature-dependent intensity change for some pixel (i, j), ΔT is the temperature change measured, and $a_{i\times j}$ and $b_{i\times j}$ are the linear model parameters for pixel (i, j). Then, from the fitted polynomial, the slope parameter $a_{i\times j}$ is retrieved and placed in the same pixel (i, j) of the parametric image. Finally, the CBEUS image is generated after normalizing the pixel values into the interval [0, 255]. Figure 3.1 illustrates this process.

For clarification purposes, three examples of actual values in the experiment with the digital phantom are presented. Pixel locations from each of the different tissue type regions were chosen. For the muscle tissue region, the array [65, 59, 56, 52, 49, 49, 48, 44, 40] was retrieved. By fitting the first-order polynomial, the model $\hat{y} = -2.73x + 62.27$ was obtained, where the slope parameter is -2.73. Next, for the lipid tissue region, the array [87, 87, 90, 93, 96, 96, 93, 101, 93] was retrieved. Similarly, the model $\hat{y} = 1.25x + 87.89$ is obtained, with the slope parameter as 1.25. Finally, for the water medium region, the external region where the temperature images contain black pixels was used. The retrieved array should be [0, 0, 0, 0, 0, 0, 0, 0, 0], and the model would be $\hat{y} = 0$, and the slope parameter is 0.

The slope parameters are placed in the corresponding pixel locations of the CBEUS image, and the normalization process is applied. In this sense, the lowest value in the matrix will be converted to 0 (black) and the highest value will be converted to 255 (white). As shown in Figure 3.1d, the muscle region is darker, which relates to the fact that this type of tissue presented a negative slope parameter, i.e. pixels get darker (pixel values get lower) as the temperature rises.

The modeling starts by considering the following practical scenario: the ultrasonic transducer is applied to the surface of some system, generating a base image that represents the system at a given temperature T_1. After some time, as the heating process occurs, the system temperature rises, and the generated image now represents the same system at a new given temperature T_2. In this practical scenario, there are two images and the desire to estimate the difference $\Delta = T_2 - T_1$.

Images are a type of unstructured data, and it is necessary to define a set of processes to extract information to produce features. Since images are a collection of pixels, and there is a direct relation between pixel $< i, j >$ of images T_1 and T_2 if, and only if, the transducer position was not altered during the process, it is possible to track how the pixel value was affected by the temperature variation.

Four different approaches for modeling this scenario were developed constructively. It starts by considering a collection containing M images that are ordered by the referenced temperature. Hence, this collection can be represented as a set:

$$C = \{T_1, T_2, ..., T_M\} \tag{3.2}$$

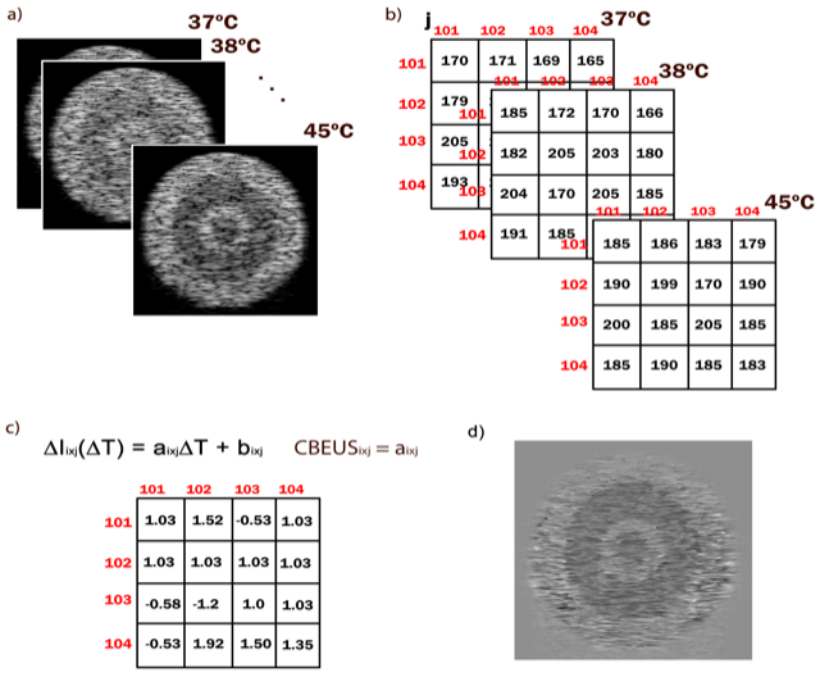

Figure 3.1: Process for creating a CBEUS image. a) set of images ordered by temperature level in ascending order. b) small section of the images with grayscale values of the pixels. c) the resulting slope parameters for each pixel in the parametric image. d) the final CBEUS image obtained after pixel value normalization.

Figure 3.2 illustrates the possible combinations that can be obtained using eight different images ordered by temperature. For the sake of the algorithm, M must be at least 2. In this sense, for a given collection C containing M images, a group G containing $M-1$ sets $\{S_1, S_2, ..., S_{M-1}\}$ can be generated so that:

- S_1 contains $N-1$ groups with 2 images and $\Delta = 1$
- S_2 contains $N-2$ groups with 3 images and $\Delta = 2$
- ...
- S_{M-1} contains 1 group with M images and $\Delta = M-1$

3.2 Proposed Approaches

The first idea was to select the same pixel value from the first (T_1) and last (T_2) images of each group and associate them with the expected Δ value, as depicted in Figure 3.3. But although the raw pixel value can be used, it was also proposed the usage of the CBEUS image information to incorporate the structure description capability of the parametric image, as shown in Figure 3.4, which was the actual

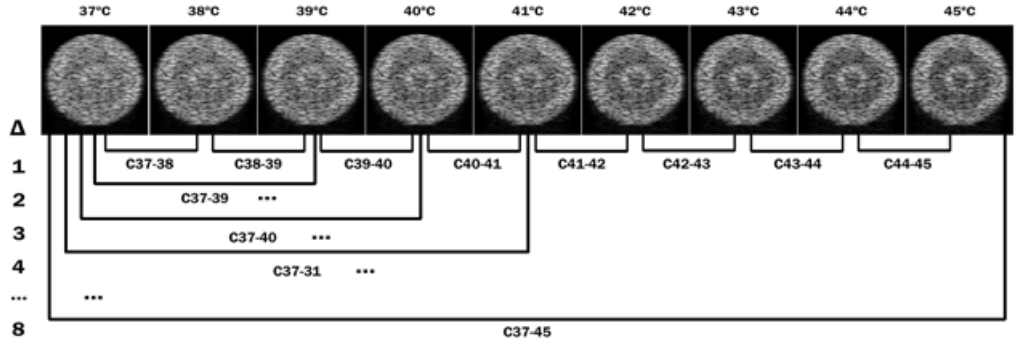

Figure 3.2: Illustration of the possible arrangements for generating CBEUS images from a set of B-mode images ordered by represented temperature.

motivation for creating the sets of images in the collection, as the computation of a first-order polynomial demand at least two points, and the more points, the better is the interpolation.

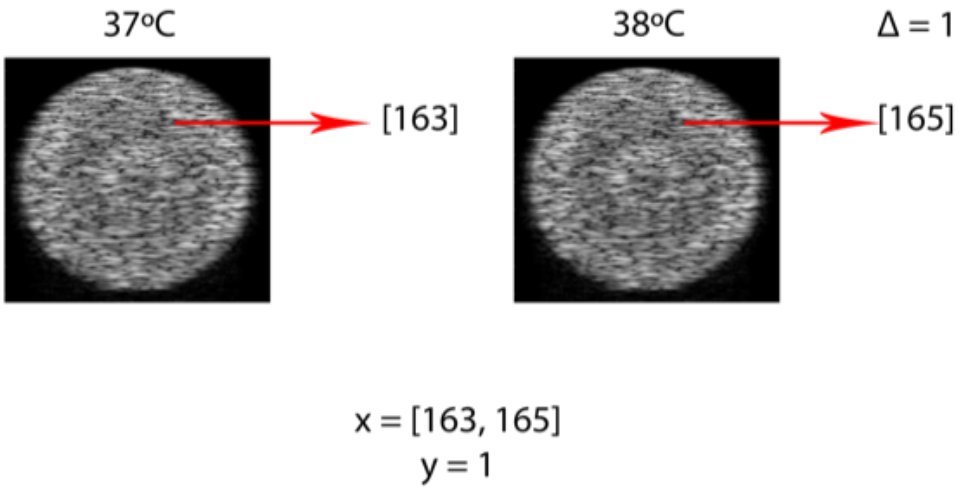

Figure 3.3: Pictorial representation of the first data modeling approach used. By considering two B-mode temperature images, the same pixel of both images is selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images.

By recalling collection C, for each base image, a set of CBEUS images with their respective Δ values were generated. The third and fourth approaches taken were to extract the grayscale values of both images within a certain neighborhood of size N, reshape it in the form of a vector, and associate it with the known temperature variation Δ. In the third approach - as shown in Figure 3.5 - the neighborhood was extracted only from the base image, while approach four (Figure 3.6) extracts the same neighborhood from both images. Since the thermal expansion of tissues may

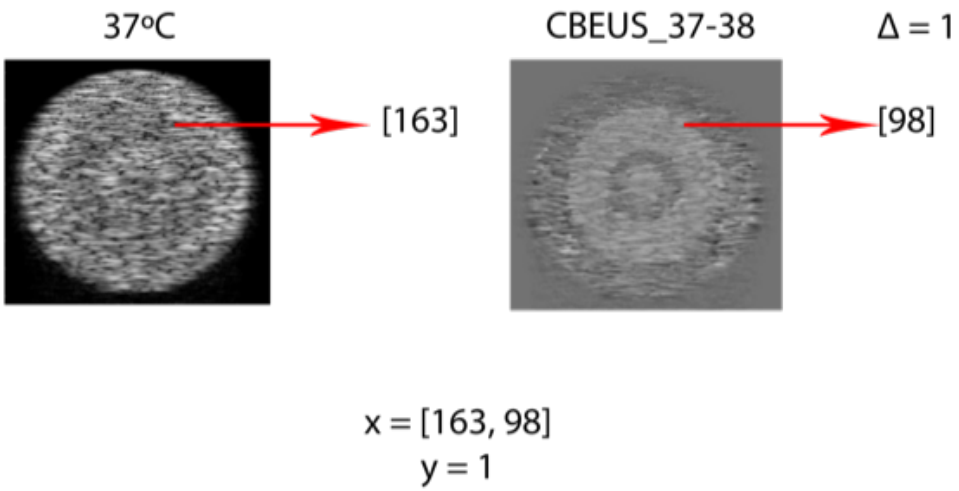

Figure 3.4: Pictorial representation of the second data modeling approach used. By considering a B-mode temperature image and the CBEUS image computed with a certain temperature variation, the same pixel of both images is selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images.

lead to a situation where each pixel does not represent the same site in the structure, this procedure is also convenient to account for any pixel displacement caused by temperature rise.

By iteratively performing this process through all viable pixels within all images, a dataset is generated. It must be stated that the dataset is unbalanced, as the sets within collection C present a different number of images.

In this sense, it can be stated that there is a group of observations with a set of known expected outputs and that there are new observations with a similar structure that require the estimation of this output, which is the definition of a supervised learning problem. Hence, it would be possible to use a machine learning algorithm to learn data patterns and build a predictor.

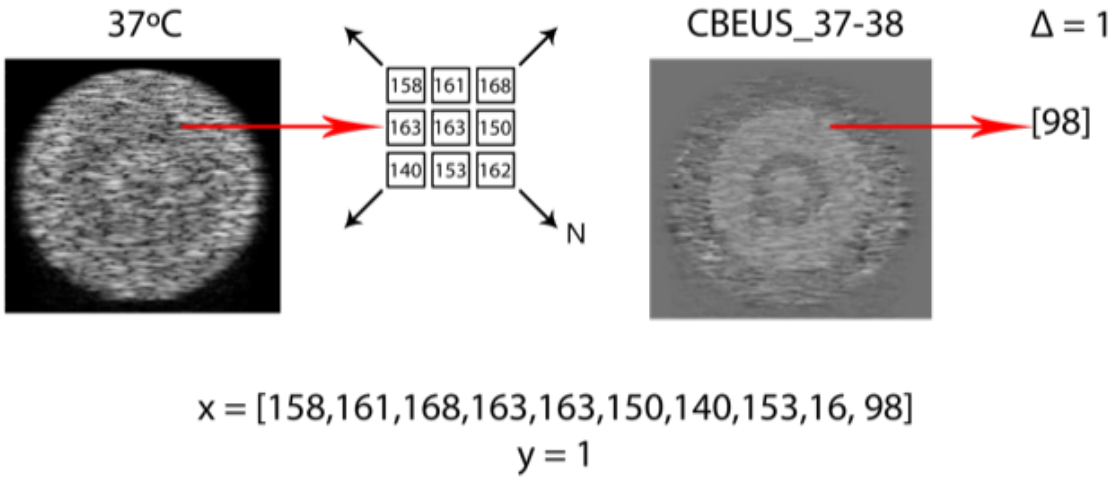

Figure 3.5: Pictorial representation of the third data modeling approach used. By considering a B-mode temperature image and the CBEUS image computed with a certain temperature variation, for the same pixel of both images, a neighborhood of size N for the temperature image and the single pixel are selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images. A valid pixel implies that the neighborhood of size N can be retrieved, i.e. it is within the image bounds.

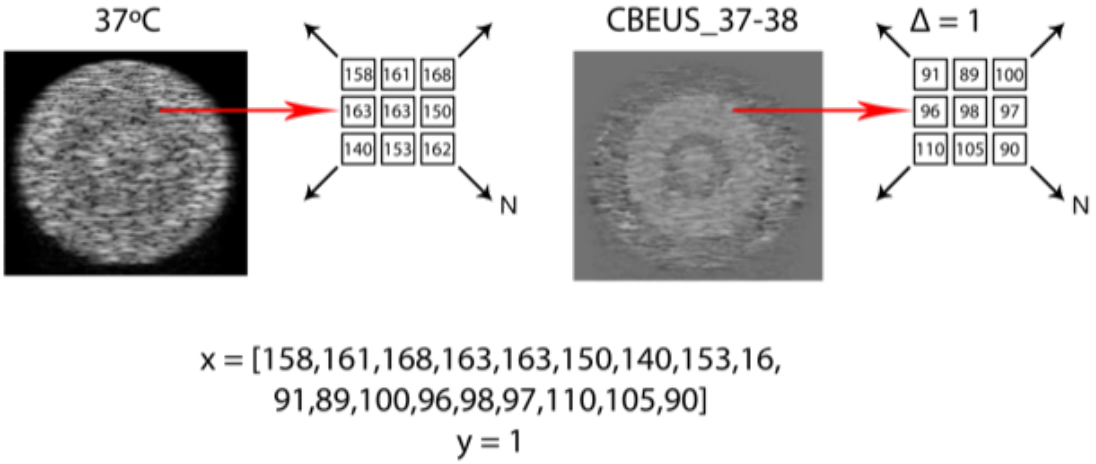

Figure 3.6: Pictorial representation of the fourth data modeling approach used. By considering a B-mode temperature image and the CBEUS image computed with a certain temperature variation, for the same pixel of both images, a neighborhood of size N is selected to generate an observation to the dataset. The expected output of the dataset is the given Δ. The process is repeated for all valid pixels in both images. A valid pixel implies that the neighborhood of size N can be retrieved, i.e. it is within the image bounds.

3.3 Machine Learning Models

After generating the dataset from the images, it is possible to feed the data to a supervised machine learning model. This section briefly describes two groups of models that were used in the experiments, while the following two chapters provide a deep overview of the weightless neural models that were developed during the first steps of this work. Nevertheless, it should be stated that a deeper exploration of other models' performances should be addressed in future projects.

3.3.1 Tree-based methods

Both the Random Forest (RF) and the Gradient Boosting Decision Trees (GBDT) are ensemble models derived from the classic Decision Tree (DT) [39]. A tree is a data structure formed by nodes that can have one or more child nodes. In a tree, there are three types of nodes: the root node, which is a single node that represents the starting element of the tree; internal nodes, which are nodes that have one or more child nodes and a father node; and leaf nodes, which are nodes without child nodes. Figure 3.7 illustrates this structure.

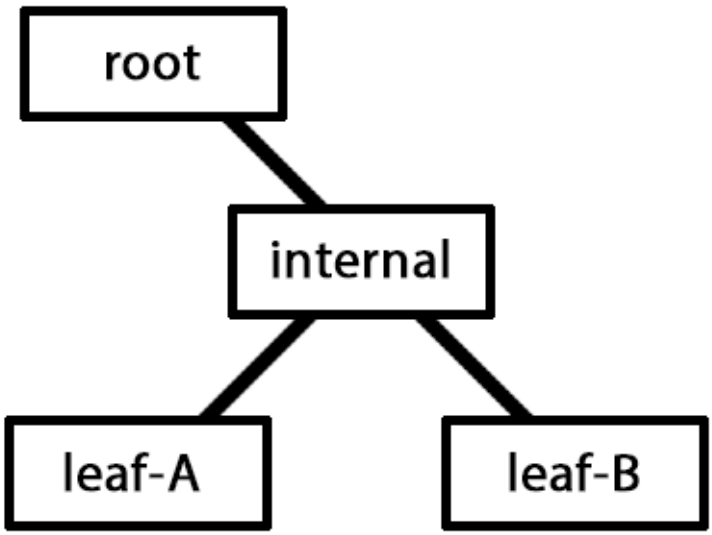

Figure 3.7: Classic tree data structure.

The training phase of a DT model builds a tree based on the dataset. Different algorithms for training can be used like C4.5 [40] and CART [41]. The usual approach is to use a metric to identify a feature that can be used to effectively create a partition of the data. Given the current node, this feature is used to create

an if/else statement, and child nodes are generated to address the corresponding decisions. Then, the algorithm repeats for each child node.

As previously mentioned, RF is an ensemble method. Ensemble methods are a special class of models that group simple models that work together and combine their responses to improve the overall performance. These simple models are known as weak learners. In the RF model (and also in the GBDT model), the weak learners are DTs.

GBDT [42], as implied by the model's name, uses a technique known as *boosting* [43]. This is one of the possible techniques used by ensemble models to direct how the dataset is approached by the *weak learners* and guides the training process. The idea is to add new DTs to the ensemble as the previous DTs are trained, and these new models are focused on correcting the errors generated by the previous learners. This process is illustrated in Figure 3.8 The process of minimizing the error within the new models is associated with the idea of the gradient descent algorithm, where we use a learning rate η to define the contribution of each new *estimator*. The smaller the learning rate, the less the next learner will adapt to previous errors.

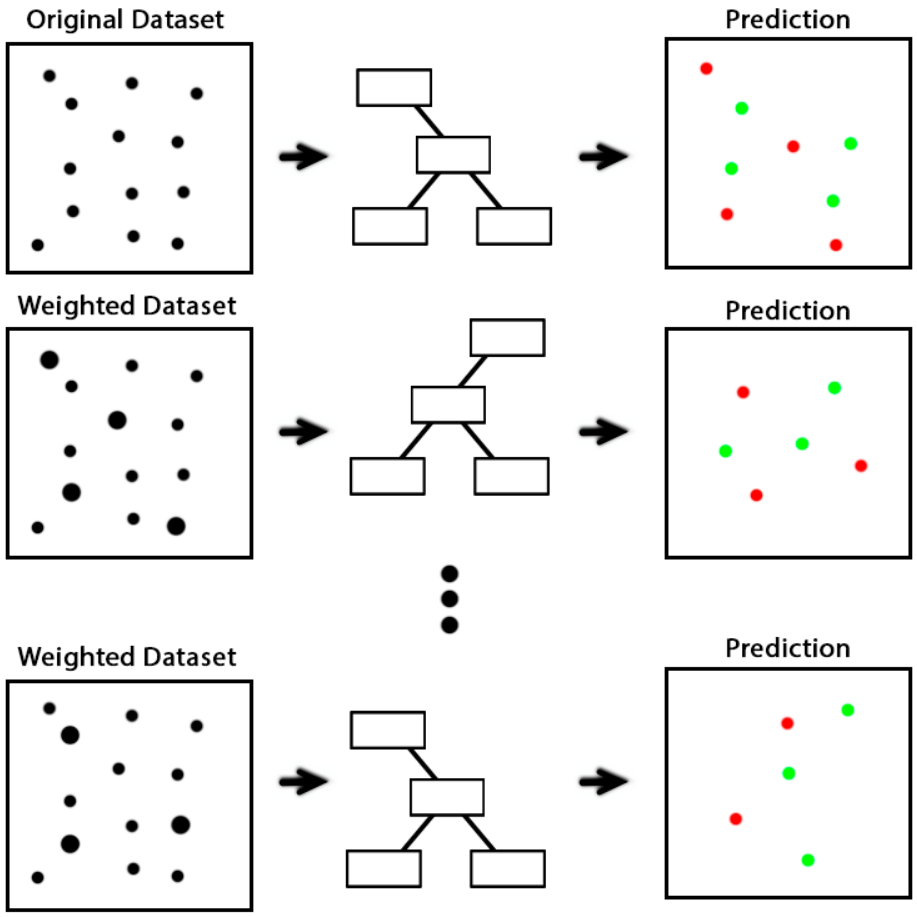

Figure 3.8: Examplification of the boosting technique.

3.3.2 Deep Neural Networks

Deep Learning stands as the current basis for modern artificial intelligence systems, as it has been applied to a wide range of domains reaching state-of-the-art results for applications like image recognition [44], voice generation [45] and speech recognition [46]. This comes from the ability of deep neural models to detect and learn non-linear patterns from data.

A neural network is a classic machine learning model that aims to mimic a living brain [47]. It is composed of several elements known as *neurons* that are units of computation with a series of weights associated with its inputs and an activation function that generates an output. In a neural network, a set of neurons is organized in the shape of layers. Typically, a Deep Neural Network (DNN) is classified as a neural network with at least two hidden layers. Neural networks learn from data through the use of the backpropagation algorithm, in which the data is presented to the network and, after a validation process, the weights are updated to better suit the predictions.

As the number of layers increases, the more powerful the network becomes. But this comes with the downside of demanding more time to adjust the weights. The capability of DNN's is better explored by using graphics processing units (GPU), which can perform numerous computations in parallel, allowing the distribution calculations involved in the network training process, accelerating this, which is its most computationally costly stage.

An important step in constructing a DNN's topology includes adjusting the regularization and normalization mechanisms that will be used in it. In this sense, two important parameters are the Batch Normalization [48], which is the process of normalizing the inputs in the network layers so that the optimization process can be improved, and the Dropout Rate [49], which is the probability of a neuron to be deactivated during the training phase.

In general, DNN models present a series of requirements to provide good performance, which includes a vast number of training examples to be able to detect non-linear patterns between the data and being able to generalize, and also a robust hardware infrastructure that makes it possible to accelerate the model training process, which becomes more expensive as the complexity of the network also increases. Based on these characteristics, part of the proposal of this work consisted of the elaboration and exploration of two weightless neural network models, given that they are a class of models that generalize well even with few examples and their main characteristic is an efficient training process with little expenditure of resources.

3.3.3 RegressionWiSARD

In contrast to the classic perceptron model [50], weightless neural models do not contain weight parameters to be adjusted. The *n-tuple* classifier presented in [51] stands as a reference model from which other weightless neural systems were developed. Initially proposed for handwritten character recognition, this method receives a binary input divided into n subsets and addressed to node structures.

RegressionWiSARD (ReW) extends the original WiSARD model by applying the additional accumulative dimension of the n-Tuple Regression Network, so that its architecture is now based on RAM-memory locations with two dimensions: counter and *sum*, with both values being updated for each training pattern. Initially, all memory location values are set to zero. The ReW model keeps the binary input restriction and the pseudo-random mapping of the input, while it does not present a group of discriminators. Since the output it tries to predict is within a continuous space, only one set of RAM-nodes is needed.

Regarding the training phase, the ReW model is similar to WiSARD: a pseudo-random mapping is applied and each n-tuple is related to a specific memory address. However, the process of directing the observation to a discriminator is replaced by adding the expected output to the *sum* cell in each memory address that was activated by the input pattern. The training process is described by figure 3.9. The classification phase also applies random mapping, while the RAM activation is now performed by collecting all *sum* values of the addresses along with all counter values, followed by an average computation. The n-Tuple Regression Network proposed the usage of a simple mean computation. The ReW model extends this computation by allowing the model to perform different types of average computations, described in the following list:

- simple mean: $\frac{1}{n}\sum_{i=0}^{N}\frac{sum_i}{c_i}$;
- power mean: $(\frac{1}{n}\sum_{k=1}^{n}(y_k/c_k)^p)^{1/p}$; where p is the power
- median: central value of $\frac{y_i}{c_i}$, with i in range [0, n]
- harmonic power mean: $\frac{n}{\sum_{i=0}^{n}\frac{c_i}{y_i}}$
- geometric mean: $(\prod_{i=0}^{n}\frac{y_i}{c_i})^{\frac{1}{n}}$
- exponential mean: $log(\frac{\sum_{i=0}^{n}e^{\frac{y_i}{c_i}}}{n})$

One aspect to be noticed is that when during the prediction phase a memory location that was never trained is accessed, ReW will respond as a *do not know* answer prediction. This answer must be adapted according to the problem domain. For the sake of the current work, this prediction is treated as 0.

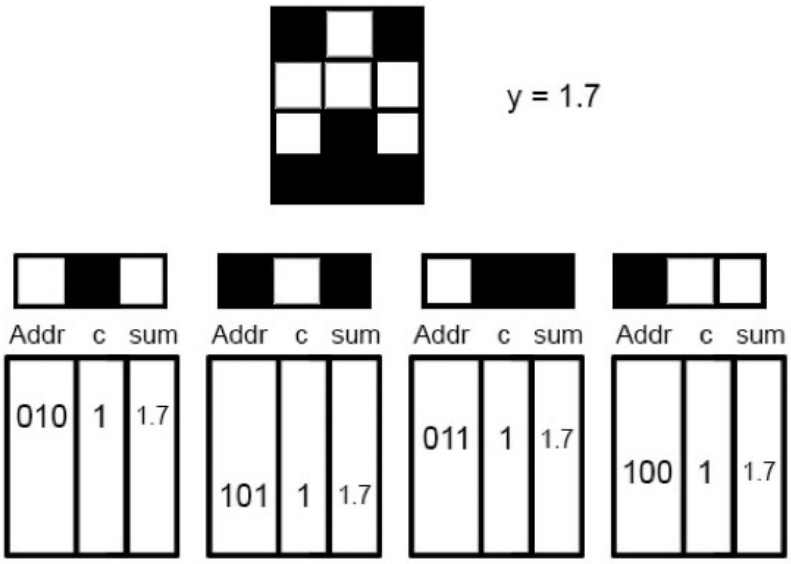

Figure 3.9: Description of the training process of the RegressionWiSARD model. A binary pattern is presented alongside its expected output. The pattern is divided into n tuples and each tuple is addressed to a RAM node. The address formed by the tuple increments the counter by 1 and the *sum* by the expected output.

3.3.4 ClusRegressionWiSARD

Inspired by ClusWiSARD, the ClusRegressionWiSARD (CReW) model is a network formed by a set of ReWs, each one with distinct pseudo-random mappings but with the same retina and the same address size. Since ReW RAM-like nodes operate as only one discriminator, CReW starts as a single ReW model. Then, in the training phase, the input pattern is presented to this ReW, and a score value that corresponds to the number of memory addresses that were accessed and contained a counter value greater than a value k is returned. If this score is greater than a predefined threshold, the pattern is learned. Otherwise, a new ReW model is created. For each of the following observations, the input pattern is presented to all ReWs, and the one that returns the highest response is selected for learning the input. This process is illustrated in Figure 3.10. As for the prediction phase, each ReW will classify the input submitted to the CReW model, and the highest score will perform the prediction. If there is a tie between two or more ReWs, the bleaching process is applied.

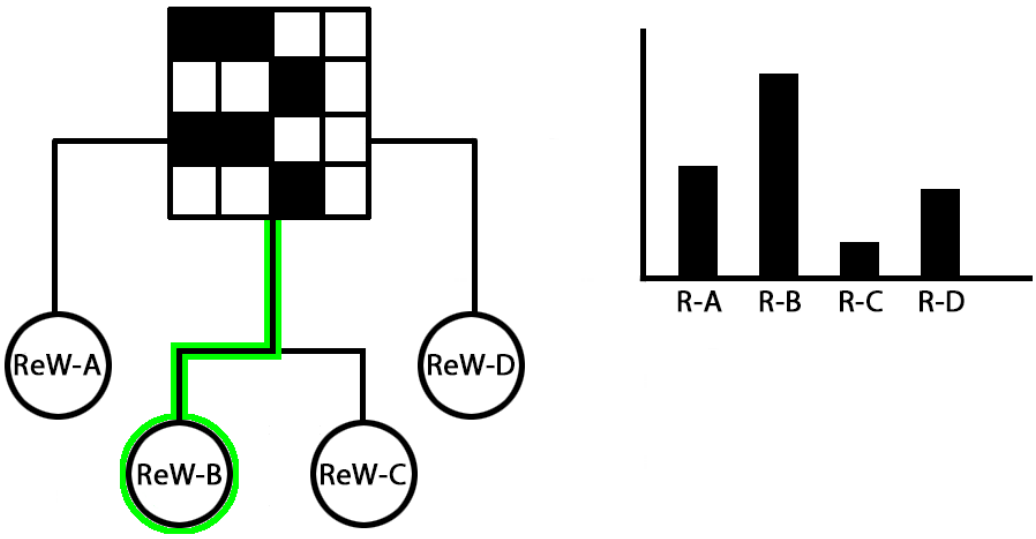

Figure 3.10: Example representation of the CReW model process to select a ReW model for training.

Chapter 4

Computational Results

This chapter presents the computational results of the experiments. For each model, a train/test split of the dataset was performed, with 30% of the data reserved for model testing, which comprises the reported results. The process was repeated 10 times and the average of the Mean Absolute Error (MAE) and Mean Squared Error (MSE) metrics were measured. The standard deviation is associated with each value. For both metrics, the closer the reported value is to 0.0, the better. Also, the time taken for each model to finish the training and prediction processes was measured, and the values were displayed in seconds (s).

4.1 Experimental Framework

This section presents the main elements relating to the experiments. Initially, the sources of ultrasound images that were used are detailed, in addition to providing more details on the final format of the datasets created using each proposed approach. Next, details are provided about the parameters used by the models. Finally, the numerical results of the experiments carried out are presented.

4.1.1 Details on Simulation Data

In the experiments, a digital phantom was used as shown in Figure 4.1, which consists of a set of concentric rings, where the inner circle and the second ring (white pixels) simulate muscle tissue, and the first and third rings simulate lipid tissue (black pixels). This structure is immersed in an aqueous medium (gray region). The phantom's dimension is 20mm x 20mm (height x thickness).

Figure 4.2 illustrates the generated B-mode images that represent different temperatures in the range of 37-45ºC. The authors of [38] generated and provided both the phantom and the temperature images of this simulation using the k-Wave open-source MATLAB toolbox [72].

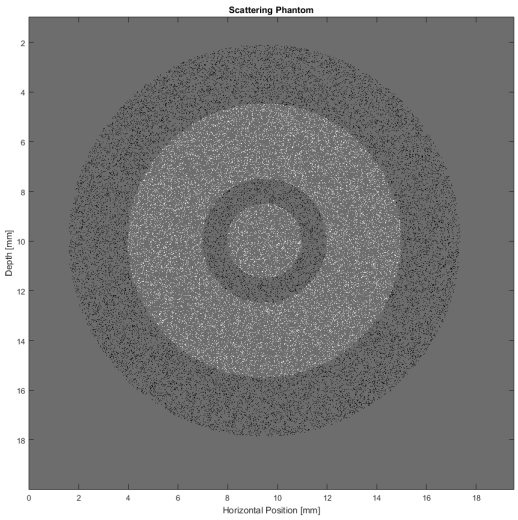

Figure 4.1: Digital phantom used for B-mode image generation. White pixels simulate muscle tissue, black pixels simulate lipid tissue (fat), and the gray region simulates water. This figure is based on the one presented in [38].

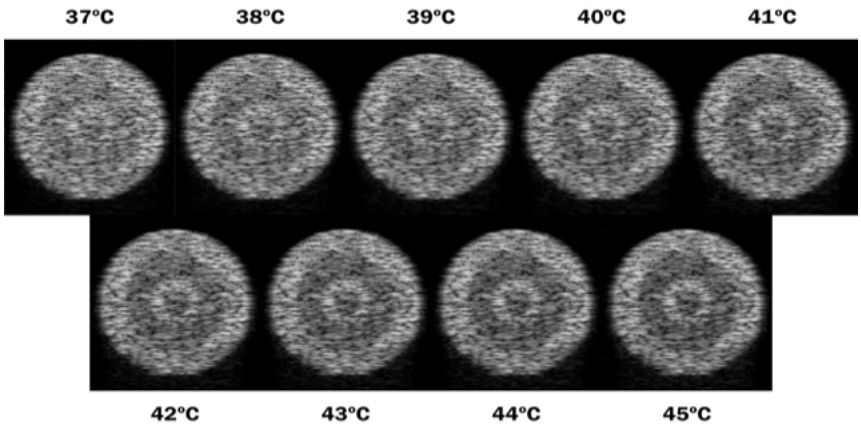

Figure 4.2: Collection of the resulting B-mode images representing different temperatures obtained using the numeric phantom. In each figure, the related temperature is the same across the whole image.

Table 4.1 shows the configuration of the dataset generated using the simulation data in each of the proposed approaches. In each line, the number of input features of the observations, the total number of observations, and how the 70/30 partition was used to form the training and prediction sets are recorded. It is possible to observe the increase in the dimensionality of the input data as the approach used becomes more complex. A second factor to note is the gradual decrease in the number of observations. This occurs because, as the value of the neighborhood N increases, the pixels on the edge of the image begin to become unfeasible options

as it is not possible to capture a sufficiently large neighborhood without reaching a region outside the image domains.

		Simulation		
Approach	Features	Total	Training	Prediction
1	2	3,510,000	2,457,000	1,053,000
2	2	3,510,000	2,457,000	1,053,000
3 (N=1)	10	3,465,144	2,425,600	1,039,544
3 (N=3)	50	3,376,296	2,363,407	1,012,889
3 (N=5)	122	3,288,600	2,302,020	986,580
4 (N=1)	18	3,465,144	2,425,600	1,039,544
4 (N=3)	98	3,376,296	2,363,407	1,012,889
4 (N=5)	242	3,288,600	2,302,020	986,580

Table 4.1: Information regarding the generation of datasets using each of the proposed approaches based on simulation data. The table shows, for each approach, the number of input features, the total size of the dataset, and the size of each partition after dividing the training set and the prediction set.

4.1.2 Details on $In - Vitro$ Data

The experiments were also performed using B-mode ultrasound images generated from *in vitro* samples of porcine muscle tissue used in [73]. In this work, the authors aimed to analyze the required dimensions of the region of interest within an image where the contrast generated by the CBEUS image can be visualized by an observer.

The sample was placed in a heat bath in a PVC chamber with water heated by a copper tube. An Ultrasonix SonixMDP scanner with an Ultrasonix L14-5/38 linear transducer was used for generating the B-mode images, while the video capture was performed using the CamStudio software.

First, the sample was maintained at a temperature of 36^{o}C for 30 minutes to achieve thermal equilibrium. Then, a 30fps video of 5 seconds was recorded. Next, the authors increased the system temperature by 1^{o}C, maintained the sample at this configuration for 5 minutes, and recorded another 5-second video. The process was repeated until it reached the final temperature of 45^{o}C.

Figure 4.3 illustrates B-mode images generated from this sample. Similarly, in the simulation case, a collection of images representing different temperatures in a certain range was used, but in this scenario, we start with a base image at 36^{o}C up to 45^{o}C. It is worth mentioning that the B-mode images for each temperature are similar to each other and one cannot detect temperature differences by visual inspection.

Similarly to the previous section, table 4.2 shows information about the data partitions using the *in-vitro* sample. The expected behavior in this situation is the

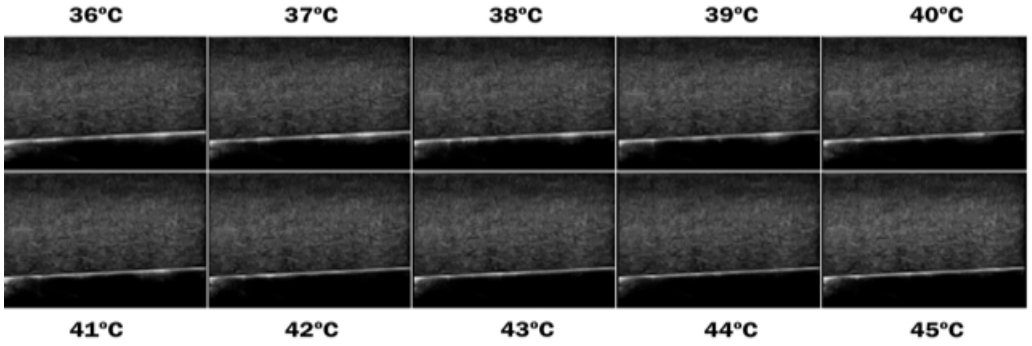

Figure 4.3: Resulting B-mode images representing temperatures of 36ºC and 45ºC obtained using *in vitro* porcine skeletal striated muscle. In each figure, the related temperature is the same across the whole image. We use different images for all integer temperature values between the ones displayed in the figure, but they do not present significant differences visually and were omitted for simplification purposes.

same as the previous one: greater dimensionality of the input data and a smaller number of features as the approach used becomes more complex.

		In-vitro		
Approach	Features	Total	Training	Prediction
1	2	6,075,000	4,252,500	1,822,500
2	2	6,075,000	4,252,500	1,822,500
3 (N=1)	10	6,007,680	4,205,376	1,802,304
3 (N=3)	50	5,874,120	4,111,884	1,762,236
3 (N=5)	122	5,742,000	4,019,400	1,722,600
4 (N=1)	18	6,007,680	4,205,376	1,802,304
4 (N=3)	98	5,874,120	4,111,884	1,762,236
4 (N=5)	242	5,742,000	4,019,400	1,722,600

Table 4.2: Information regarding the generation of datasets using each of the proposed approaches based on real data. The table shows, for each approach, the number of input features, the total size of the dataset, and the size of each partition after dividing the training set and the prediction set.

4.1.3 Computational Environment

The experiments were developed using the Python3 programming language and the numpy, sklearn [74], xgboost, TensorFlow modules. All experiments were performed using a system consisting of an Intel Xeon 2.3GHz processor, 60GB of RAM, NVidia T4 GPU, and Debian GNU Linux version 11.

4.2 Numerical Results

Table 4.3 shows the hyperparameters and configurations used in each of the models. These parameter values were obtained empirically, where a finite set of different values was tested, and these configurations returned the best results so far. In the final section, it is discussed the necessity of further hyperparameter exploration.

Another important aspect of the DNN model refers to other parameter configurations that would not be well explained in a tabular format. In the experiments, considerably simple topologies were used. In all cases, the initial weights of the networks were initialized randomly. Combinations between two, three, and four fully connected layers were tested, where the number of neurons present in each layer was varied. Among the tests carried out, alternatives were tested using combinations of the number of neurons within the set [1024, 512, 256, 128] where either all layers had the same number of neurons or there was a decreasing sequence. In all cases, as mentioned previously, the networks had only one output layer, as the problem was treated as a regression problem. The activation function used in all layers was RelU, and the dropout and bath normalization techniques were applied to all layers. Finally, the topology that presented the best results had three dense layers with 1024 neurons each.

Model	**Parameter**	**Explored Values**	**Best Value**
RF	Max depth	3, 6, 10	6
	#estimators	10, 100, 1000	1000
GBDT	η	0.1, 0.3, 0.5	0.3
	Max depth	3, 6, 10	6
	#estimators	10, 100, 1000	1000
DNN	Dropout rate	10%, 30%	30%
	Batch normalization	Yes, No	Yes
	Epochs	10, 20, 100	100
ReW	Thermometer	5, 10, 20	10
	Address size	2, 5, 10, 20, 30	10
	Mean	Simple, Median, Exponential	Median
CReW	Thermometer	5, 10, 20	10
	Address size	2, 5, 10, 20, 30	10
	Mean	Simple, Median, Exponential	Median
	Thershold	0.5, 0.7, 0.9	0.7

Table 4.3: Parameters of the models Random Forest (RF), Gradient-boosting Decision Tree (GBDT), Deep Neural Network (DNN), RegressionWiSARD (ReW) and ClusRegressionWiSARD (CReW) that returned the best results across the computational experiments. The values were obtained empirically after exploring a finite set of different configurations.

4.3 Results on Simulation Data

This section presents the experimental results of the experiments applied to the dataset generated from the digital phantom. Tables **??-??** summarize the numerical results. Figures 4.4-4.9 illustrate the behavior of these results by comparing each metric with each one of the proposed approaches.

	MAE (ºC)	MSE (ºC)
RF	1.67 ± 0.00e+00	3.89 ± 8.00e-03
XGB	1.66 ± 1.17e-02	3.89 ± 3.00e-03
XGB-GPU	1.66 ± 2.22e-16	3.89 ± 6.63e-03
DNN	1.87 ± 3.00e-03	4.36 ± 4.00e-03
REW	1.67 ± 4.00e-03	4.36 ± 4.00e-03
CREW	1.67 ± 3.00e-03	3.89 ± 1.14e-02

Table 4.4: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #1.

	Training Time (s)	Prediction Time (s)
RF	141.30 ± 2.15e+00	3.93 ± 1.68e-01
XGB	17.52 ± 4.67e-01	2.95 ± 5.19e-02
XGB-GPU	18.66 ± 3.81e-01	0.37 ± 4.00e-03
DNN	452.02 ± 9.68e+00	60.65 ± 2.26e+00
REW	0.30 ± 5.55e-17	0.34 ± 5.55e-17
CREW	2.83 ± 1.47e-02	1.74 ± 1.22e-02

Table 4.5: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #1.

	MAE (ºC)	MSE (ºC)
RF	1.39 ± 6.32e-03	2.94 ± 6.32e-03
XGB	1.18 ± 0.00e+00	2.28 ± 9.17e-03
XGB-GPU	1.18 ± 0.00e+00	2.28 ± 4.44e-16
DNN	1.48 ± 3.00e-03	3.58 ± 8.31e-03
REW	1.68 ± 1.14e-02	3.89 ± 7.75e-03
CREW	1.68 ± 2.22e-16	3.92 ± 9.17e-03

Table 4.6: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #2.

	Training Time (s)	Prediction Time (s)
RF	214.60 ± 2.63e+00	3.85 ± 2.32e-01
XGB	20.09 ± 7.03e-01	3.05 ± 3.30e-02
XGB-GPU	19.63 ± 3.54e-01	0.39 ± 4.58e-03
DNN	448.23 ± 8.80e+00	59.00 ± 2.66e+00
REW	0.30 ± 7.00e-03	0.32 ± 5.39e-03
CREW	2.89 ± 2.53e-02	1.82 ± 1.80e-02

Table 4.7: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #2.

	MAE (ºC)	MSE (ºC)
RF	1.39 ± 7.48e-03	2.92 ± 6.00e-03
XGB	1.15 ± 7.00e-03	2.23 ± 4.47e-03
XGB-GPU	1.15 ± 4.58e-03	2.25 ± 0.00e+00
DNN	1.63 ± 2.22e-16	4.99 ± 5.39e-03
REW	1.67 ± 3.00e-03	3.89 ± 6.00e-03
CREW	1.67 ± 1.19e-02	3.89 ± 8.31e-03

Table 4.8: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 1.

	Training Time (s)	Prediction Time (s)
RF	691.00 ± 3.24e+00	3.84 ± 1.30e-01
XGB	21.79 ± 7.52e-01	2.95 ± 3.54e-02
XGB-GPU	22.92 ± 5.35e-01	0.45 ± 5.39e-03
DNN	387.91 ± 2.39e+01	61.16 ± 1.99e+00
REW	1.22 ± 3.38e-02	1.69 ± 2.80e-02
CREW	13.93 ± 4.26e-01	9.09 ± 3.03e-01

Table 4.9: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 1.

	MAE (ºC)	MSE (ºC)
RF	1.38 ± 2.22e-16	2.89 ± 6.00e-03
XGB	1.11 ± 8.94e-03	2.10 ± 9.17e-03
XGB-GPU	1.12 ± 2.22e-16	2.12 ± 0.00e+00
DNN	1.10 ± 0.00e+00	6.83 ± 5.39e-03
REW	1.67 ± 8.00e-03	3.89 ± 7.00e-03
CREW	1.67 ± 8.31e-03	3.90 ± 8.31e-03

Table 4.10: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 3.

	Training Time (s)	Prediction Time (s)
RF	355.52 ± 2.15e+00	4.47 ± 1.56e-01
XGB	52.59 ± 8.99e-01	3.15 ± 6.86e-02
XGB-GPU	38.18 ± 9.19e-01	1.22 ± 1.74e-02
DNN	390.86 ± 3.79e+01	60.89 ± 1.71e+00
REW	7.22 ± 1.05e-01	13.16 ± 4.29e-01
CREW	75.11 ± 1.29e+00	72.06 ± 1.32e+00

Table 4.11: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 3.

	MAE (ºC)	MSE (ºC)
RF	1.37 ± 3.00e-03	2.82 ± 5.39e-03
XGB	1.09 ± 0.00e+00	2.03 ± 8.94e-03
XGB-GPU	1.09 ± 0.00e+00	2.03 ± 8.31e-03
DNN	1.78 ± 7.00e-03	4.62 ± 8.88e-16
REW	1.67 ± 4.00e-03	3.88 ± 4.44e-16
CREW	1.66 ± 6.00e-03	3.89 ± 5.39e-03

Table 4.12: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 5.

	Training Time (s)	Prediction Time (s)
RF	7293.00 ± 2.12e+00	7.36 ± 2.25e-01
XGB	118.89 ± 3.03e+00	4.25 ± 5.02e-02
XGB-GPU	67.33 ± 1.08e+00	3.29 ± 2.76e-02
DNN	439.31 ± 2.19e+01	60.10 ± 1.80e+00
REW	8.27 ± 5.12e-02	15.21 ± 2.66e-01
CREW	83.68 ± 1.50e+00	74.00 ± 8.71e-01

Table 4.13: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #3 and neighborhood size N = 5.

	MAE (ºC)	MSE (ºC)
RF	1.36 ± 3.00e-03	2.80 ± 6.32e-03
XGB	0.74 ± 4.00e-03	1.16 ± 1.00e-02
XGB-GPU	0.74 ± 0.00e+00	1.16 ± 0.00e+00
DNN	3.18 ± 0.00e+00	7.73 ± 8.88e-16
REW	1.67 ± 6.63e-03	3.88 ± 4.44e-16
CREW	1.67 ± 3.00e-03	3.89 ± 4.47e-03

Table 4.14: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 1.

	Training Time (s)	Prediction Time (s)
RF	1522.34 ± 1.81e+00	3.90 ± 1.60e-01
XGB	29.72 ± 7.05e-01	2.99 ± 2.02e-02
XGB-GPU	27.98 ± 6.79e-01	0.53 ± 4.58e-03
DNN	457.15 ± 2.55e+01	55.35 ± 2.54e+00
REW	2.21 ± 1.64e-02	3.29 ± 1.83e-02
CREW	25.84 ± 7.41e-01	19.82 ± 3.75e-01

Table 4.15: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 1.

	MAE (ºC)	MSE (ºC)
RF	1.29 ± 2.22e-16	2.87 ± 4.00e-03
XGB	0.62 ± 6.40e-03	0.85 ± 0.00e+00
XGB-GPU	0.62 ± 8.72e-03	0.85 ± 4.00e-03
DNN	1.06 ± 3.00e-03	2.01 ± 4.47e-03
REW	1.67 ± 5.39e-03	3.90 ± 6.00e-03
CREW	1.67 ± 9.43e-03	3.90 ± 0.00e+00

Table 4.16: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 3.

	Training Time (s)	Prediction Time (s)
RF	8901.84 ± 2.46e+00	5.78 ± 1.14e-01
XGB	102.29 ± 4.23e+00	3.14 ± 5.33e-02
XGB-GPU	61.06 ± 1.30e+00	4.72 ± 6.24e-02
DNN	434.37 ± 2.57e+01	58.90 ± 1.70e+00
REW	13.89 ± 5.55e-02	25.84 ± 1.58e-01
CREW	138.62 ± 8.04e+00	113.25 ± 8.02e+00

Table 4.17: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 3.

	MAE (ºC)	MSE (ºC)
RF	1.25 ± 0.00e+00	2.53 ± 7.00e-03
XGB	0.58 ± 0.00e+00	0.73 ± 4.47e-03
XGB-GPU	0.58 ± 8.31e-03	0.73 ± 7.48e-03
DNN	1.03 ± 3.00e-03	3.39 ± 6.00e-03
REW	1.67 ± 1.00e-02	3.89 ± 1.17e-02
CREW	1.67 ± 5.39e-03	3.90 ± 0.00e+00

Table 4.18: Computational results regarding MAE and MSE for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 5.

	Training Time (s)	Prediction Time (s)
RF	13110.40 ± 2.51e+00	8.00 ± 1.08e-01
XGB	259.19 ± 1.22e+01	3.35 ± 6.48e-02
XGB-GPU	32.52 ± 6.29e-01	1.36 ± 1.62e-02
DNN	442.21 ± 1.43e+01	60.17 ± 1.17e+00
REW	16.00 ± 1.01e-01	31.70 ± 1.85e-01
CREW	167.87 ± 1.23e+01	123.27 ± 6.08e+00

Table 4.19: Computational results regarding training and prediction time for estimating temperature variation on simulation data using approach #4 and neighborhood size N = 5.

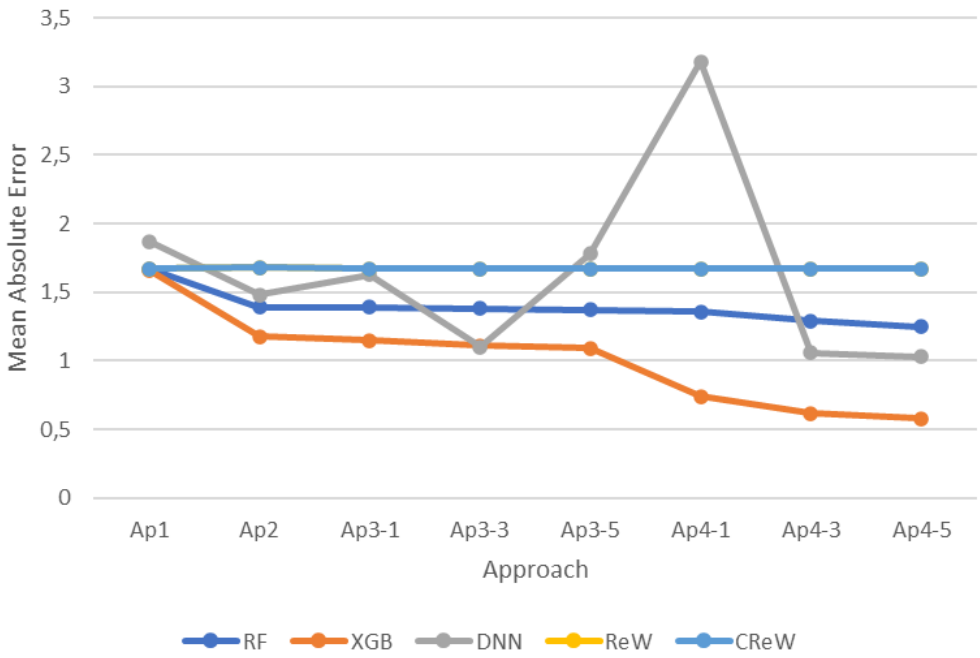

Figure 4.4: Comparison of the experimental results considering how the MAE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MAE for the 30% reserved data.

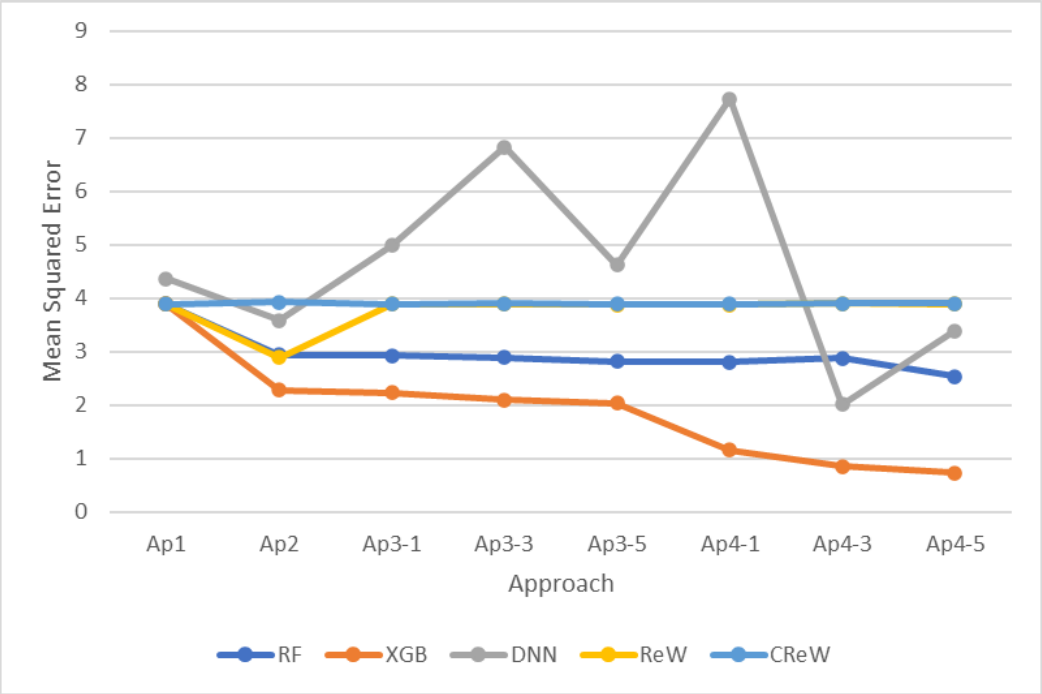

Figure 4.5: Comparison of the experimental results considering how the MSE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MSE for the 30% reserved data.

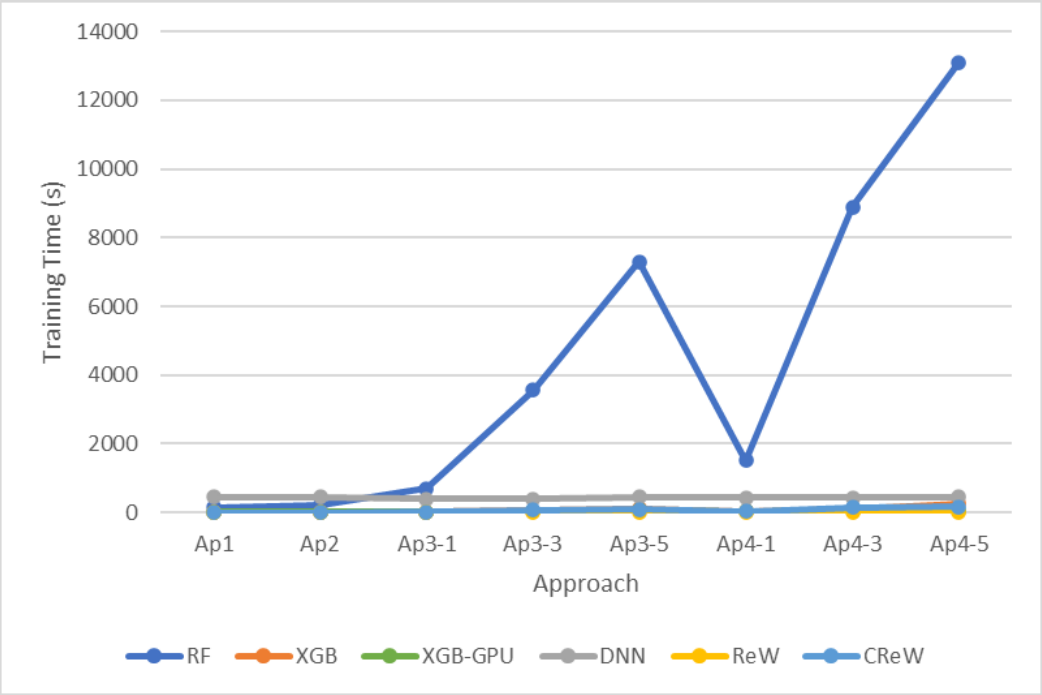

Figure 4.6: Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data.

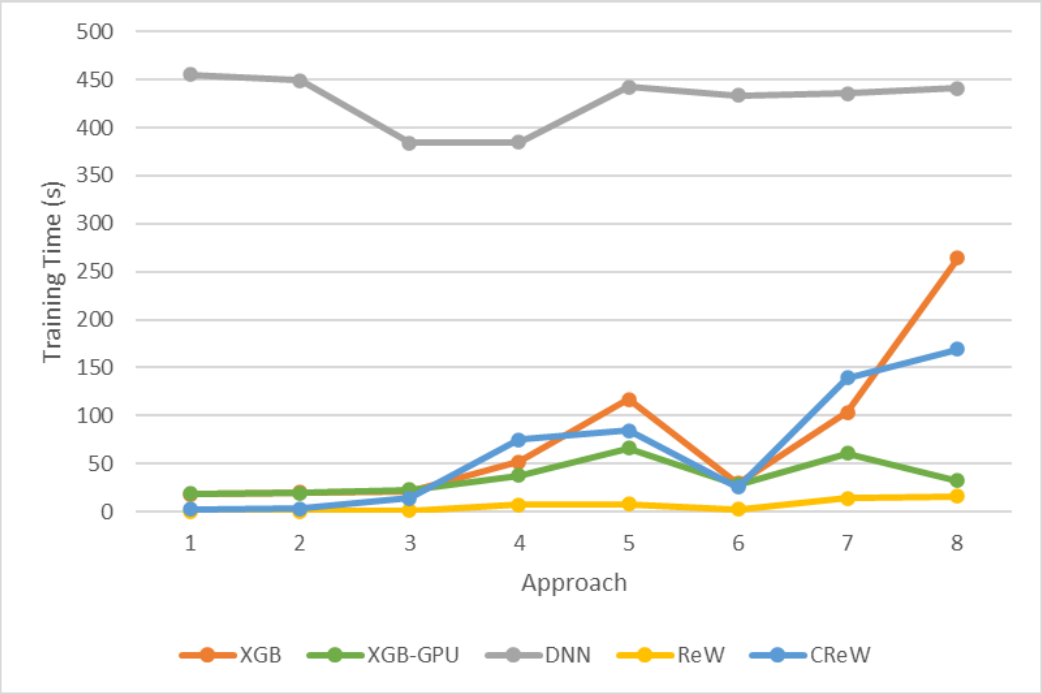

Figure 4.7: Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data. In this figure, the RF model was suppressed to evaluate the other models better.

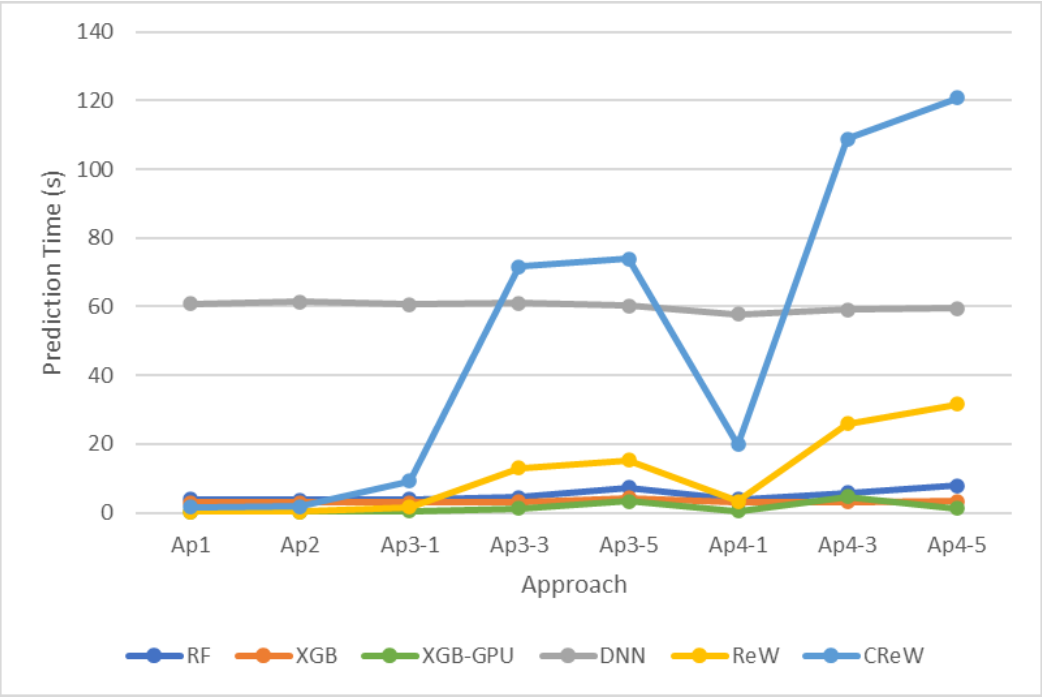

Figure 4.8: Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data.

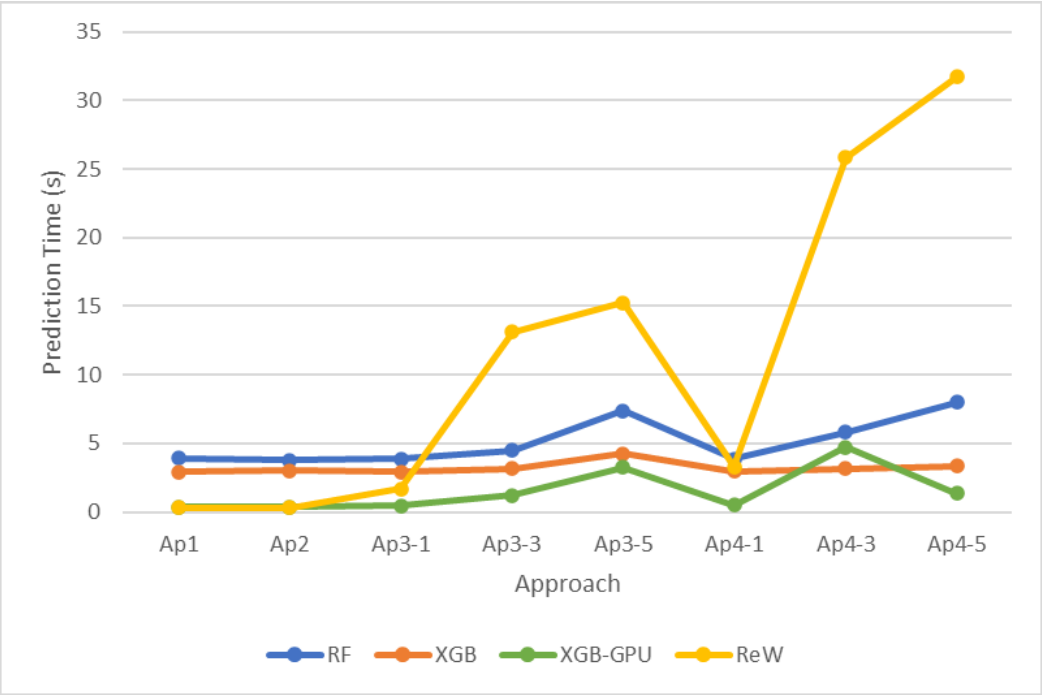

Figure 4.9: Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data. In this figure, the DNN and CReW models were suppressed to evaluate the other models better.

4.4 Results on *In-Vitro* Data

In this section, the same structure is performed to show the experimental results when the proposed modeling was applied to the images generated from the $in-vitro$ porcine sample. Tables 4.20-4.35 show the numerical results of the experiments, while figures 4.10-4.15 illustrate how these numerical values modified along each approach change.

	MAE (^{o}C)	MSE (^{o}C)
RF	1.87 ± 3.00e-03	4.88 ± 6.32e-03
XGB	1.66 ± 2.22e-16	3.89 ± 1.17e-02
XGB-GPU	1.66 ± 7.48e-03	3.89 ± 7.00e-03
DNN	1.87 ± 4.47e-03	4.36 ± 6.00e-03
REW	1.87 ± 7.00e-03	4.89 ± 0.00e+00
CREW	1.67 ± 1.00e-02	3.89 ± 4.44e-16

Table 4.20: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #1.

	Training Time (s)	Prediction Time (s)
RF	242.31 ± 2.16e+00	4.99 ± 1.56e-01
XGB	17.99 ± 3.98e-01	2.94 ± 1.62e-02
XGB-GPU	18.81 ± 5.77e-01	0.37 ± 5.39e-03
DNN	449.20 ± 7.84e+00	60.20 ± 2.97e+00
REW	0.46 ± 6.32e-03	0.53 ± 6.32e-03
CREW	2.84 ± 1.05e-01	1.76 ± 1.35e-01

Table 4.21: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #1.

	MAE (ºC)	MSE (ºC)
RF	1.29 ± 8.00e-03	2.74 ± 5.39e-03
XGB	1.11 ± 1.04e-02	2.24 ± 7.48e-03
XGB-GPU	1.11 ± 4.90e-03	2.24 ± 3.00e-03
DNN	1.78 ± 4.58e-03	5.68 ± 4.58e-03
REW	1.87 ± 2.22e-16	4.88 ± 6.32e-03
CREW	1.87 ± 7.00e-03	4.90 ± 0.00e+00

Table 4.22: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #2.

	Training Time (s)	Prediction Time (s)
RF	327.16 ± 3.78e+00	7.74 ± 7.38e-01
XGB	32.21 ± 7.58e-01	4.05 ± 8.32e-02
XGB-GPU	30.11 ± 4.56e-01	0.65 ± 6.32e-03
DNN	806.21 ± 1.30e+01	109.11 ± 2.42e+00
REW	0.47 ± 3.00e-03	0.51 ± 7.00e-03
CREW	4.96 ± 1.07e-01	2.76 ± 5.72e-02

Table 4.23: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #2.

	MAE (ºC)	MSE (ºC)
RF	1.30 ± 6.63e-03	2.73 ± 6.00e-03
XGB	1.09 ± 0.00e+00	2.16 ± 0.00e+00
XGB-GPU	1.09 ± 4.47e-03	2.16 ± 5.39e-03
DNN	1.62 ± 3.00e-03	5.15 ± 0.00e+00
REW	1.87 ± 3.00e-03	4.89 ± 0.00e+00
CREW	1.88 ± 5.00e-03	4.89 ± 4.00e-03

Table 4.24: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #3 and neighborhood size N = 1.

	Training Time (s)	Prediction Time (s)
RF	1359.87 ± 7.95e+00	7.64 ± 1.05e+00
XGB	40.09 ± 9.85e-01	4.00 ± 3.22e-02
XGB-GPU	37.82 ± 6.35e-01	0.75 ± 6.40e-03
DNN	814.04 ± 2.00e+01	108.55 ± 2.69e+00
REW	2.12 ± 4.41e-02	2.67 ± 6.22e-02
CREW	23.43 ± 1.14e-01	14.38 ± 4.43e-01

Table 4.25: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #3 and neighborhood size N = 1.

	MAE (ºC)	MSE (ºC)
RF	1.29 ± 6.00e-03	2.71 ± 5.39e-03
XGB	1.04 ± 7.00e-03	2.03 ± 3.00e-03
XGB-GPU	1.04 ± 2.22e-16	2.03 ± 3.00e-03
DNN	1.15 ± 4.00e-03	3.57 ± 1.22e-02
REW	1.87 ± 5.39e-03	4.88 ± 6.00e-03
CREW	1.87 ± 2.22e-16	4.89 ± 0.00e+00

Table 4.26: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #3 and neighborhood size N = 3.

	Training Time (s)	Prediction Time (s)
RF	5882.65 ± 4.25e+01	9.15 ± 5.15e-01
XGB	104.18 ± 7.60e-01	4.26 ± 7.48e-02
XGB-GPU	65.00 ± 6.20e-01	2.10 ± 1.70e-02
DNN	810.69 ± 2.39e+01	109.80 ± 2.51e+00
REW	11.53 ± 7.32e-02	21.07 ± 4.46e-01
CREW	119.92 ± 3.19e-02	106.71 ± 9.85e-02

Table 4.27: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #3 and neighborhood size N = 3.

	MAE (ºC)	MSE (ºC)
RF	1.28 ± 9.80e-03	2.70 ± 5.39e-03
XGB	1.06 ± 8.31e-03	2.03 ± 6.00e-03
XGB-GPU	1.06 ± 5.39e-03	2.04 ± 1.28e-02
DNN	1.67 ± 0.00e+00	8.16 ± 0.00e+00
REW	1.87 ± 2.22e-16	4.89 ± 6.40e-03
CREW	1.87 ± 8.72e-03	4.89 ± 6.00e-03

Table 4.28: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #3 and neighborhood size N = 5.

	Training Time (s)	Prediction Time (s)
RF	10761.28 ± 5.90e+01	12.90 ± 6.57e-01
XGB	209.63 ± 2.61e+00	15.03 ± 7.66e-02
XGB-GPU	111.92 ± 1.25e+00	7.09 ± 4.50e-02
DNN	795.98 ± 1.47e+01	105.55 ± 3.01e+00
REW	57.11 ± 9.38e-02	69.24 ± 3.23e-01
CREW	299.28 ± 1.56e-01	255.61 ± 1.11e-02

Table 4.29: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #3 and neighborhood size N = 5.

	MAE (ºC)	MSE (ºC)
RF	1.22 ± 5.39e-03	2.50 ± 0.00e+00
XGB	0.68 ± 8.00e-03	1.13 ± 5.39e-03
XGB-GPU	0.68 ± 1.11e-16	1.13 ± 2.22e-16
DNN	1.62 ± 4.00e-03	6.68 ± 1.02e-02
REW	1.87 ± 2.22e-16	4.90 ± 0.00e+00
CREW	1.87 ± 4.47e-03	4.90 ± 0.00e+00

Table 4.30: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #4 and neighborhood size N = 1.

	Training Time (s)	Prediction Time (s)
RF	2477.71 ± 6.93e+00	7.76 ± 5.18e-01
XGB	53.21 ± 8.26e-01	4.45 ± 2.29e-02
XGB-GPU	44.99 ± 5.97e-01	0.90 ± 3.00e-03
DNN	787.10 ± 3.62e+01	105.99 ± 3.50e+00
REW	3.81 ± 1.86e-02	4.98 ± 2.36e-02
CREW	41.48 ± 1.23e-01	26.20 ± 1.49e-02

Table 4.31: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #4 and neighborhood size N = 1.

	MAE (ºC)	MSE (ºC)
RF	1.16 ± 7.48e-03	2.29 ± 6.00e-03
XGB	0.57 ± 1.11e-16	0.86 ± 0.00e+00
XGB-GPU	0.57 ± 6.32e-03	0.86 ± 7.81e-03
DNN	1.02 ± 4.90e-03	3.99 ± 4.90e-03
REW	1.87 ± 5.39e-03	4.88 ± 4.58e-03
CREW	1.87 ± 3.00e-03	4.89 ± 7.48e-03

Table 4.32: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #4 and neighborhood size N = 3.

	Training Time (s)	Prediction Time (s)
RF	9871.54 ± 3.97e+01	12.93 ± 6.76e-01
XGB	177.26 ± 6.82e-01	12.45 ± 4.82e-02
XGB-GPU	104.55 ± 1.62e+00	7.47 ± 8.89e-02
DNN	785.92 ± 1.86e+01	106.75 ± 2.26e+00
REW	10.30 ± 7.34e-02	18.43 ± 1.39e-01
CREW	119.43 ± 7.91e+00	95.00 ± 7.01e+00

Table 4.33: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #4 and neighborhood size N = 3.

	MAE (ºC)	MSE (ºC)
RF	1.14 ± 2.22e-16	2.18 ± 3.00e-03
XGB	0.52 ± 4.00e-03	0.73 ± 3.00e-03
XGB-GPU	0.52 ± 1.11e-16	0.73 ± 6.32e-03
DNN	0.97 ± 8.31e-03	4.77 ± 7.00e-03
REW	1.87 ± 3.00e-03	4.89 ± 0.00e+00
CREW	1.87 ± 6.00e-03	4.89 ± 6.00e-03

Table 4.34: Computational results regarding MAE and MSE for estimating temperature variation on real data using approach #4 and neighborhood size N = 5.

	Training Time (s)	Prediction Time (s)
RF	15278.82 ± 1.62e+02	15.69 ± 1.25e+00
XGB	368.65 ± 7.33e+00	12.44 ± 6.61e-02
XGB-GPU	183.38 ± 6.37e-01	5.81 ± 2.09e-02
DNN	785.32 ± 2.48e+01	104.82 ± 1.43e+00
REW	59.35 ± 1.13e-01	68.38 ± 2.12e-01
CREW	269.87 ± 9.85e+00	203.25 ± 1.80e+01

Table 4.35: Computational results regarding training and prediction time for estimating temperature variation on real data using approach #4 and neighborhood size N = 5.

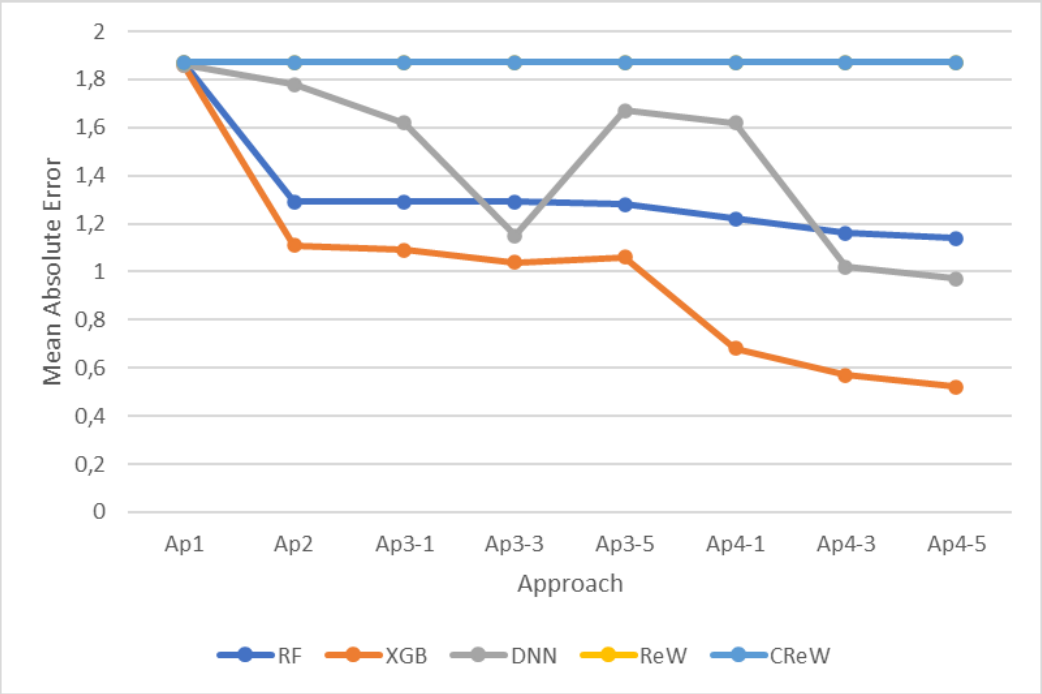

Figure 4.10: Comparison of the experimental results considering how the MAE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MAE for the 30% reserved data.

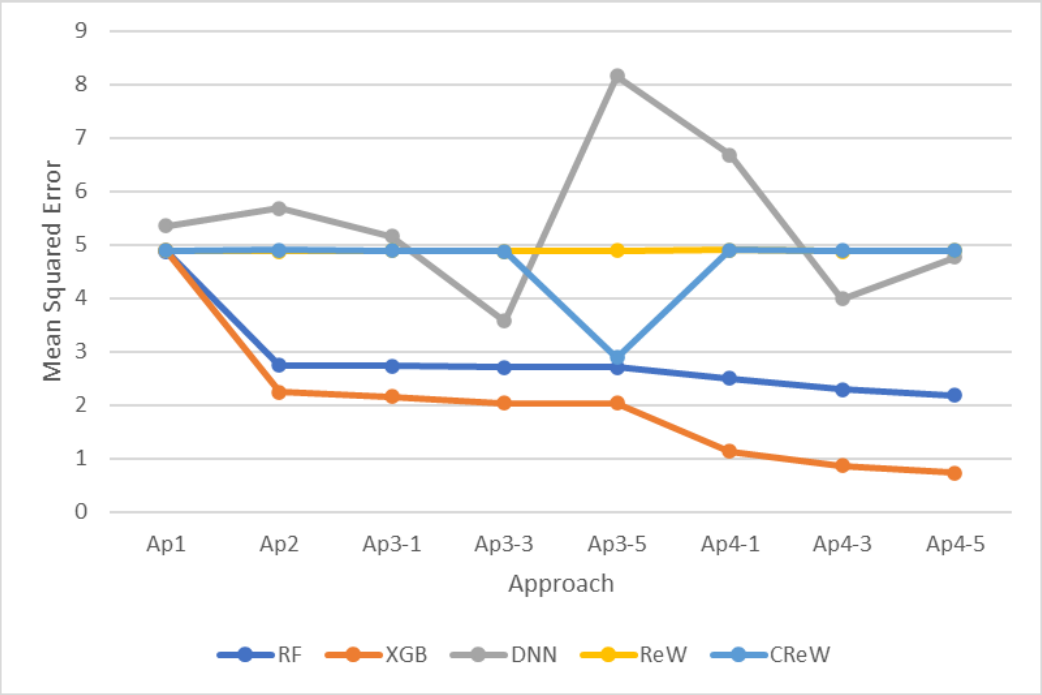

Figure 4.11: Comparison of the experimental results considering how the MSE for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the MSE for the 30% reserved data.

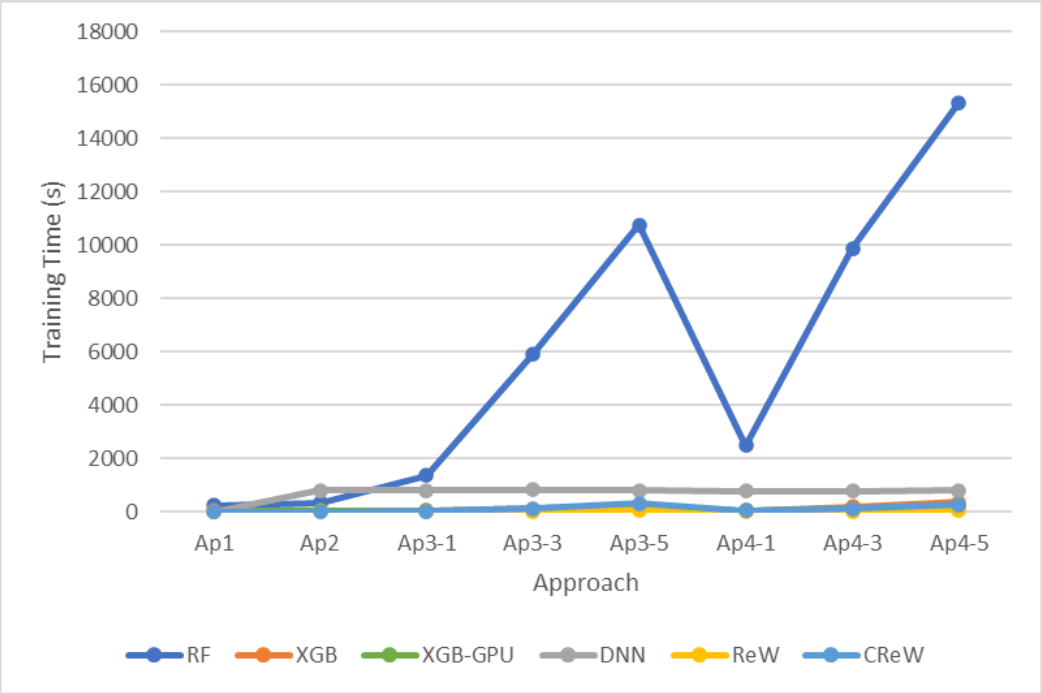

Figure 4.12: Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data.

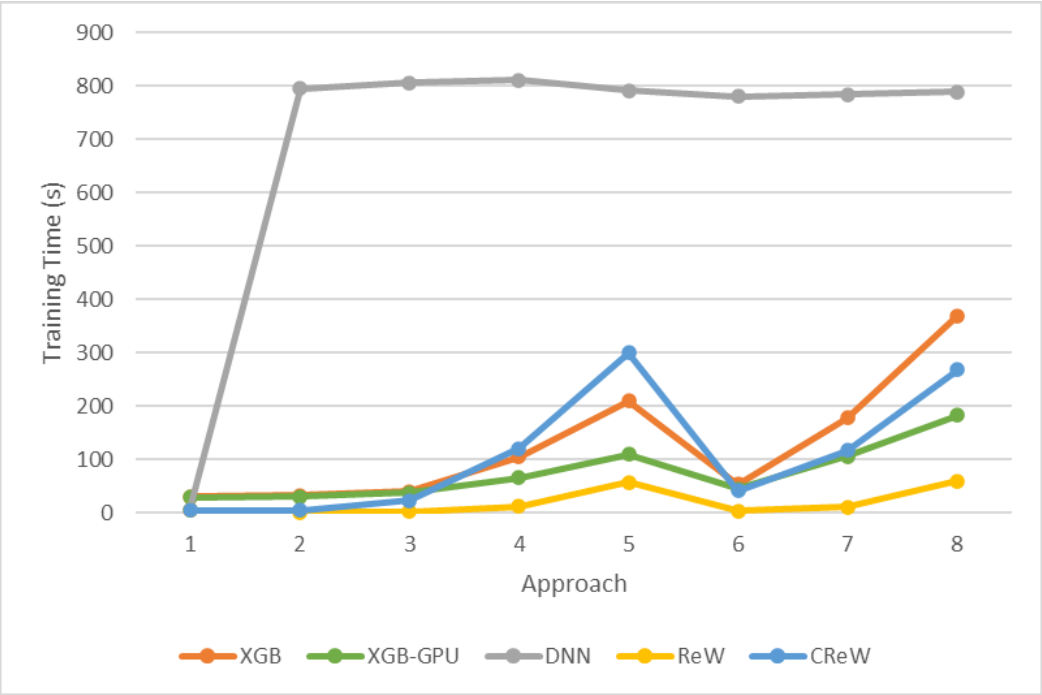

Figure 4.13: Comparison of the experimental results considering how the training time for each model changed given each one of the proposed approaches. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to train each model using the 70% reserved data. In this figure, the RF model was suppressed to evaluate the other models better.

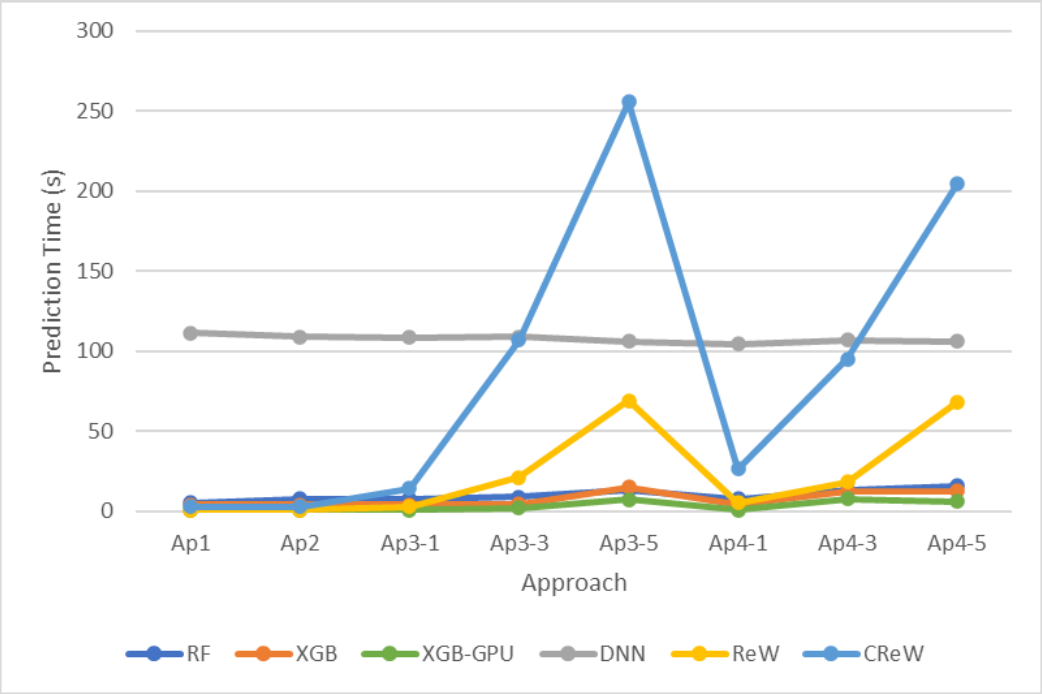

Figure 4.14: Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data.

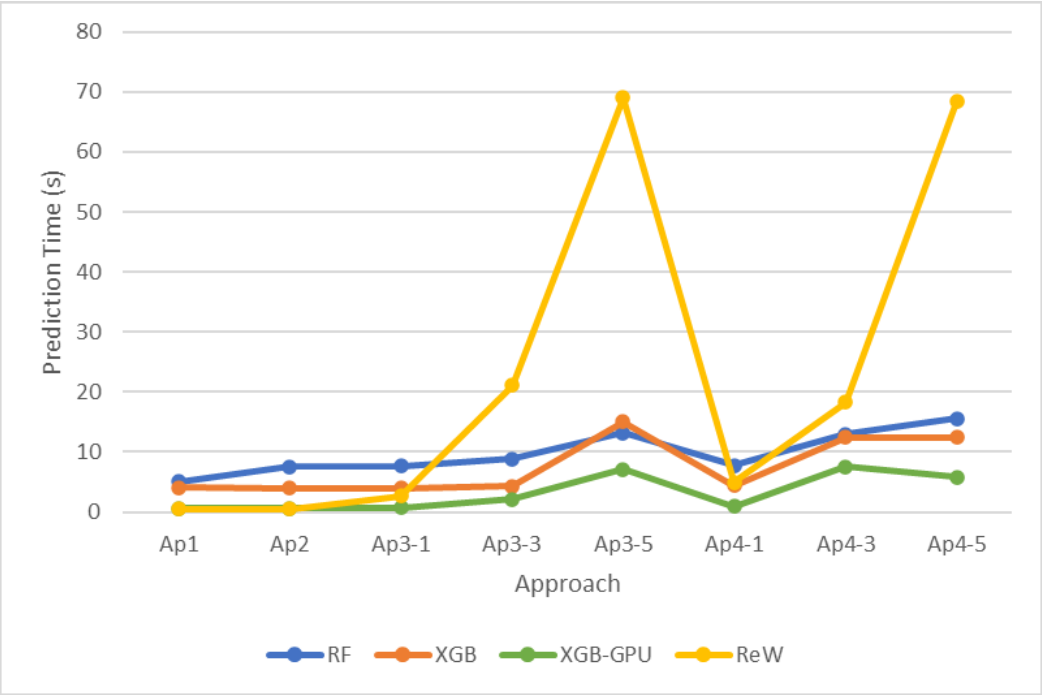

Figure 4.15: Comparison of the experimental results considering how the prediction time for each model changed given each one of the proposed approaches using the real data. The X-axis indicates each one of the approaches, while the Y-axis indicates the time taken to generate the predictions for the 30% reserved data. In this figure, the DNN and CReW models were suppressed to evaluate the other models better.

Chapter 5

Discussion

This chapter presents the main discussions regarding the results obtained and which were presented in the previous chapter. Initially, some considerations are presented regarding the general results obtained by the models. Next, the error analysis continues based on the results obtained by the GBDT model, given that it was the one that performed best in terms of error metrics.

5.1 Initial Considerations

Considering the GBDT model could reach results that are within an acceptable range for clinical applications, the results for this model will be used for exploratory data analysis. The results of a random iteration of the experiments were taken but, as suggested by the considerably low standard deviation of the results, the same patterns can be found in the other experiments. Although further investigation needs to be performed, some initial thoughts on why the other two models could not reach better results can be presented.

Regarding the DNN model, it is worth mentioning that it was not possible to perform an exhaustive hyperparameter and topology exploration due to infrastructure limitations, and only a few simple combinations that are commonly used for initial evaluation were tested. Although works like [76] can be cited that demonstrate that tree-based methods usually outperform DNN on tabular data, there are also a group of DNN models based on transformers architecture – like TransformerTab [77] – which was designed for application on this type of data and are candidates for future experiments. Nevertheless, it can be observed from the results that this model's results improved as the approaches were enhanced.

Also, it is worth noticing that DNN models have the ability to capture implicit patterns in data when there is a hierarchical representation of features, as described in works such as [78] and [79]. However, it is possible that the approaches presented in this work were not able to correctly express this hierarchy, thus compromising the

performance of the architectures used. Nevertheless, with the addition of specific filters in the topology and attention mechanisms, this limitation can be overcome and provides scope for future work.

The RegressionWiSARD presented a similar behavior as its classification counterpart: while suboptimal prediction performance, the overall time required to train the model surpassed even the XGB using a GPU. Both WNN models presented the same values across all experiments, which indicates that they could not accurately estimate the data and two possibilities should be further investigated. The first one is regarding the binarization process. Perhaps it was not possible to find the correct bit addressing and parameter configuration that enables the models to perform well. The second is regarding the data used and the output format. Although a numerical continuous output, the granularity of the target was low. If another observation in a different temperature interval was presented to the network, it could be possible to contribute to the models' performance.

5.2 Error Analysis on Simulation Data

Even though the temperature variation can be considered a continuous variable, the data used presented discrete values. In this way, it is possible to analyze what was the error for each expected Δ. As previously mentioned, Δ represents the expected temperature variation for a given input data. When evaluating the models' performance, this value with the model's prediction is compared. The difference between these two values is what is called *prediction error*. For the data not used for training, some inputs should be predicted as $\Delta = 1$ but are predicted as some other value, which may be small or large. The same holds for $\Delta = 2$, $\Delta = 3$, and so on. Then, a separation of this data into groups represented by the respective value of Δ that should be predicted was performed.

A kernel density estimation (KDE) plot was generated for each group to describe how the absolute value of the prediction error on the group was distributed. Besides, it must considered that there is a starting temperature in each Δ. For example, it is possible to consider a variation of 1ºC from 37ºC to 38ºC or from 42ºC to 43ºC. In this sense, each plot splits the data for each starting temperature in the form of a colored line. Figures 5.1-5.8 show the generated plots considering the predicted error on the dataset that was created using the simulation data.

For example, it is possible to visualize that there is not a considerable difference in the error distribution between the different starting temperatures, as the lines for all temperatures in the same plot describe similar behavior. This can be related to the monotonical aspect of temperature increase described earlier, and the GBDT model was able to capture it.

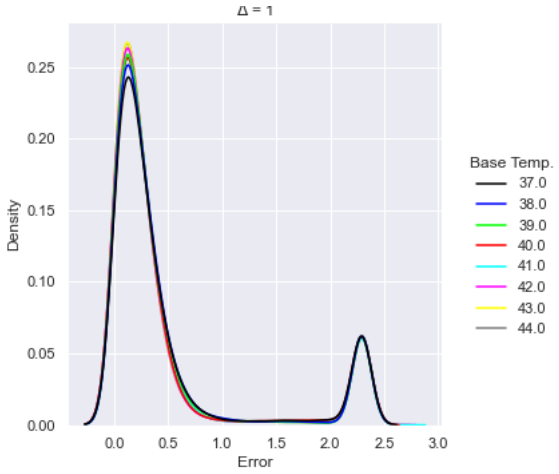

Figure 5.1: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 1$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

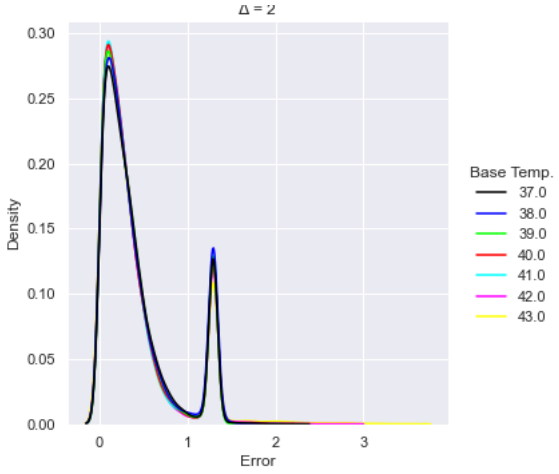

Figure 5.2: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 2$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

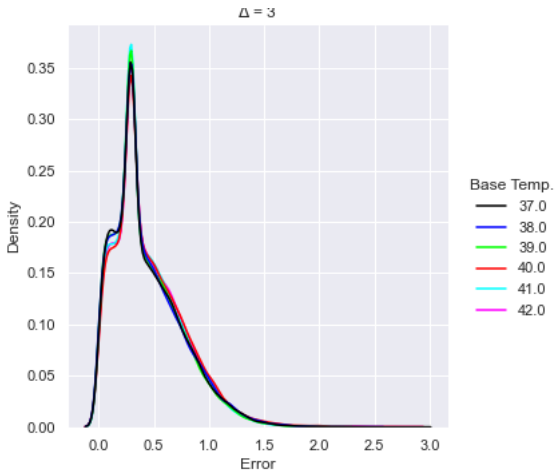

Figure 5.3: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 3$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

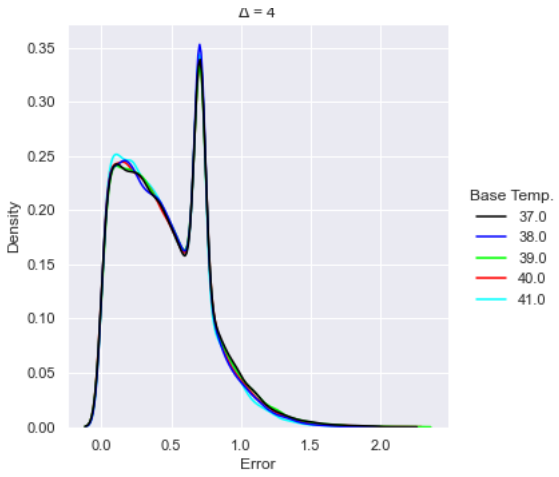

Figure 5.4: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 4$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

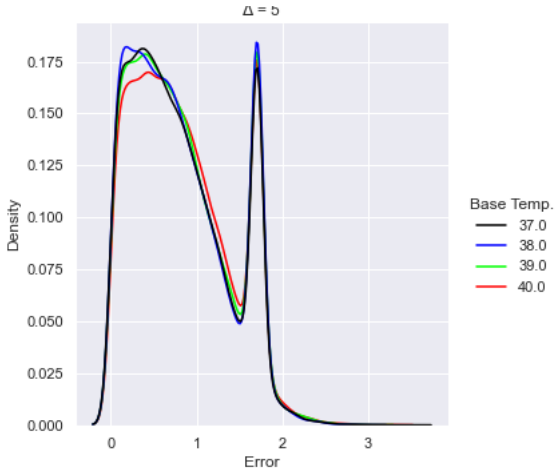

Figure 5.5: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 5$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

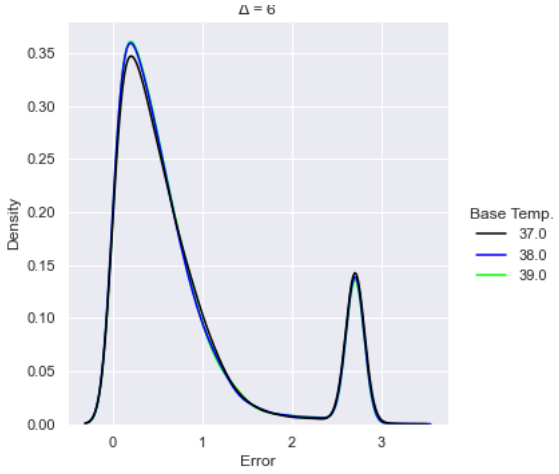

Figure 5.6: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 6$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

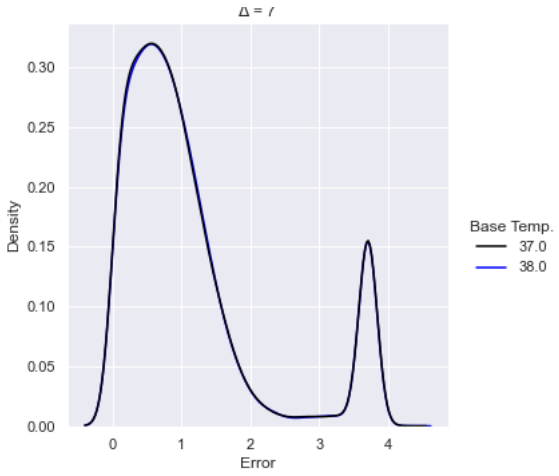

Figure 5.7: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 7$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

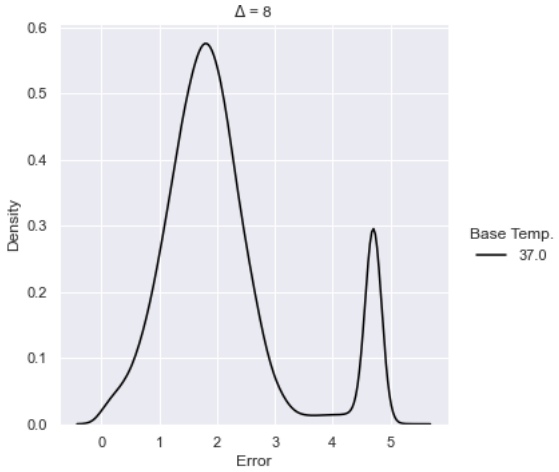

Figure 5.8: KDE plot of prediction error on simulation data considering an expected temperature variation $\Delta = 8$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X.

A common feature in all plots is the presence of a peak around the value of 3.4ºC. This fact was due to the external region composed of water, which always has the same behavior (slope 0). These points appear in all temperature variations and, when combined with the imbalance of classes, cause the final result for these points to be given by a weighted average. To better understand this effect, consider a neighborhood k = 1. When considering the external water region, the inputs are built by selecting i) pixels in the temperature image with a grayscale value of 0 (a black pixel) and all eight pixels within its neighborhood, which are also black in the majority of the cases), and ii) pixels in the CBEUS image that, in this scenario, presented a slope parameter of 0.0, since there was no variation along the heating procedure. Then, our input would be a zero-array of size 18 (9 from the temperature and 9 from the CBEUS). This input is found across all possible temperature variations, which leads to the same input being addressed to all possible expected outputs.

In situations like that, regression models usually output the average value between the possible predictions, which would be 4.5ºC in this case. It turns out that the dataset is not balanced: the lower temperature variations appear more than the higher temperature variations. This causes the average to move in the direction of the lower values, and, in our experiments, reach a value around 3.4ºC. As this behavior occurs in a significant portion of the dataset, it could create peaks in the KDE plots.

Nevertheless, by considering most of the plots it is possible to observe that prediction errors for smaller Δ variations are concentrated close to a margin of 0.5ºC. On the other hand, as Δ increases, the prediction error range also increases, although it is worth recalling that there are fewer training observations for high-temperature variation, which can impact the models' performance.

The analysis can also be performed by considering the region where the predictions were made. Figures 5.9-5.16 present heatmaps obtained from all predictions made for the possible variations considering the base temperature image of 37ºC. Each heatmap presents a gradient pattern that ranges from white to red. The variable in question is the absolute prediction error, i.e. for each pixel in a given image, the absolute value of the difference between the expected temperature output (represented in the image title) and the prediction made by the model. In each figure, there are two kinds of pixels. Gray pixels were used for training the model or could not be accessed because of neighborhood limitations. All other pixels were used for prediction. Low errors are associated with the white color, as complete white means an error of 0ºC. As the error increases, the pixel turns red.

It is possible to observe that, apart from $\Delta = 3$ and $\Delta = 4$, all heatmaps show a concentration of red pixels around the phantom muscle and lipid regions in the

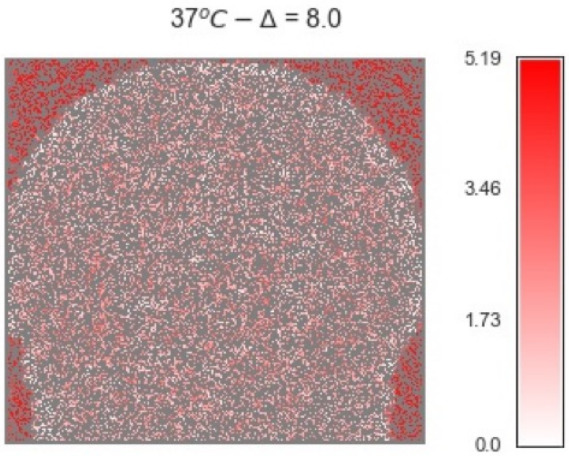

Figure 5.9: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 8ºC.

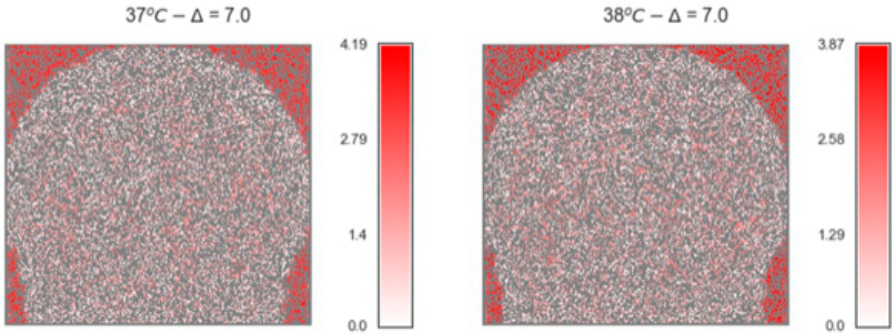

Figure 5.10: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 7ºC.

phantom, which once again relates to the water medium and the presence of the same variation pattern in all temperatures. The two temperatures that present lower error on these regions are exactly the ones that are close to the weighted average previously mentioned.

Another aspect that is worth mentioning is that the scale of error increases along with the temperature, mainly when considering temperature variations starting from 6ºC. One possible explanation is related to imbalanced data, as fewer examples in these groups can affect the prediction performance. On the other hand, once again the region representing the water medium negatively affects the results, since it is concentrated around a value that is closer to the lower temperature variations.

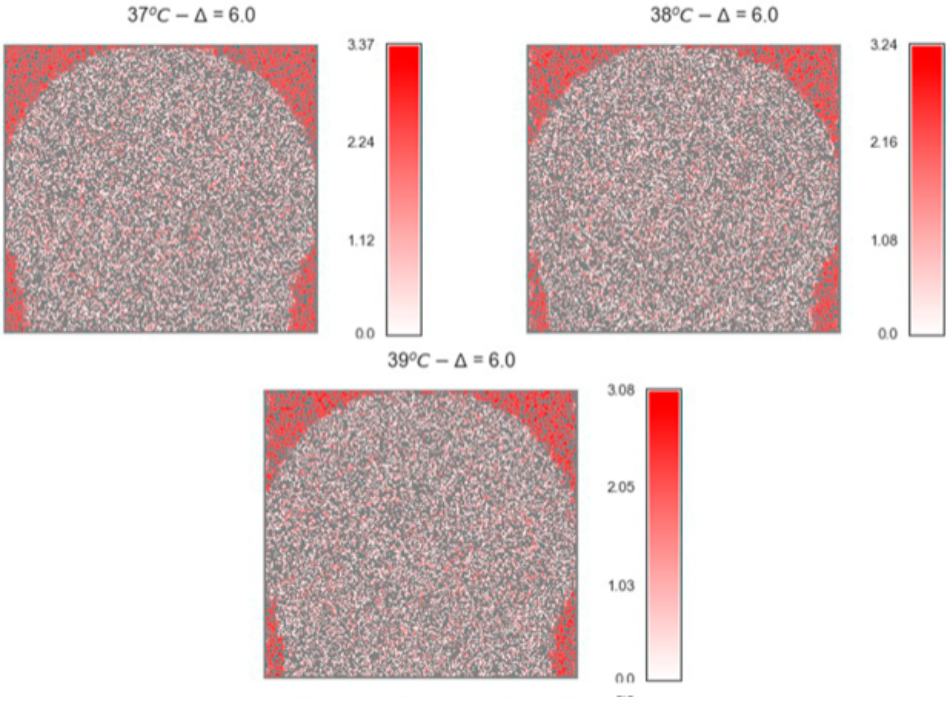

Figure 5.11: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 6ºC.

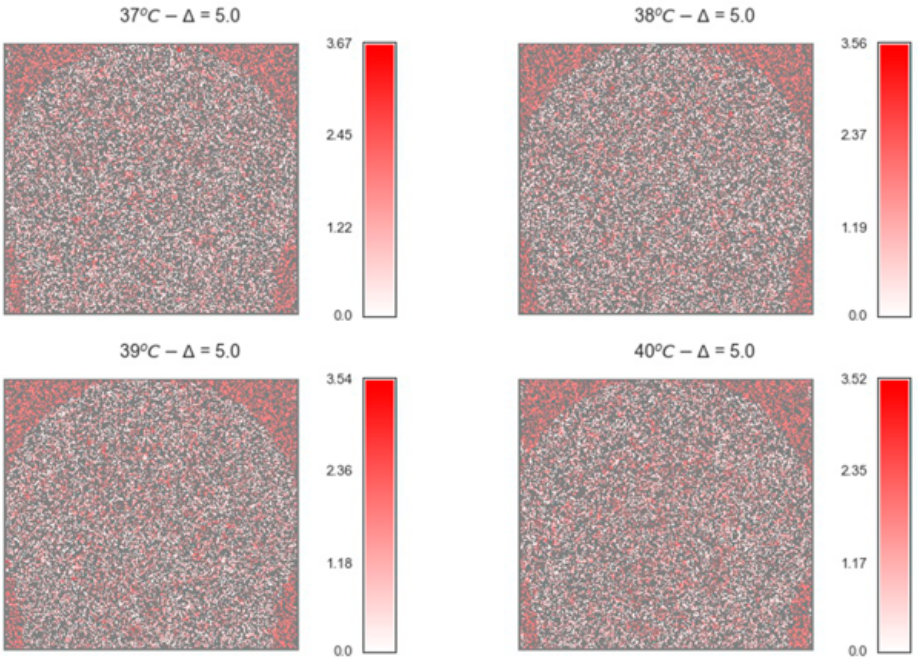

Figure 5.12: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 5ºC.

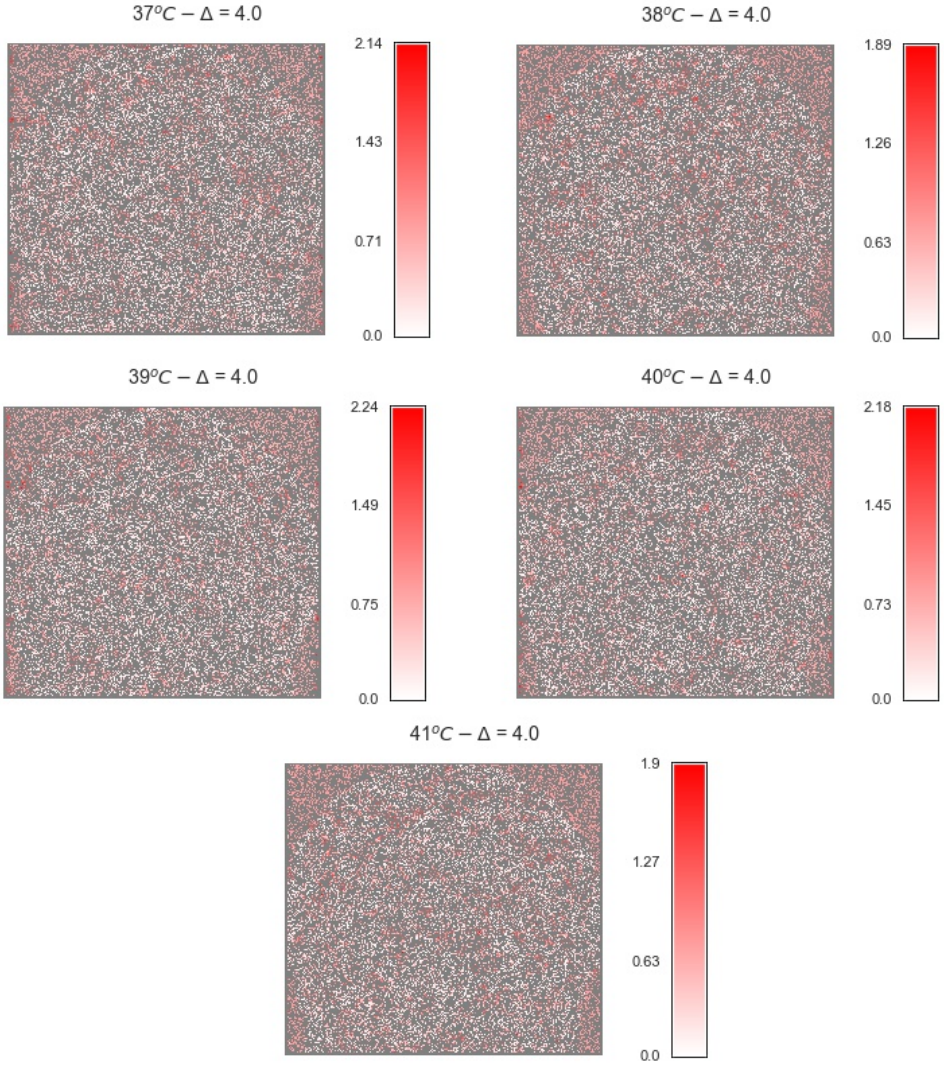

Figure 5.13: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 4ºC.

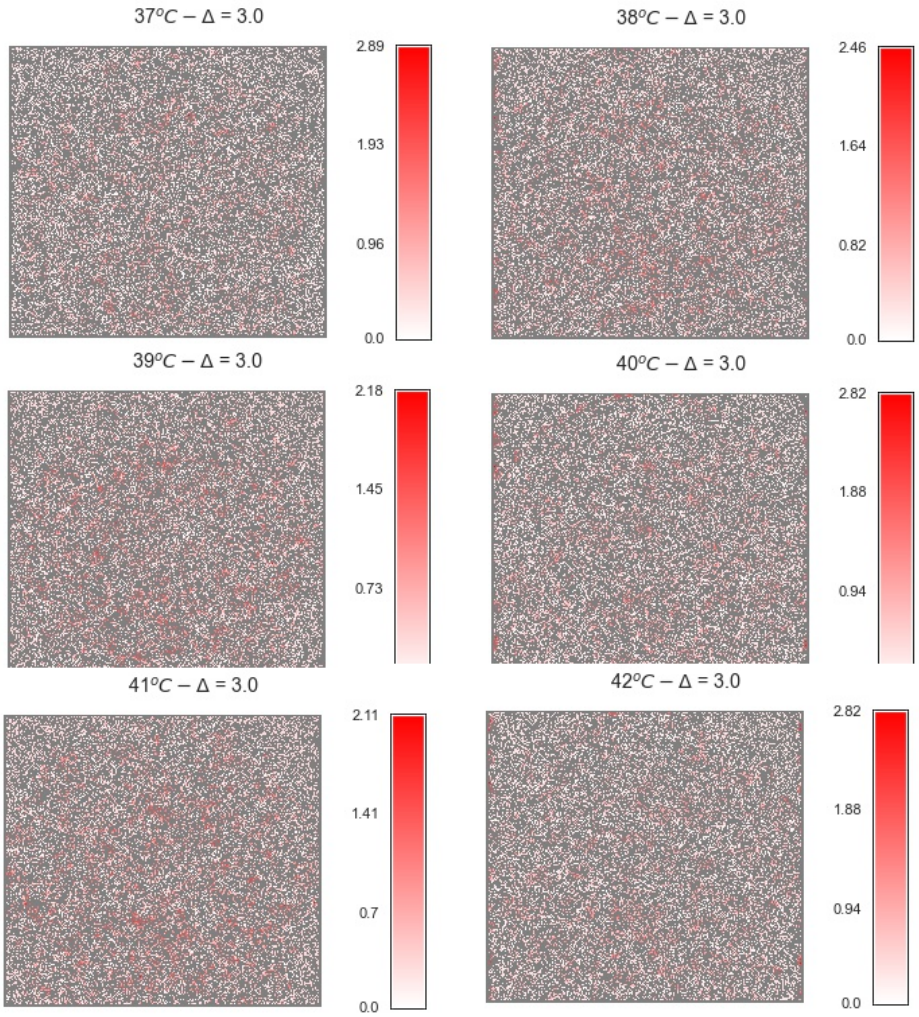

Figure 5.14: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 3ºC.

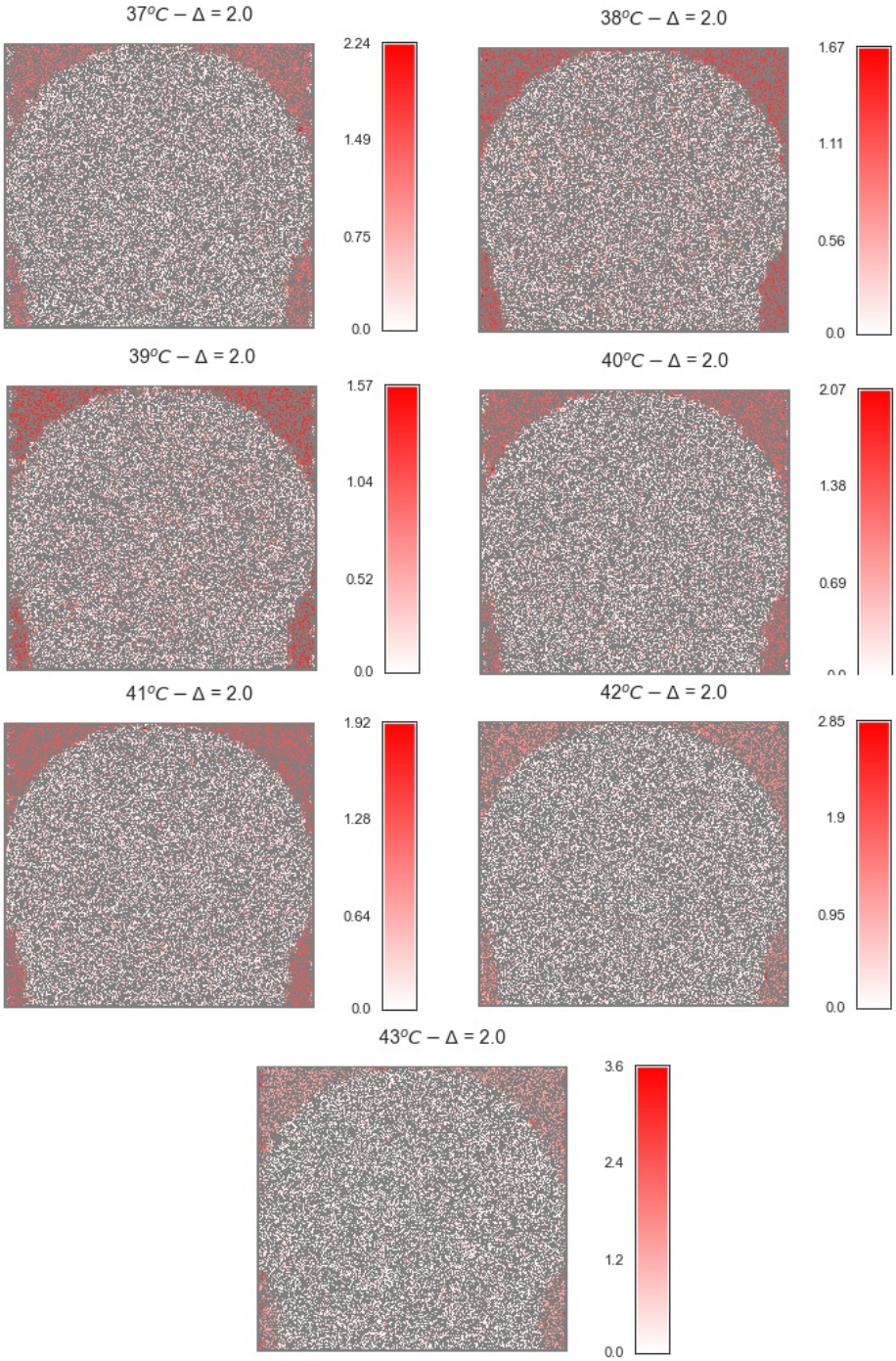

Figure 5.15: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 2ºC.

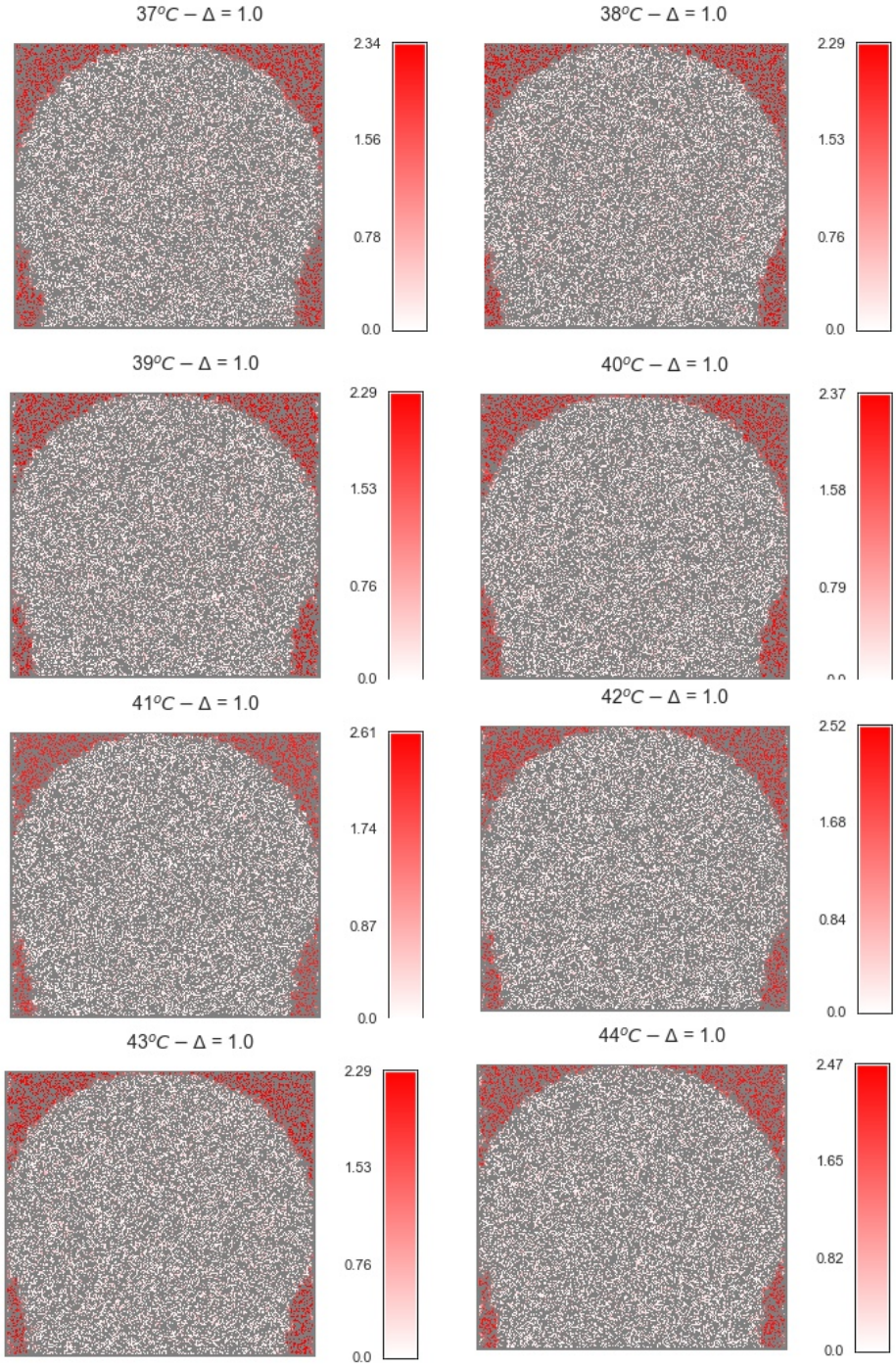

Figure 5.16: Comparison of the prediction error of the GBDT model from a spatial perspective on the simulation data and considering a temperature variation Δ of 1ºC.

5.3 Error Analysis on $in - vitro$ Data

The same set of analyses was performed for the porcine muscle data. Figures 5.17-5.25 show the KDE plots referring to the *in vitro* data, while figures 5.26-5.33 display the heatmaps to illustrate the absolute error from a spatial perspective. The results present a fairly similar behavior observed on the simulation data: the KDE plots of the lower base temperature display an error concentration close to 0.5^{o}C and also present a second data concentration, which indicates that there are also a collection of pixels that present the same behavior along all possible temperatures and is affected by the data unbalance.

Regarding the heatmaps, it is possible to observe that the concentration of pixels with the highest error rate is located in the lower part of the image, which refers to a region where there is a layer of lipid tissue, while the upper part consists of the muscle tissue. In the muscle region, it is observed that the degree of error presents similar behavior to that of the simulation data, where the lowest values are found in variations of 1^{o}C.

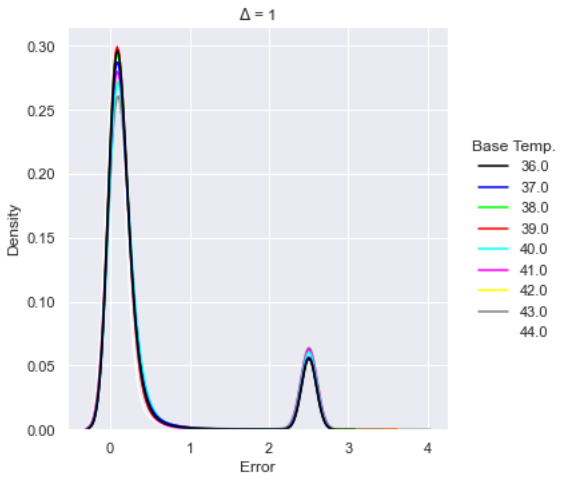

Figure 5.17: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 1$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

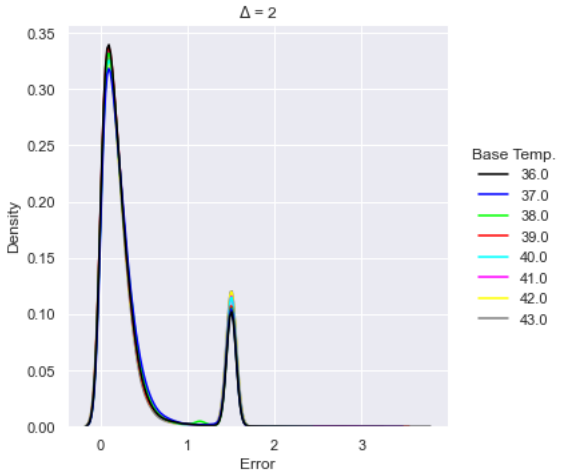

Figure 5.18: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 2$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

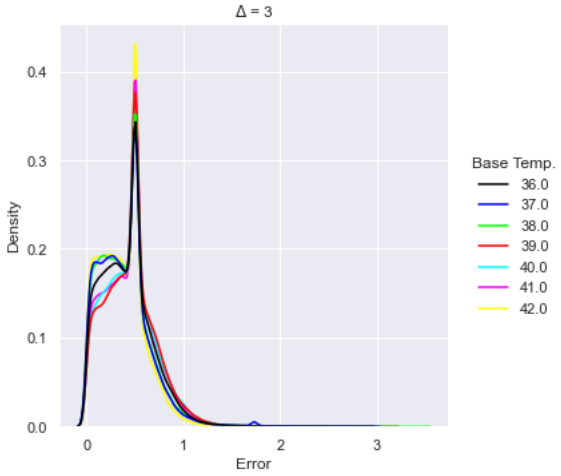

Figure 5.19: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 3$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

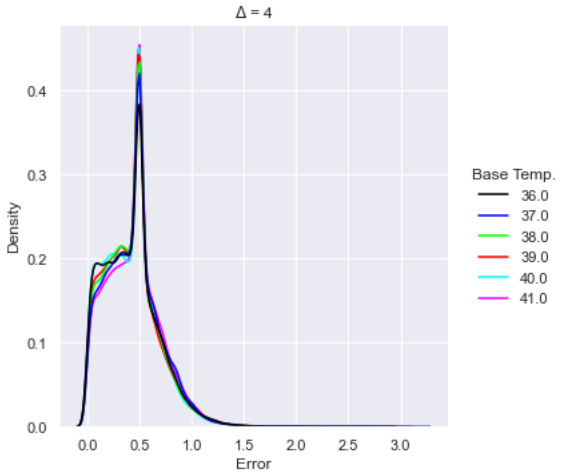

Figure 5.20: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 4$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

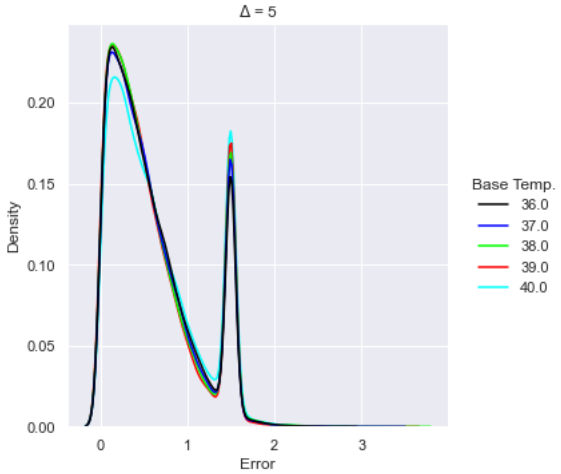

Figure 5.21: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 5$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

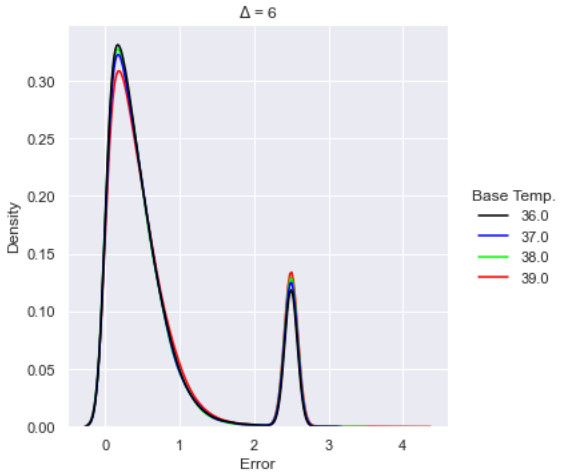

Figure 5.22: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 6$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

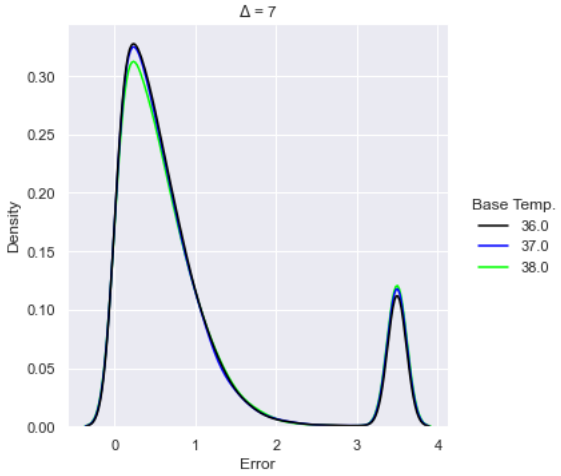

Figure 5.23: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 7$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

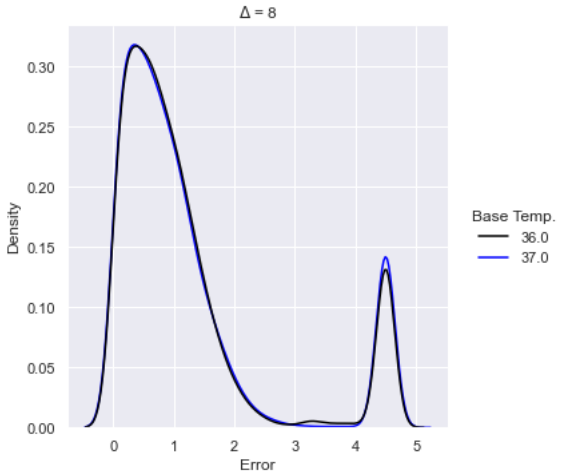

Figure 5.24: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 8$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X. Each colored line represents a base starting temperature.

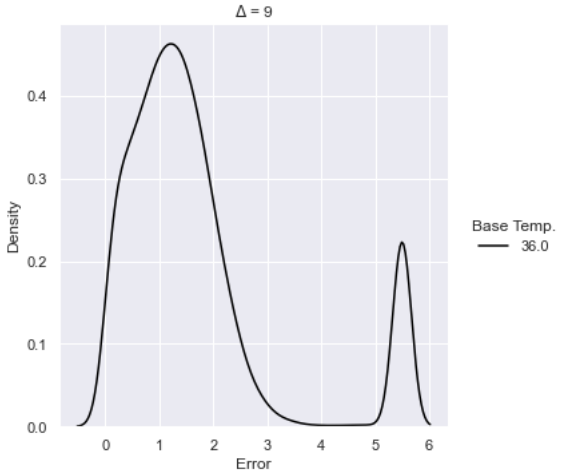

Figure 5.25: KDE plot of prediction error on real data considering an expected temperature variation $\Delta = 9$. The X-axis indicates the absolute error between the expected output and the predicted value. The Y-axis indicates the normalized number of observations within that range of error indicated by X.

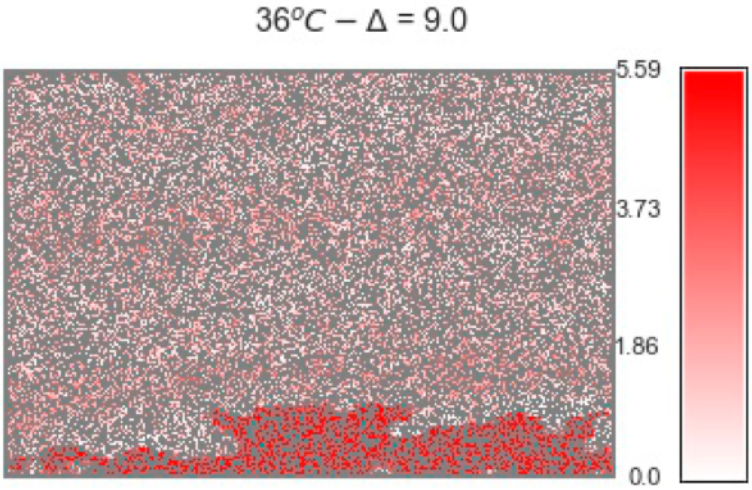

Figure 5.26: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 9^{o}C.

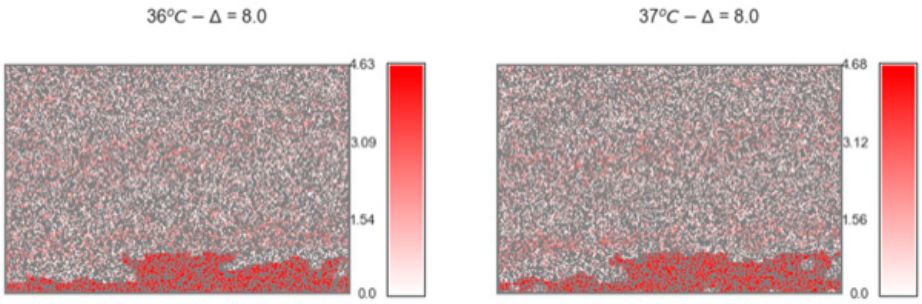

Figure 5.27: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 8^{o}C.

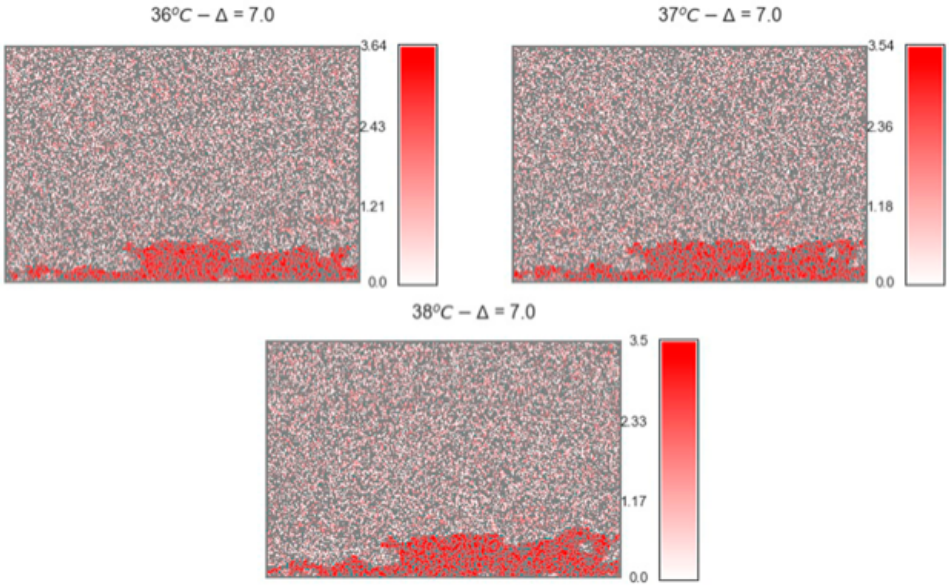

Figure 5.28: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 7ºC.

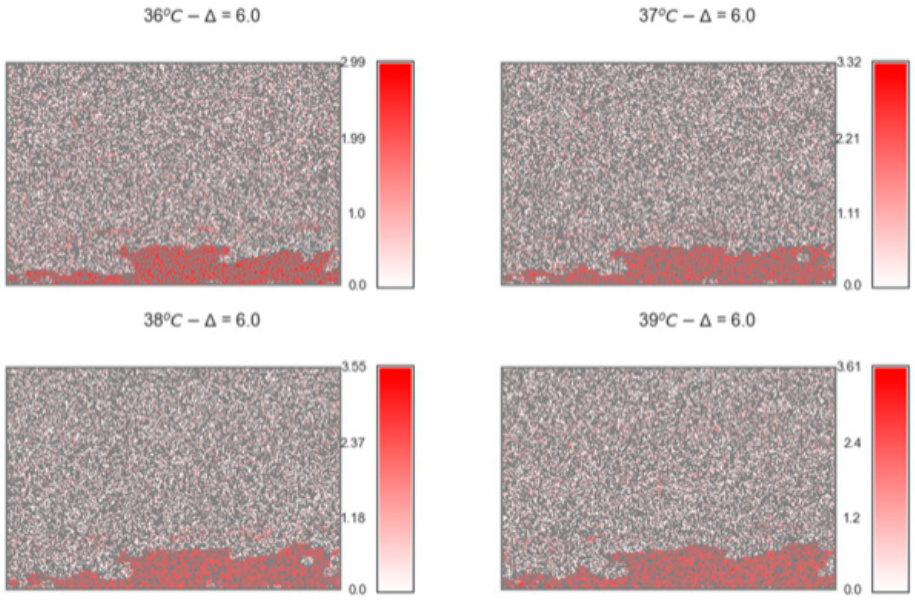

Figure 5.29: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 6ºC.

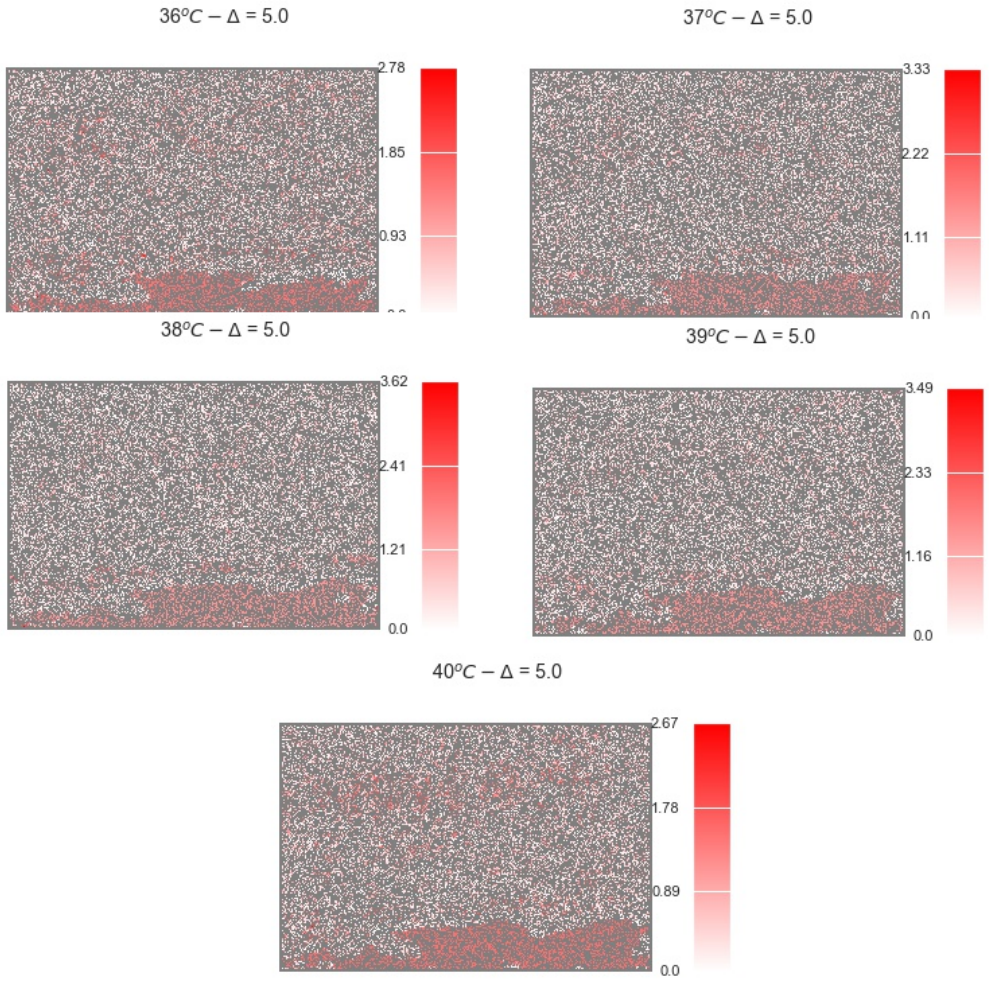

Figure 5.30: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 5^{o}C.

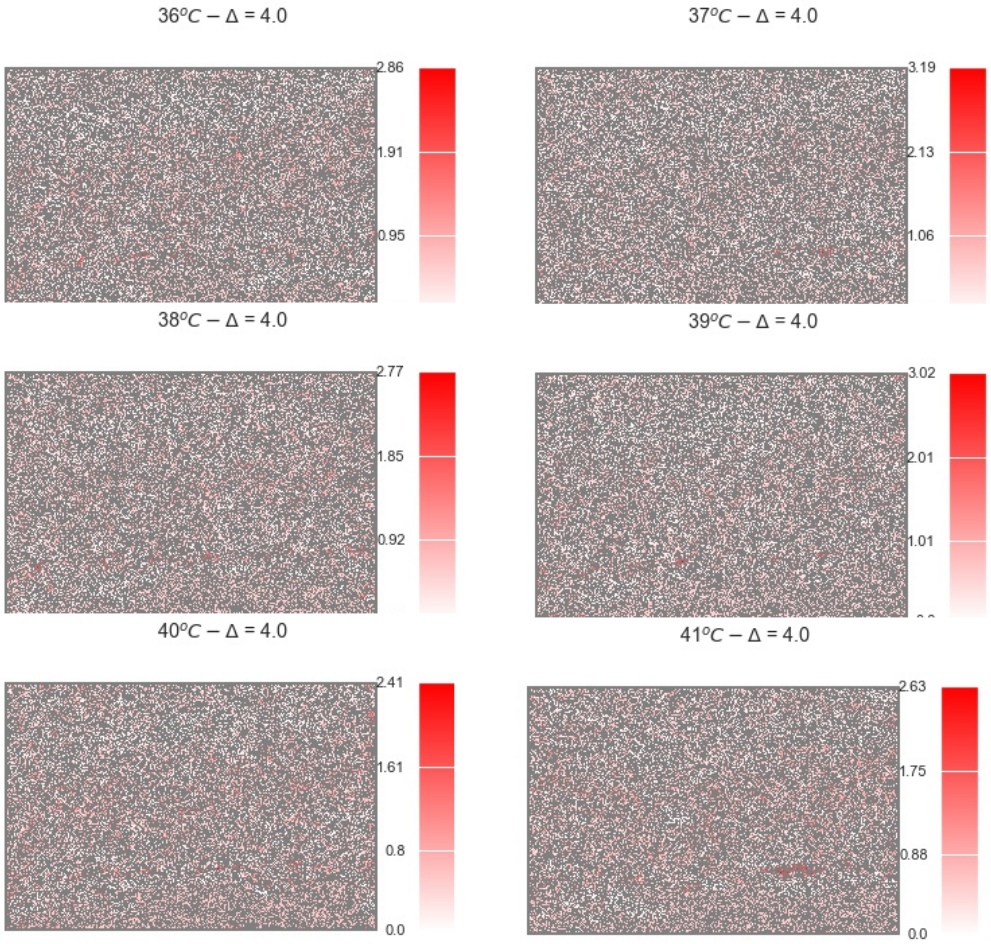

Figure 5.31: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 4^{o}C.

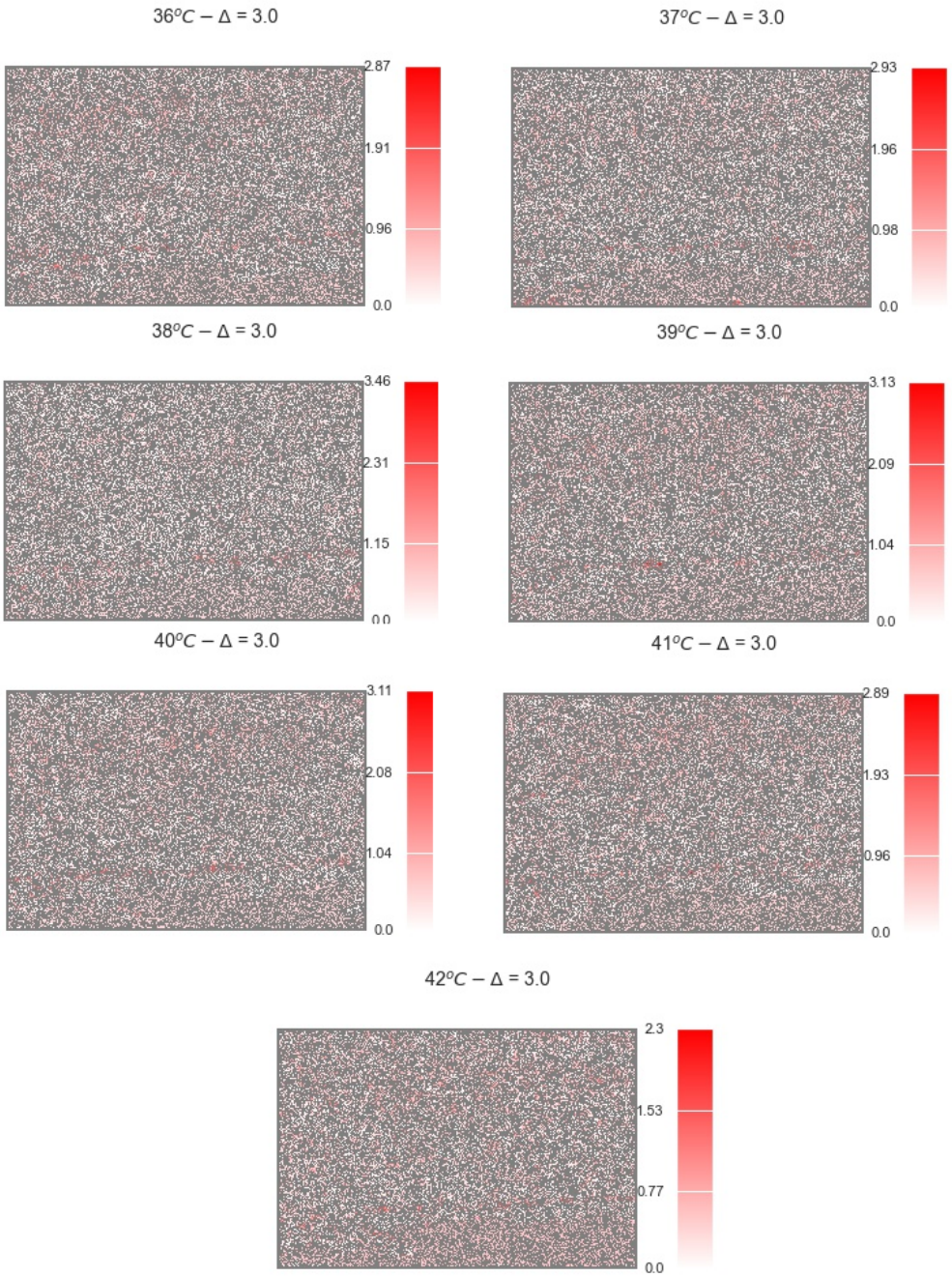

Figure 5.32: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 3^{o}C.

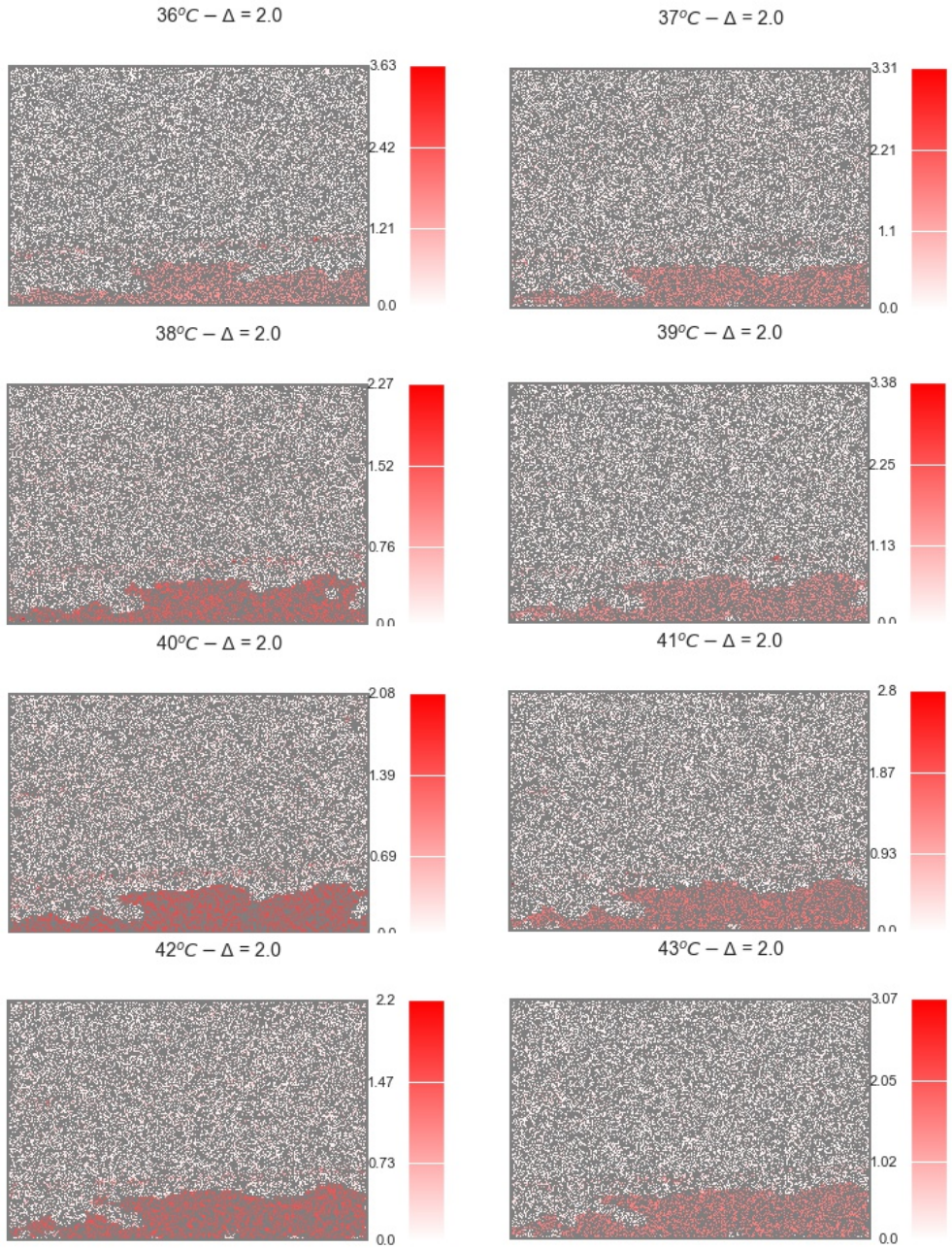

Figure 5.33: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 2ºC.

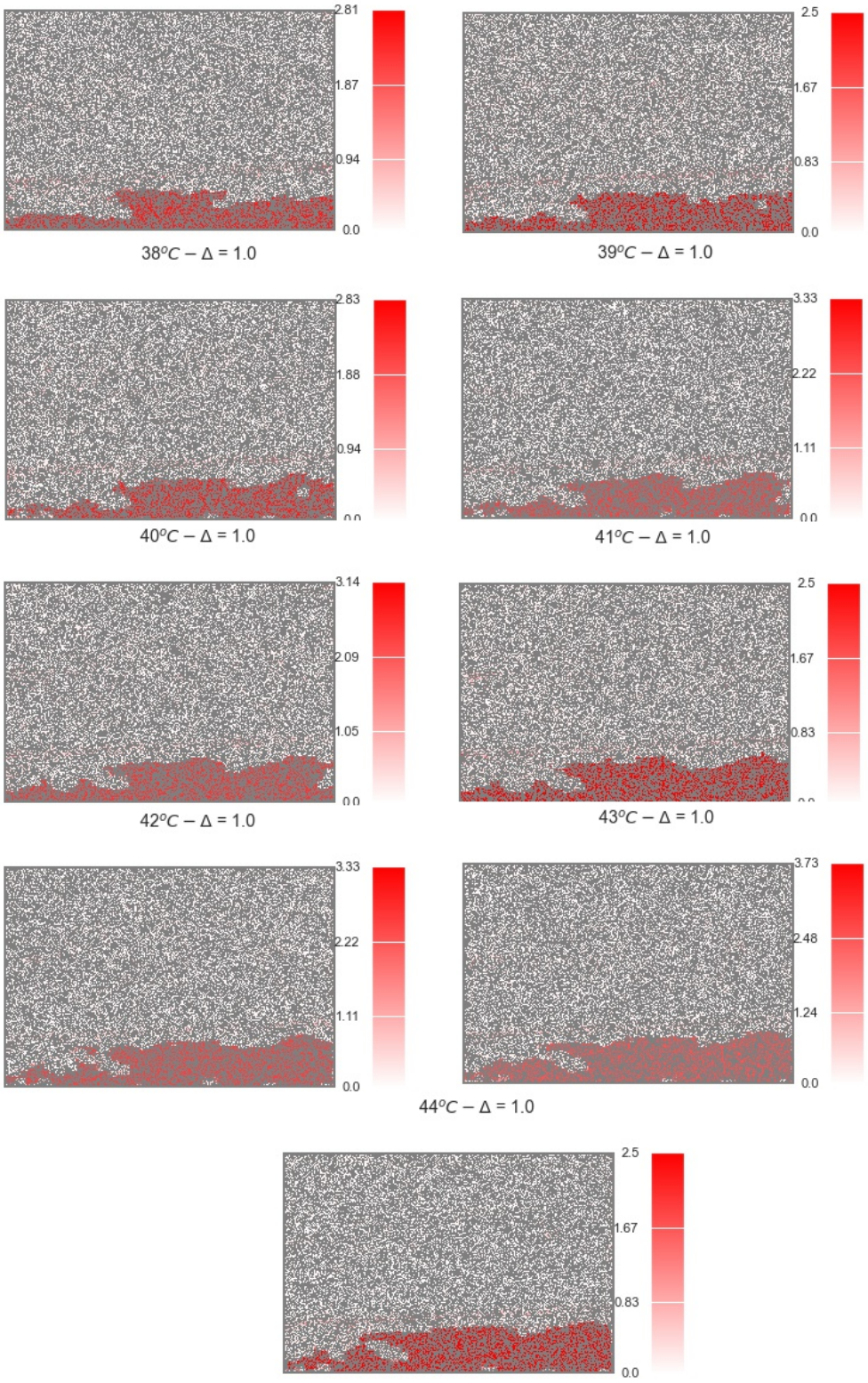

Figure 5.34: Comparison of the prediction error of the GBDT model from a spatial perspective on the real data and considering a temperature variation Δ of 1ºC.

One last aspect worth mentioning is the data imbalance mentioned during the analysis. Table 5.1 shows the average percentage of the degree of imbalance in the data sets from each of the sources used. It is possible to observe that there is on average ten times more data from the temperature variation class $\Delta = 1$ when compared to the number of observations referring to the greatest possible temperature variation in each scenario.

Δ	Simulation	*In-vitro*
1	22%	20%
2	19%	18%
3	17%	16%
4	14%	13%
5	11%	11%
6	8%	9%
7	6%	7%
8	3%	4%
9	-	2%

Table 5.1: Average degree of imbalance between the different temperature variations existing in the datasets based on the simulation and real data.

5.4 Error Analysis on Muscle Tissue

Given that there is knowledge regarding the types of tissue in the regions existing in the phantom, it is possible to map the behavior of the angular coefficients computed during the creation of the CBEUS image. The table **??** shows, for each type of tissue in the phantom, the number of image pixels corresponding to the type of tissue and, subsequently, the number of pixels with each of the possible directions of the angular coefficient: positive (greater than zero), negative (less than zero) or neutral (equal to zero). In addition, the percentage of each type of growth is also shown.

It is possible to observe that muscle tissue pixels have a much better defined behavior concerning the others. It is noted that the angular coefficients are mostly negative, i.e. the image tends to be darker in regions related to this type of tissue. The other types of pixels (lipid tissue and water) have a less regular behavior, presenting high variation between negative and neutral coefficients. This led to an attempt to observe how the error behaved in the more regular region. In this way, we sought to isolate the pixels in the muscle tissue region and evaluate the model's performance considering only these observations.

Figure 5.35a shows the heat map generated for the temperature variation $\Delta = 1$ considering the initial temperature of 37^{o}C, while Figure 5.35b shows the same heat map with isolation of the muscle tissue region. It is possible to observe that this

Type	Total	Coefficient	Count	%
Lipid	3545	Positive	1,607	45.33
		Negative	1,890	53.31
		Zero	48	1.35
Muscle	2541	Positive	357	14.05
		Negative	2,184	85.95
		Zero	0	0
Water	98774	Positive	33,186	33.6
		Negative	44,763	45.32
		Zero	20,825	21.08

Table 5.2: Behavior of the angular coefficients of the CBEUS parametric image when splitting the image between the different types of tissue in the digital phantom.

region is composed of mostly bright pixels, indicating a lower error level. Not only that, by computing the average absolute error of the pixels in this region, it was possible to obtain a value of 0.41, thus reaching a value lower than the established gold standard.

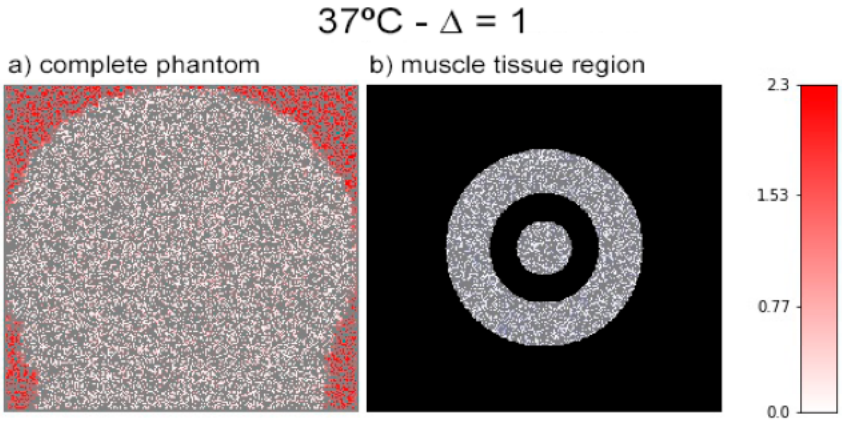

Figure 5.35: Error visualization of muscle-related region in the phantom figure. a) the standard heatmap considering the base image at 37^{o}C and temperature variation $\Delta = 1$, as presented in Figure 5.9. b) The same heatmap but displaying only the regions related to muscle tissue in the phantom.

Chapter 6

Conclusion

This work aimed to address the problem of numerically estimating the degree of temperature variation in B-mode ultrasound images. To accomplish this task, a data modeling was proposed that allows the interpretation of this as a supervised learning problem, enabling the use of machine learning models to find intrinsic relationships between ultrasound image data and be able to generalize these relationships to efficiently estimate variation. By efficiency, it was tried to achieve error rates within a margin of 0.5^{o}C, which is considered in the literature as the gold standard in this type of problem. The proposed data modeling was built after four iterations of experiments resourced to the use of information retrieved from CBEUS, which is a parametric image that is generated based on the changes in backscattered energy during the process of temperature variation, and that describes how the intensity of pixels in ultrasound images varied throughout the process of increasing temperature. After the modeling proposal, data sets were created using two sources as a basis, one of which came from a phantom that represented the physical properties of tissues and whose images were generated using simulation software, while the other came from a process using an *in-vitro* sample of porcine muscle heated in a water bath and whose images were captured using a conventional ultrasound device. To evaluate the modeling, some supervised learning models were used. Models based on decision trees - Random Forest and Gradient-Boosting Decision Trees -, basic architectures of deep neural networks and also weightless neural network models were analyzed. For each model, it was measured the Mean Absolute Error and Mean Squared Error metrics, and it was also measured the time taken to complete the training and prediction tasks.

The experimental results on both generated datasets pointed out that the proposed approach would achieve its best performance metrics by using the Gradient-boosting Decision Tree (GBDT) model, as it was able to estimate the temperature variation with a mean absolute error of 0.55 degree Celsius in a simulated environment and 0.41 degree Celsius when considering only muscle tissue regions. When

applying the method on the *in vitro* real sample the model presented 0.54 degree Celsius of mean absolute error. The best performance regarding prediction would appear in small temperature variation changes (1 degree Celsius).

Alongside the proposed approach, two extensions of the weightless neural network WiSARD were presented that would enable this architecture to perform regression tasks. Both extension models were compared to classic regression models in literature and then also applied in the proposed modeling. The main reasons behind this proposal are the benefits of using the weightless architecture, like low energy consumption and easy to implement as hardware to work as an embedded system. When compared to the classic models, both proposed extensions presented similar behavior to it classification counterpart: although suboptimal estimation performance, they present low training time, which is a good characteristic for online learning domains.

Although the results provided in this research are highly promising, the current state of this work should be considered as a proof of concept, and the clinical application for the presented method still requires a deeper investigation, which leads to comments regarding the future perspective for this work, which is the focal point of discussion in the following section.

6.1 Future Work

Regarding future works, a handful of opportunities arise, both in the direction of extending the research aiming for a practical application as for improving the proposed methodology from a machine learning perspective. Considering the latter, exploring the wide range of possible hyperparameter configurations of the models is within the first experiments to be performed, especially when considering the Deep Neural Network (DNN) architectures that were applied. As stated in the chapter 5, few naive variations of DNNs were used, and it is expected that more robust configurations could improve the overall performance of this group of models. Also, the necessity to limit the training time of the DNNs was an issue that had to be addressed. Nevertheless, it should be stated that the time required for a model to be trained would not be a limit factor for real applications, since the proposed approach is not expected to deal with the challenge of online learning, which would be the case for resourcing to the weightless neural models (WNN).

Referring to the two weightless models produced during the development of this work and that perform function approximation to output a continuous numerical value - RegressionWiSARD (ReW) and ClusRegressionWiSARD (CReW) - the experiments demonstrated that both models struggled with the combination of the proposed approaches and the hyperparameter configurations. It is known that choos-

ing an inadequate binary representation for a given problem is enough to harm the WiSARD-family models. In this sense, it is of high importance to research aiming to find a proper encoding technique for this problem. Also, these two models can be seen as quite simple when compared to ensemble methods, and the usage of both ReW and CReW in an ensemble environment was already initially explored in []. Resourcing to this scenario could lead to a performance improvement, but would still find problems if the binary representation is not good enough. Still related to weightless models, it would be interesting to develop work that performs a mathematical derivation proving the feasibility of using the other proposed central tendency measures, to demonstrate the capacity of these calculations to operate as an appropriate kernel function.

Another aspect would be the possibility of changing the way the approaches were used. Although it was possible to achieve good levels of precision with the analysis of pixel values, more robust techniques could be used, such as feature extraction for dimensionality reduction. This resource could be explored in the future.

Furthermore, a very important factor to be addressed is the data set. Firstly, regarding the data variation used, which is considered small. It would be interesting if the tests performed in this work were replicated on different data samples, such as simulated data using phantoms with configurations in other formats. Secondly, regarding the granularity of the dependent variable, it was mentioned that the work only considered discrete temperature values. It would be interesting to add observations using intermediate temperatures to improve the quality of the interpolation, which would also be a factor that could bring improvements to WANN models.

Another aspect concerns the evaluation of cross-models, in the sense of evaluating the performance of models that were trained on one database when trying to estimate the variation in another database. Initial experiments were carried out in this regard and the preliminary results were promising, but further investigation is necessary. Besides, an intuition generated by the work is that it would be difficult for a single model to be able to perform adequately for each and every situation. As evaluated in chapter 5, the selected model achieved greater generalization capacity in regions of muscle tissue. Thus, one path to be followed would be to integrate with image segmentation methods to define which regions are composed of each type of tissue, followed by the application of a model specialized in that type of tissue.

Finally, one of the requirements for enabling a practical application of the method is currently limited by the fact that it expects both the transducer and the medium to remain in a fixed position during the entire process. But this would not be the case for a practical scenario. In this sense, it is proposed to develop a methodology for incorporating a motion-tracking algorithm to compensate for image mobility.

References

[1] BROSSEAU, L., YONGE, K., WELCH, V., et al. "Thermotherapy for treatment of osteoarthritis", *Cochrane Database of Systematic Reviews*, oct 2003. doi: 10.1002/14651858.cd004522.

[2] CHAN, V., PERLAS, A. "Basics of Ultrasound Imaging". In: *Atlas of Ultrasound-Guided Procedures in Interventional Pain Management*, pp. 13–19, New York, NY, Springer New York, 2011.

[3] WHO STUDY GROUP ON TRAINING IN DIAGNOSTIC ULTRASOUND : ESSENTIALS, P., STANDARDS (1996 : PHILADELPHIA, P., ORGANIZATION, W. H. "Training in diagnostic ultrasound : essentials, principles and standards : report of a WHO study group". 1998.

[4] WUST, P., HILDEBRANDT, B., SREENIVASA, G., et al. "Hyperthermia in combined treatment of cancer", *The Lancet Oncology*, v. 3, n. 8, pp. 487–497, 2002. ISSN: 1470-2045.

[5] WATSON, T. "Ultrasound in contemporary physiotherapy practice", *Ultrasonics*, v. 48, n. 4, pp. 321–329, 2008. ISSN: 0041-624X. The Resurgence of Therapeutic Ultrasound: A 21st Century Phenomenon.

[6] SACCOMANDI, P., SCHENA, E., SILVESTRI, S. "Techniques for temperature monitoring during laser-induced thermotherapy: An overview", *International journal of hyperthermia : the official journal of European Society for Hyperthermic Oncology, North American Hyperthermia Group*, v. 29, 09 2013. doi: 10.3109/02656736.2013.832411.

[7] LEWIS, M., STARUCH, R., CHOPRA, R. "Thermometry and Ablation Monitoring with Ultrasound", *International Journal of Hyperthermia*, v. 31, pp. 1–19, 03 2015. doi: 10.3109/02656736.2015.1009180.

[8] ARTHUR, M., STRAUBE, W. "Theoretical estimation of the temperature dependence of backscattered ultrasonic power for noninvasive thermometry", *Ultrasound in Medicine & Biology*, v. 20, n. 9, pp. 915–922, 1994.

[9] ARTHUR, M., TROBAUGH, J., STRAUBE, W., et al. "Temperature dependence of ultrasonic backscattered energy in motion-compensated images", *IEEE Trans Ultrason Ferroelectr Freq Control*, 2005.

[10] SHASWARY, E., ASSI, H., YANG, C., et al. "Noninvasive calibrated tissue temperature estimation using backscattered energy of acoustic harmonics", *Ultrasonics*, v. 114, pp. 106406, 2021. ISSN: 0041-624X.

[11] RAJKOMAR, A., DEAN, J., KOHANE, I. "Machine learning in medicine", *New England Journal of Medicine*, v. 380, n. 14, pp. 1347–1358, 2019.

[12] GIGER, M. L. "Machine Learning in Medical Imaging", *Journal of the American College of Radiology*, v. 15, n. 3, Part B, pp. 512–520, 2018. ISSN: 1546-1440. Data Science: Big Data Machine Learning and Artificial Intelligence.

[13] LIU, S., WANG, Y., YANG, X., et al. "Deep Learning in Medical Ultrasound Analysis: A Review", *Engineering*, v. 5, n. 2, pp. 261–275, 2019. ISSN: 2095-8099.

[14] MARACI, M., BRIDGE, C., NAPOLITANO, R., et al. "A framework for analysis of linear ultrasound videos to detect fetal presentation and heartbeat", *Medical Image Analysis*, v. 37, pp. 22–36, 2017. ISSN: 1361-8415.

[15] ZHANG, J., HE, Q., XIAO, Y., et al. "Ultrasound image reconstruction from plane wave radio-frequency data by self-supervised deep neural network", *Medical Image Analysis*, v. 70, pp. 102018, 2021. ISSN: 1361-8415.

[16] LUSQUINO FILHO, L. A. D., FRANÇA, F. M. G., LIMA, P. M. V. "Near-optimal facial emotion classification using WiSARD-based weightless system". In: *Proceedings of the 26th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 85–90, 2018.

[17] ALEKSANDER, I., THOMAS, W., BOWDEN, P. "WISARD - a radical step forward in image recognition", *Sensor Review*, v. 4, n. 3, pp. 120–124, 03 1984.

[18] STANSFIELD, D. "Underwater Electroacoustic Transducers". Bath, UK, Bath University Press, 1991.

[19] SHOH, A. "Industrial Applications of Ultrasound – A Review", *High Power Ultrasound*, v. 22, 1975.

[20] NAKANO, R., ITO, A., TOKUMARU, S. "Sustainable pest control inspired by prey–predator ultrasound interactions", *Proceedings of the National Academy of Sciences*, v. 119, n. 43, pp. e2211007119, 2022.

[21] BANI, D., CALOSI, L., FAGGIOLI, L. "Effects of highfrequency ultrasound treatment on human skin tissues", *Surg Cosmet Dermatol*, v. 6, 2014.

[22] EDLER, I., LINDSTRÖM, K. "The history of echocardiography", *Ultrasound in medicine & biology*, v. 30, n. 12, pp. 1565–1644, 2004.

[23] ROZYCKI, G. S. "Abdominal ultrasonography in trauma", *Surgical Clinics of North America*, v. 75, n. 2, pp. 175–191, 1995.

[24] MCDICKEN, W. N., ANDERSON, T. "CHAPTER 1 – Basic physics of medical ultrasound". 2011.

[25] SZABO, T. L., LEWIN, P. A. "Ultrasound Transducer Selection in Clinical Imaging Practice", *Journal of Ultrasound in Medicine*, v. 32, n. 4, pp. 573–582, 2013.

[26] HILDEBRANDT, B., WUST, P., AHLERS, O., et al. "The cellular and molecular basis of hyperthermia", *Critical Reviews in Oncology/Hematology*, v. 43, n. 1, pp. 33–56, 2002. ISSN: 1040-8428.

[27] COUSSIOS, C., FARNY, C., HAAR, G. T., et al. "Role of acoustic cavitation in the delivery and monitoring of cancer treatment by high-intensity focused ultrasound (HIFU)", *International Journal of Hyperthermia*, v. 23, n. 2, pp. 105–120, 2007.

[28] DIEDERICH, C. J., HYNYNEN, K. "Ultrasound technology for hyperthermia", *Ultrasound in Medicine Biology*, v. 25, n. 6, pp. 871–887, 1999. ISSN: 0301-5629.

[29] EBBINI, E., SIMON, C., LIU, D. "Real-time Ultrasound Thermography and Thermometry", *IEEE Signal Process Mag*, 2018.

[30] TEIXEIRA, C., ALVARENGA, A., VON KRÜGER, M., et al. "Is the Average Gray-Level from Ultrasound B-Mode Images Able to Estimate Temperature Variations in Ex-Vivo Tissue?" In: Bamidis, P. D., Pallikarakis, N. (Eds.), *XII Mediterranean Conference on Medical and Biological Engineering and Computing 2010*, pp. 315–318. Springer Berlin Heidelberg, 2010.

[31] WUST, P., CHO, C. H., HILDEBRANDT, B., et al. "Thermal monitoring: Invasive, minimal-invasive and non-invasive approaches", *International Journal of Hyperthermia*, v. 22, n. 3, pp. 255–262, jan 2006. doi: 10.1080/02656730600661149.

[32] STRAUBE, W., ARTHUR, M. "Theoretical estimation of the temperature dependence of backscattered ultrasonic power for noninvasive thermometry", *Ultrasound in Medicine & Biology*, v. 20, n. 9, pp. 915–922, jan 1994. doi: 10.1016/0301-5629(94)90051-5.

[33] MILLER, N. R., BAMBER, J. C., MEANEY, P. M. "Fundamental limitations of noninvasive temperature imaging by means of ultrasound echo strain estimation", *Ultrasound in Medicine &amp Biology*, v. 28, n. 10, pp. 1319–1333, oct 2002.

[34] ALVARENGA, A. V., TEIXEIRA, C. A. D., VON KRUGER, M., et al. "Non-invasive assessment of temperature variation in Ex-vivo renal tissue by tracking average grey-level from B-mode images". In: *2012 IEEE International Ultrasonics Symposium*. IEEE, oct 2012. doi: 10.1109/ultsym.2012.0419. Disponível em: <https://doi.org/10.1109%2Fultsym.2012.0419>.

[35] TEIXEIRA, C., ALVARENGA, A., CORTELA, G., et al. "Feasibility of non-invasive temperature estimation by the assessment of the average gray-level content of B-mode images", *Ultrasonics*, v. 54, n. 6, pp. 1692–1702, aug 2014. doi: 10.1016/j.ultras.2014.02.021. Disponível em: <https://doi.org/10.1016%2Fj.ultras.2014.02.021>.

[36] ALVARENGA, A., TEIXEIRA, C. A., VON KRÜGER, M., et al. "Uncertainty evaluation from non-invasive estimation of temperature variation using B-mode ultrasonic images from a plastic phantom", *Measurement*, v. 69, pp. 189–194, jun 2015. doi: 10.1016/j.measurement.2015.03.038. Disponível em: <https://doi.org/10.1016%2Fj.measurement.2015.03.038>.

[37] ALVARENGA, A., WILKENS, V., GEORG, O., et al. "Non-invasive Estimation of Temperature during Physiotherapeutic Ultrasound Application Using the Average Gray-Level Content of B-Mode Images: A Metrological Approach", *Ultrasound in Medicine & Biology*, v. 43, n. 9, pp. 1938–1952, sep 2017. doi: 10.1016/j.ultrasmedbio.2017.04.008. Disponível em: <https://doi.org/10.1016%2Fj.ultrasmedbio.2017.04.008>.

[38] TEIXEIRA, C., PASTRANA-CHALCO, M., SIMÕES, R., et al. "On the Feasibility of Ultrasound Imaging Enrichment by Medium-Temperature Changes", *Ultrasonic Imaging*, v. 41, n. 1, pp. 17–34, 2019.

[39] ROKACH, L., MAIMON, O. "Decision Trees". v. 6, pp. 165–192, 01 2005. doi: 10.1007/0-387-25465-X_9.

[40] SALZBERG, S. L. "C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993", *Machine Learning*, v. 16, n. 3, pp. 235–240, sep 1994. doi: 10.1007/bf00993309.

[41] BREIMAN, L. *Classification and regression trees*. Chapman & Hall, 1993. ISBN: 978-0-412-04841-8.

[42] YE, J., CHOW, J.-H., CHEN, J., et al. "Stochastic gradient boosted distributed decision trees". In: *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 2061–2064, 2009.

[43] SCHAPIRE, R. E. "The strength of weak learnability", *Machine Learning*, v. 5(2), pp. 197–227, 1990.

[44] HIJAZI, S., KUMAR, R., ROWEN, C., et al. "Using convolutional neural networks for image recognition", *Cadence Design Systems Inc.: San Jose, CA, USA*, v. 9, 2015.

[45] LING, Z.-H., KANG, S.-Y., ZEN, H., et al. "Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends", *IEEE Signal Processing Magazine*, v. 32, n. 3, pp. 35–52, 2015.

[46] ABDEL-HAMID, O., MOHAMED, A.-R., JIANG, H., et al. "Convolutional neural networks for speech recognition", *IEEE/ACM Transactions on audio, speech, and language processing*, v. 22, n. 10, pp. 1533–1545, 2014.

[47] BISHOP, C. M. "Neural networks and their applications", *Review of scientific instruments*, v. 65, n. 6, pp. 1803–1832, 1994.

[48] IOFFE, S., SZEGEDY, C. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", *CoRR*, v. abs/1502.03167, 2015. Disponível em: <`http://arxiv.org/abs/1502.03167`>.

[49] HINTON, G., SRIVASTAVA, N., KRIZHEVSKY, A., et al. "Improving neural networks by preventing co-adaptation of feature detectors", *arXiv preprint*, v. arXiv, 07 2012.

[50] NEWELL, A. "Perceptrons. An Introduction to Computational Geometry", *Science*, v. 165, n. 3895, pp. 780–782, 1969. ISSN: 0036-8075.

[51] BLEDSOE, W. W., BROWNING, I. "Pattern Recognition and Reading by Machine". In: *Eastern Joint IRE-AIEE-ACM Computer Conference*, p. 225–232, New York, NY, USA, 1959.

[52] CARVALHO, D., CARNEIRO, H., FRANÇA, F., et al. "B-bleaching: Agile Overtraining Avoidance in the WiSARD Weightless Neural Classifier". In: *ESANN*, 2013.

[53] BENESTY, J., CHEN, J., HUANG, Y., et al. "Pearson Correlation Coefficient". In: *Noise Reduction in Speech Processing*, pp. 1–4, Berlin, Heidelberg, Springer Berlin Heidelberg, 2009. ISBN: 978-3-642-00296-0. doi: 10.1007/978-3-642-00296-0_5.

[54] BACELLAR, A., SUSSKIND, Z., VILLON, L., et al. "Distributive Thermometer: A New Unary Encoding for Weightless Neural Networks". pp. 31–36, 01 2022.

[55] CARDOSO, D. O., CARVALHO, D., ALVES, D. S. F., et al. "Financial credit analysis via a clustering weightless neural classifier", *Neurocomputing*, v. 183, pp. 70–78, 2016.

[56] CARDOSO, D. O., LIMA, P. M. V., DE GREGORIO, M., et al. "Clustering data streams with weightless neural networks". In: *ESANN 2011, 19th European Symposium on Artificial Neural Networks*, pp. 201 – 206, Bruges, Belgium, 2011.

[57] CARDOSO, D. O., DE GREGORIO, M., LIMA, P. M. V., et al. "A Weightless Neural Network-Based Approach for Stream Data Clustering". In: *Intelligent Data Engineering and Automated Learning - IDEAL 2012 - 13th International Conference*, pp. 328—-335, Natal, Brazil, 2012.

[58] CARNEIRO, H. C. C., FRANÇA, F. M. G., LIMA, P. M. V. "Multilingual part-of-speech tagging with weightless neural networks", *Neural Networks*, v. 66, pp. 11–21, 2015.

[59] CARNEIRO, H. C. C., PEDREIRA, C. E., FRANÇA, F. M. G., et al. "A universal multilingual weightless neural network tagger via quantitative linguistics", *Neural Networks*, v. 91, pp. 85–101, 2017.

[60] MIRANDA, I. D., ARORA, A., SUSSKIND, Z., et al. "LogicWiSARD: Memoryless Synthesis of Weightless Neural Networks". In: *2022 IEEE 33rd*

International Conference on Application-specific Systems, Architectures and Processors (ASAP), pp. 19–26, 2022.

[61] SUSSKIND, Z., ARORA, A., MIRANDA, I. D. D. S., et al. "Weightless Neural Networks for Efficient Edge Inference". 2022. Disponível em: <`https://arxiv.org/abs/2203.01479`>.

[62] NASCIMENTO, D., DE CARVALHO, R., MORA-CAMINO, et al. "A WiSARD-based multi-term memory framework for online tracking of objects". In: *Proceedings of the 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 19–24, 2015.

[63] DE SOUZA, D., FRANÇA, F., LIMA, P. "Real-time music tracking based on a weightless neural network". In: *Proceedings of the 2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems*, 2015.

[64] LUSQUINO FILHO, L., GUARISA, G., LIMA FILHO, A., et al. "Classifying Actions Units with ClusWiSARD". In: *Proceedings of the 28th International Conference on Artificial Neural Networks*, 2019.

[65] LUSQUINO FILHO, L. *Extending Wisard To Perform Ensemble Learning, Regression, Multi-Label and Multi-Modal Tasks*. Tese de Doutorado, PESC/COPPE/UFRJ, 2021.

[66] LUSQUINO FILHO, L., OLIVEIRA, L., LIMA FILHO, A., et al. "Prediction of palm oil production with an enhanced n-tuple regression network". In: *Proceedings of the 27th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 301–306, 2019.

[67] KOLCZ, A., N., A. "n-tuple Regression Network", *Neural Networks*, v. 9, pp. 855–869, 1996.

[68] SPECHT, D. "Probabilistic neural networks", *Neural Networks*, v. 3, n. 1, pp. 109–118, 1990. ISSN: 0893-6080.

[69] HAND, D. "Kernel discriminant analysis", *Chinchester: Research Study Press*, 1982.

[70] NADARAYA, E. "On Estimating Regression", *Theory of Probability & Its Applications*, v. 9, n. 1, pp. 141–142, 1964.

[71] DUA, D., GRAFF, C. "UCI Machine Learning Repository". 2017. Disponível em: <http://archive.ics.uci.edu/ml>. Acessado em 10/12/2023.

[72] TREEBY, B., COX, B. "k-Wave: MATLAB toolbox for the simulation and reconstruction of photoacoustic wave fields", *J Biomed Opt.*, 2010.

[73] RIGUEIRA, J., PASTRANA-CHALCO, M., TEIXEIRA, C., et al. "Evaluation of the Gray Scale Intensity Variation of Ultrasonic Images from Porcine Muscle Tissue as a Function of Temperature". In: *2019 Global Medical Engineering Physics Exchanges/ Pan American Health Care Exchanges (GMEPE/PAHCE)*, pp. 1–4, 2019. doi: 10.1109/GMEPE-PAHCE.2019.8717340.

[74] PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., et al. "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, v. 12, pp. 2825–2830, 2011.

[75] CHEN, T., GUESTRIN, C. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16*, pp. 785–794, 2016.

[76] SHWARTZ-ZIV, R., ARMON, A. "Tabular Data: Deep Learning is Not All You Need", *CoRR*, v. abs/2106.03253, 2021. Disponível em: <https://arxiv.org/abs/2106.03253>.

[77] HUANG, X., KHETAN, A., CVITKOVIC, M., et al. "TabTransformer: Tabular Data Modeling Using Contextual Embeddings", *CoRR*, v. abs/2012.06678, 2020. Disponível em: <https://arxiv.org/abs/2012.06678>.

[78] SUK, H.-I., LEE, S.-W., SHEN, D. "Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis", *NeuroImage*, 2014.

[79] ISLAM, M. A., JIA, S., BRUCE, N. D. B. "How Much Position Information Do Convolutional Neural Networks Encode?" *CoRR*, v. abs/2001.08248, 2020. Disponível em: <https://arxiv.org/abs/2001.08248>.

[80] MYERS, L., SIROIS, M. "Spearman Correlation Coefficients, Differences between". In: *Encyclopedia of Statistical Sciences*, American Cancer Society, 2004. ISBN: 9780471667193. doi: https://doi.org/10.1002/0471667196.ess5050.

Appendix A

Publications

Following is the list of papers accepted for publication during the development of this work:

A.1 Journal articles

1. LUSQUINO FILHO, L. A. D.; OLIVEIRA, L. F. R. ; LIMA FILHO, A. ; GUARISA, G. ; FELIX, L. M. ; LIMA, P. M. V. ; FRANÇA, F. M. G. . Extending the Weightless WiSARD Classifier for Regression. NEUROCOMPUTING, 2020.

2. OLIVEIRA, L. F. R; FRANÇA, F. M. G.; PEREIRA, W. C. A., *A data-driven approach for estimating temperature variations based on B-mode Ultrasound Images and Changes in Backscattered Energy.* Ultrasonic Imaging, 2023.

A.2 Book chapters

1. LUSQUINO FILHO, L. A. D.; OLIVEIRA, L. F. R.; CARNEIRO, H. C. C., GUARISA G. P.; LIMA FILHO, A. S.; FRANÇA, F. M. G. ; LIMA, P. M. V., A Weightless Neural System for Empathy Prediction - Accepted for OMG-Challenges Book, Knowledge Technology Group, Springer, 2020.

A.3 Complete works published in proceedings of conferences

1. LIMA FILHO, A. ; GUARISA, G. ; LUSQUINO FILHO, L. A. D. ; OLIVEIRA, L. F. R. ; COSENZA, C. ; FRANÇA, F. M. G. ; LIMA, P. M. V. . Interpretation of Model Agnostic Classifiers via Local Mental Images. In:

European Symposium on Artificial Neural Networks, 2020, Brugge. Proc. of ESANN 2020, 2020.

2. LUSQUINO FILHO, L. A. D.; OLIVEIRA, L. F. R. ; LIMA FILHO, A. ; GUARISA, G. ; LIMA, P. M. V. ; FRANÇA, F. M. G. . Prediction of palm oil production with an enhanced n-Tuple Regression Network. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 2019, Bruges. Proc of ESANN 2019, 2019. v. 27. p. 301-306.

3. LUSQUINO FILHO, L. A. D.; GUARISA, G. ; OLIVEIRA, L. F. R. ; LIMA FILHO, A. ; FRANÇA, F. M. G. ; LIMA, P. M. V. Action Units Classification Using ClusWiSARD. In: International Conference on Artificial Neural Networks, 2019, Munich. Artificial Neural Networks and Machine Learning, ICANN 2019: Image Processing, 2019. p. 409-420.

A.4 Extended abstracts published in proceedings of conferences

1. LUSQUINO FILHO, L. A. D.; OLIVEIRA, L. F. R.; CARNEIRO, H. C. C., GUARISA G. P.; LIMA FILHO, A. S.; FRANÇA, F. M. G.; LIMA, P. M. V., *A weightless regression system for predicting multi-modal empathy*, Workshop Affective Behavior Analysis in-the-wild, Proceedings of the 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020), Buenos Aires, Argentina, 2020.

Appendix B

Weightless Neural Networks

This appendix explores weightless neural networks by presenting an introduction to this type of neural network architecture and its main concepts, resourcing mainly to the n-tuple networks WiSARD and ClusWiSAD, which serve as starting points for the regression methods that were presented in chapter 3. After each model's introduction, a brief review of previous works using these models demonstrates their wide range of applications and evolution through time.

B.1 WiSARD

WiSARD stands for ***Wi****lkie,* ***S****tonham and* ***A****leksander* ***R****ecognition* ***D****evice* and it's the first commercial product based on weightless neural networks and applied for handwritten digits recognition [17]. Its architecture is based on multiple discriminators, where each discriminator is responsible for recognizing a single class, as shown in figure B.1. The internal nodes are RAM-like structures, i.e. a set of pairs $< key, value >$ where the key is a binary address and the value is a boolean flag that represents whether the address was accessed or not.

When the network is first initialized, all addresses of the RAM neurons are set to 0. Given an input, observation to be learned in the shape of a binary word and a label, a pseudo-random mapping is applied to the binary word, which creates a new order of bits. The new entry is then divided into n segments (tuples) of length m. Each tuple is then addressed to a neuron of the discriminator that relates to the given label. The value related to each tuple is, then, set to 1. This training phase is illustrated by figure B.2.

Given an observation to be classified, the same pseudo-random mapping is applied and the input is divided into n tuples. However, differently from the training phase where a single discriminator is used, in this phase, all discriminators must be accessed. Each address formed by the tuples is verified if the corresponding neurons contain the value *1*. If it does, the neuron indicates its activation. Each discrim-

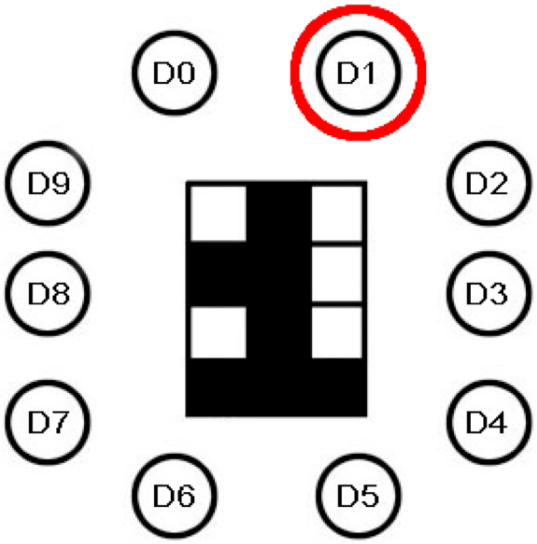

Figure B.1: Example of discriminators in a digits classification scenario. A total of 10 discriminators, each one responsible for recognizing a single digit. In the training phase, the input is addressed to the corresponding discriminator.

inator returns the number of neurons that were activated and this value is called *activation degree*. The class chosen by the network is the one that has the largest activation degree. A random one is selected if there is a tie between two or more discriminators.

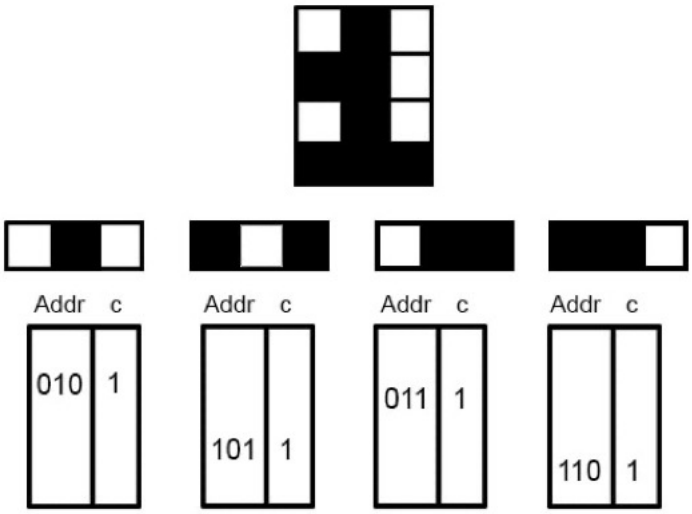

Figure B.2: Training phase of the WiSARD model. The pseudo-random mapping creates a new order of the input bits and the new input is divided according to the defined address size. The counter value c of the corresponding address in each RAM node registers the activation.

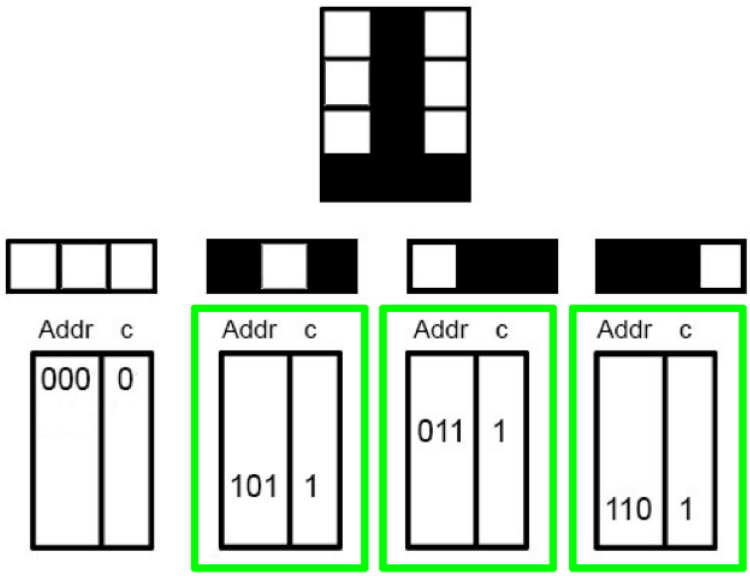

Figure B.3: Classification phase of the WiSARD model. The same pseudo-random mapping used in training creates a new order of the input bits and the new input is divided according to the defined address size. For each RAM node, the corresponding address is verified and, if the memory address was accessed, the node is considered activated and the discriminator reports the number of active nodes.

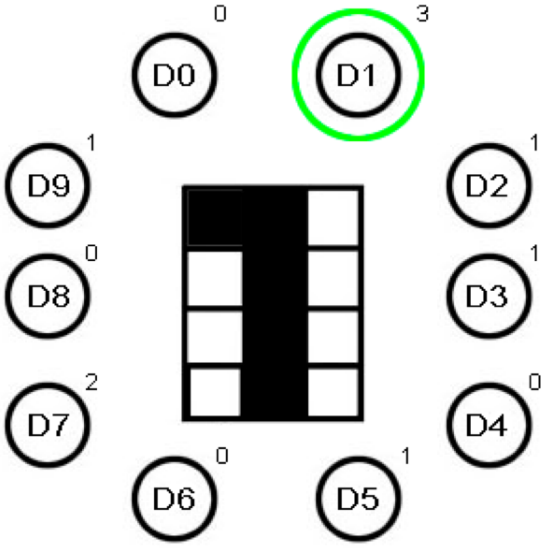

Figure B.4: To complete the classification of the presented pattern, the WiSARD model compares the response of all discriminators. The one with the highest response defines the class.

B.1.1 Bleaching

A problem faced by the WiSARD model is known as saturation, which is when, after an excessive amount of training, many of the possible RAM addresses of the discriminators are filled up. This causes more than one discriminator to respond to the stimuli and leads to the model being in doubt as to which class the presented

pattern belongs to.

To work around this problem, the *bleaching* [52] technique was developed, which consists of changing how addresses work. Instead of recording the value 1, the address operates as a counter: once a sequence of bits is displayed to an address, the previous value is incremented by 1. With that, a threshold is defined so that, given a tuple, only the RAM structures whose value stored in the given address is greater than or equal to the threshold are activated.

B.2 Binary Encoding

The WiSARD model requires binary inputs so it can apply its learning algorithm. Thus, a binarization process is required to deal with values representing integer and floating point values or text. Although a simple binary conversion of the respective value could be applied, it brings certain negative aspects. For example, consider the binary conversion of integer values 0, 1, 7 and 8 using a 4-bit word. The process would lead to the following:

- 0 : 0000
- 1 : 0001
- 7 : 0111
- 8 : 1000

As a matter of comparison, values 0 and 1 are close to each other both in the numeric perspective and in the binary representation. But the same does not hold when comparing the pairs 0 and 8, and 7 and 8. In the first case, the values are numerically distant, but the binary representation presents the same difference as the one from 0 and 1. On the other hand, the second case presents two values that are close to each other, but with a considerable difference in binary representation. In this sense, it is necessary to provide a binarization process that overcomes these problems.

B.2.1 Thermometer Encoding

The binary thermometer encoding generates binary words from numerical values. It works by identifying the lower and higher values of the variable to be converted. Then, by defining the total number of bits n, a set of n thresholds is created, given by a step that is computed using the formula $\frac{(max-min)}{n}$. Starting from the minimum value, the threshold increments by accumulating the step.

Figure B.5 illustrates the process of converting the value 4 into an 8-bit word considering a given set of data. Given a collection of values S and the desired number of bits $n = 8$, the procedure starts by computing the maximum (18) and minimum (-2) values. Then, the step is computed as $\frac{(18-(-2))}{8} = 2{,}5$. The threshold starts at the minimum value and the following ones are incremented by the step. For a given value to be converted, a binary word containing 8 zeros is generated. Then, all bits are flipped at the indexes where the value is higher than the specified threshold. In this example, the value 4 is converted to 11100000.

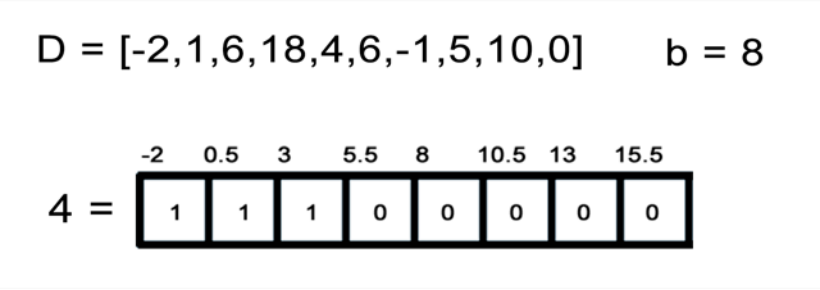

Figure B.5: Illustrative example of the linear thermometer.

Defining the number of bits n is a critical step. If too small, a lot of information regarding the feature can be lost. On the other hand, long binary words may affect the models' performance. Besides, a dataset is usually formed by a group of different features, with different distributions and ranges of values. A naïve approach that could be used is to use the same value for all features, although that could harm the model performance. Since there is a difference between the relevance of features with respect to the expected output, this difference should be reflected in the number of bits used to address the features: the most relevant ones should be addressed with a wider number of bits, while the less relevant features should be addressed with fewer bits. In this sense, a heuristic for defining the number of bits n is proposed.

In the scope of data preprocessing, feature engineering takes a fundamental role in the machine learning pipeline. The majority of the different techniques require some sort of preparation for the input data so that a model can either be more accurate or faster. Among the different steps that are applied during preprocessing, one, in particular, provides the tools for developing the proposed heuristic: choosing features.

As stated earlier, features in the dataset present different levels of importance when compared to the expected output or even with other features. This importance can be measured with values called *correlation coefficients*, which are statistical measures that show how two variables are related. It is usually a value that ranges from -1 to 1, where ± 1 indicates a full correlation and 0 indicates no correlation at all.

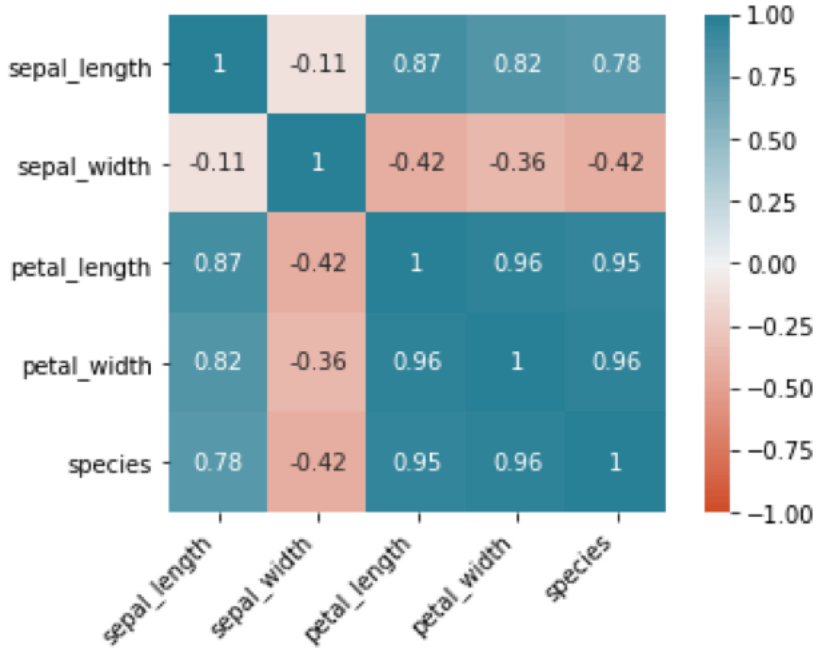

Figure B.6: Correlation matrix for the Iris dataset.

One of the most used correlation coefficients is Pearson's r, also referred to as *Pearson product-moment correlation coefficient* [53]. Developed by the English mathematician Karl Pearson, it measures the linear relationship between two variables. For a given sample, the correlation coefficient is computed as described by equation B.1.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=0}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=0}^{n}(y_i - \bar{y})^2}} \tag{B.1}$$

The usage of correlation coefficients usually appears in the form of correlation matrices associated with a heat-map, which allows visual perception of the relationship's behavior of the variables. A correlation matrix M is a symmetrical matrix where rows and columns represent one of the dataset's features and each index $m_{i,j}$ represents the correlation coefficient between features i and j. The main diagonal of a correlation matrix is always 1, since it represents the correlation between the feature and itself.

In Appendix **??**, a series of experiments are performed comparing this heuristic with the proposed ones. The results show that the heuristic method can improve the models' performance in various scenarios. Although this cannot be considered in all cases, using this method can be seen as a good starting point.

B.2.2 Distributive Thermometer

One characteristic of thermometer encoding is the fact that it assumes a certain distribution of data. In particular, the linear thermometer encoding assumes the data is uniformly distributed. Given that this may not hold, authors in [54] proposed a novel thermometer technique that aims to make the binarization independent from the distribution.

This is done by changing how the thresholds are computed. Instead of splitting into n intervals of the same size, the divisions are done based on percentiles of the data. This method is useful in the sense that it better covers regions where there is a higher concentration of data. In the conventional linear thermometer, points in the same region are represented by the same binary word. Figure B.7 illustrates the difference between the two thermometer techniques.

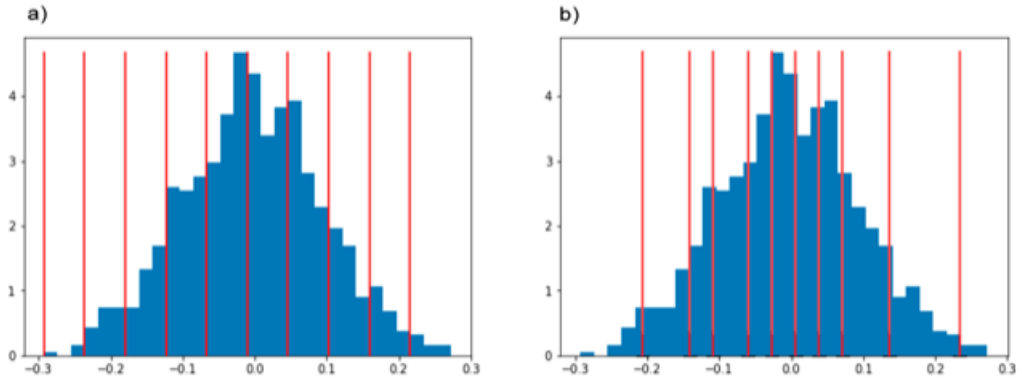

Figure B.7: Comparison between the thresholds generated by a) the conventional linear thermometer and b) the distributive thermometer.

B.3 ClusWiSARD

A common problem that affects classification models is when observations of the same class present different characteristics (not in the sense of being an outlier). That is particularly noticed in classification performance when the observation has never been presented to the model. To overcome this situation, an extension of the WiSARD model is proposed in [55], named ClusWiSARD.

As an extension, ClusWiSARD also operates with discriminators and RAM-based nodes with an access counter. The main difference is in the fact that the model is capable of creating new discriminators related to the same class if the presented training observation is not similar enough to the knowledge stored within its current discriminator, which acts like a clustering system. This model can handle both supervised and unsupervised learning tasks.

The training phase of the ClusWiSARD model, then, works slightly similar to the one in WiSARD, where the observation is presented to its corresponding discriminator. But in this case, there is a group of discriminators. Since the pattern will only be addressed to one discriminator, before the training happens, each discriminator evaluates the pattern and returns a score r. To learn the pattern, a discriminator must satisfy:

$$r \geq \min\left(N, r_0 + \frac{N|d|}{\gamma}\right), \tag{B.2}$$

where r is the score of the discriminator when classifying the observation, $|d|$ is the discriminator size, N is its number of RAMs, r_0 is a threshold, which indicates the minimum response expected by a discriminator, and γ is also a threshold, which indicates the growth interval, that is, the speed that the discriminators increase their size.

The classification phase is similar to the one in the WiSARD model, and the pattern is presented to all discriminators. If there is a tie between discriminators of different classes, the bleaching procedure is applied. On the other hand, in case of ties between discriminators of the same class, there is no need for applying bleaching, since the class is already decided.

B.4 Applications of WiSARD-based Models

WiSARD-based models have been successfully reported in different application domains, dealing with both structured and unstructured data. In [55] the authors apply the ClusWiSARD model to a credit analysis problem using two reference datasets and comparing the models' performance with the Support Vector Machine (SVM) model. Results demonstrated that the weightless network was highly competitive when considering classification accuracy while surpassing the SVM model by two orders of magnitude in training time.

Authors in [56] and [57] explore the context of clustering streams of data, which is a task that requires a model that can perform on scenarios of online data acquisition. Another extension of the WiSARD model is presented in this work called StreamWiSARD. The model was able to perform faster and with a higher quality than two classic stream clustering models.

In a text-processing environment, authors in [58] and [59] developed a multilingual part-of-speech (POS) tagging system, where the goal is to, given a sentence or text, define how each word acts in the sentence based on its definition and context, i.e. if its a noun, a verb, adjective, etc. The main concern in this problem is that the training process involved is highly intense and time-consuming. Thus, based on the

WiSARD models' efficiency it was possible to perform hyperparameters' calibration for different languages and either surpassed or matched state-of-the-art results.

As previously stated, due to its simplicity and architecture, WiSARD-based models are easy to implement as hardware and can operate as efficient edge devices. In [60] the authors propose a new model named LogicWiSARD, that incorporates a new dimension of operations to the traditional WiSARD architecture, as it generates a mapping of the RAM nodes into logic functions. The model is then implemented using an FPGA - an integrated circuit that can be programmed - and evaluates facing an MLP model, showing that the weightless architecture consumes up to 80% less energy, also surpassing versions that implement DNN models. Authors in [61] explore a new weightless architecture

Regarding image and video processing, WiSARD-based models were also applied. In [62] the authors developed a face tracking system, where the main objective is to enable a piece of software through the use of a camera to track the movement of a person's face in real-time. [63] also uses the WiSARD model to perform real-time music tracking. Both works point to the capacity of the weightless model to generate a fast classification response.

Other works related to image and video processing are presented in [16] and [64], where the authors seek to develop an automatic emotion recognition system based on the FACS system, in which specific combinations of muscle activation - known as action units - are used to determine human emotions according to facial expressions. In this work, a WiSARD-based architecture was able to reach state-of-the-art results in the proposed comparative experiments.

B.5 Weightless Models for Regression

This chapter presents the two weightless-based regression models developed: RegresionWiSARD and ClusRegressionWiSARD. First, a brief review of the n-tuple regression is presented, followed by a description of the algorithms of both models. Then, the results of performance comparisons of both models are shown. It is worth noticing that these models are also presented in [65], as they were developed collaboratively in the original paper [66] that includes both authors as *first author*.

B.5.1 n-Tuple Regression Network

Regression is a classic machine learning task, where the goal is to generate a scalar real-valued output y given a D-dimensional input $\mathbf{x}$, which comes from the random variables Y and X, respectively. Performing this task implies at finding a relationship between the input and the output. It is usually assumed that X and Y are distributed

according to a continuous joint probability density function $f(\mathbf{x}, \mathrm{y})$, and the

The n-Tuple Regression Network [67] is an adaptation of the classic n-Tuple Classifier architecture that aims to implement a General Regression Neural Network (GRNN) [68], which is capable of approximating unknown system mappings through the estimation of its regression function using a set of training observations. Although the GRNN presents an implementation problem due to its need to store the entire training set for operating the network, the usage of the n-Tuple architecture was proved to avoid this issue since elements of the training set are intrinsically stored in the network nodes.

The method is developed based on the kernel method [69], which is a type of non-parametric regression and is applied when there is no explicit knowledge about the system. It does so by assigning a smooth monotonically decreasing function to every sample ($\mathbf{x}$,y) taken from the distribution. In particular, the GRNN the usage of the Nadaraya-Watson kernel regression estimator [70].

The authors consider a given input $\mathbf{x}$ to the network in terms of three components:

$$\mathbf{x} \rightarrow \begin{cases} \{t_1(x), t_2(x), ..., t_k(x)\} \\ \{w_1(x), w_2(x), ..., w_k(x)\} \\ \{a_1(x), a_2(x), ..., a_k(x)\} \end{cases} \tag{B.3}$$

where $t_i(x)$ is the address of the i-th RAM node formed by the i-th tuple from the binary representation of $\mathbf{x}$, $w_i(x)$ is the expected output value from the observation, and $a_i(x)$ is as access counter. Then, each node structure are updated according to the following increment rules:

$$\begin{aligned} w_k(x^i) &\leftarrow w_k(x^i) + y^i \\ a_k(x^i) &\leftarrow a_k(x^i) + 1 \end{aligned} \tag{B.4}$$

Then, the output of the network is computed by normalizing the sum of the addressed weights with the sum of their respective counter values.

$$\hat{y}(x) = \frac{\sum_{k=0}^{K} w_k(x)}{\sum_{k=0}^{K} a_k(x)} \tag{B.5}$$

B.5.2 Interpolation Capability Comparison

Initially, ReW and the original n-Tuple Regression network were compared. This was done by trying to replicate some of the experiments in [67]. The main comparison to be computed is the interpolation capability of different averages. For the sake of simplicity, it is considered that a ReW using the Simple Mean denotes the original

n-Tuple Regression Network while the other averages relate to the proposed models. First, a function $u(k)$ is defined as follows:

$$\mathrm{u(k)} = \begin{cases} \sin(\frac{2\times\pi\times k}{250}), & k \leq 500 \\ 0.8 \times \sin(\frac{2\times\pi\times k}{250}) + 0.2 \times \sin(\frac{2\times\pi\times k}{25}), & otherwise \end{cases}$$

The function is defined in the interval [0, 1000]. From this interval, a random sample is retrieved containing 1,000 pairs (x, y). This sample is used for training all combinations of models. After the training is finished, all models predict 10,000 pairs in a linear space inside the defined interval. From the prediction, the mean absolute error metric (MAE) is computed. This procedure is repeated 10 times, with the average of these results being drawn. Figure B.8 illustrates the predicted function.

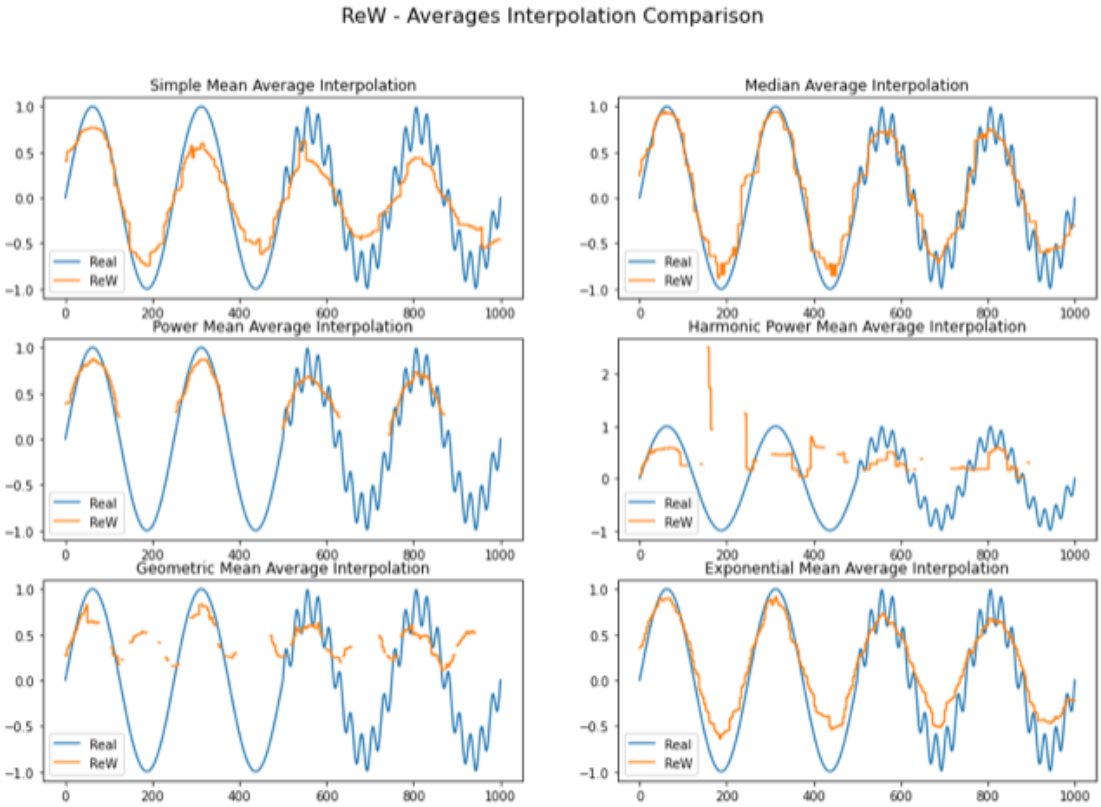

Figure B.8: ReW interpolation capability. The blue line indicates the real function u(k), while the orange line indicates the predicted values from ReW. From the results obtained, it is possible to observe limitations regarding three of the analysed averages as they are unable to generate a continuous line.

In the plots, it is possible to observe that three of the proposed averages are not able to interpolate the whole function. The parts of the lines that are not plotted denote *the do not know* answer from the model. Although the Power Mean presented a better performance compared to Harmonic Power and Geometric means, all of

these calculations shared the characteristic of not predicting values below 0. The other averages behaved quite similarly, and a more precise analysis of the methods can be done with the table results. By using the Median calculation, there was a slight improvement in the model performance, as shown in table B.1.

Model	**MAE**
Simple Mean	0.201
Median	**0.125**
Exponential Mean	0.225

Table B.1: Comparison of MAE values from ReW models while varying the type of average. It is possible to observe that Median appears as a better type of computing the output in contrast to the Simple Mean as suggested by the original n-tuple regressor for this particular task.

Appendix C

Results on Hyperparameter Exploration of the ReW Model

C.1 Models Evaluation

This section shows the computation results of a series of comparisons between the RegressionWiSARD model and a small collection of popular regression models described in Table C.1. The comparisons are made in terms of the Mean Absolute Error metric, and the training and prediction speed of the models, measured in seconds. To evaluate the models, four regression datasets from the UCI Machine Learning repository [71] were used, described as follows:

- Abalone: this dataset aims to predict the age of abalone given some physical attributes. It contains 8 attributes and 4177 observations, with the target variable in the range [1, 29];
- Auto Imports: using 25 features, this dataset aims to predict the price of an automobile. It has 201 observations and the target variable is in the range [5118, 35056];
- Auto Insurance: this dataset has 63 observations and a single feature to relate the number of insurance claims to the total payment. The target variable is within the range [0, 422.2];
- House Prices: one of the most used dataset for benchmarks in regression, it aims to predict the price of a house given a set of 13 characteristics of the building. The target variable is in the range [5, 50] among 506 observations.

For each dataset, a 10-fold cross-validation process was used, and the mean averages of the metrics are reported, alongside the standard deviations. For ReW and CReW, an exploration of the tuple size, output mean and thermometer size

Model	Alias	Parameters
K-nearest neighbors	knn	neighbors: 5
Decision Tree	dt	criterion: mae
Random Forest	rf	estimators: 100
Support Vector Machine	svr	kernel: RBF; C=1.0
Multi-layer Perceptron	mlp	learning_rate: 0.01; hidden_layer: (50, 50)
Gradient Boost	gb	learning_rate: 0.1; estimators: 1000 max_depth: 9
XGBoost	xgb	learning_rate: 0.1; estimators: 1000; max_depth: 9

Table C.1: Regression models configuration

was performed by varying the base thermometer value in the set [50,100,150,200], the tuple size in the set [2,5,10,15,20,25,31]. Numerical results for each dataset are reported in Tables C.2-C.5.

Model	MAE	Training Time	Prediction Time
knn	1.58 ± 0.50	$4.77e{-}03 \pm 1.78e{-}03$	$5.88e{-}03 \pm 2.31e{-}03$
dt	2.15 ± 0.54	$0.02 \pm 6.38e{-}03$	$3.51e{-}04 \pm 1.18e{-}04$
rf	1.56 ± 0.44	$0.83 \pm 7.15e{-}03$	$9.36e{-}03 \pm 3.43e{-}04$
svr	$\mathbf{1.51} \pm 0.53$	0.55 ± 0.04	$0.06 \pm 3.27e{-}03$
mlp	1.53 ± 0.41	5.01 ± 2.99	$3.62e{-}03 \pm 3.55e{-}03$
gb	1.66 ± 0.44	1.31 ± 0.06	$1.79e{-}03 \pm 9.29e{-}05$
xgb	1.63 ± 0.43	9.91 ± 10.13	$5.59e{-}03 \pm 2.00e{-}03$
ReW	1.75 ± 0.50	$8.82e{-}03 \pm 3.05e{-}04$	$1.25e{-}03 \pm 4.26e{-}05$
CReW	1.86 ± 0.86	$2.80e{-}03 \pm 8.70e{-}05$	$8.64e{-}04 \pm 4.39e{-}05$

Table C.2: Results on the Abalone dataset.

Model	MAE	Training Time	Prediction Time
knn	2065.72 ± 1031.71	$2.65e{-}04 \pm 3.49e{-}05$	$7.77e{-}04 \pm 1.32e{-}04$
dt	2303.06 ± 623.03	$2.75e{-}03 \pm 7.79e{-}04$	$1.61e{-}04 \pm 3.18e{-}05$
rf	1824.74 ± 629.77	$0.21 \pm 2.97e{-}03$	$6.09e{-}03 \pm 2.45e{-}04$
svr	4265.74 ± 2127.53	$1.28e{-}03 \pm 4.03e{-}05$	$2.04e{-}04 \pm 1.13e{-}05$
mlp	4029.47 ± 1062.94	0.11 ± 0.07	$2.41e{-}04 \pm 2.36e{-}05$
gb	1776.86 ± 674.64	$0.17 \pm 3.51e{-}03$	$2.90e{-}04 \pm 1.55e{-}05$
xgb	$\mathbf{1672.72} \pm 531.36$	1.02 ± 0.87	$3.76e{-}03 \pm 3.39e{-}03$
ReW	2096.12 ± 634.61	$2.90e{-}03 \pm 3.99e{-}05$	$3.52e{-}04 \pm 1.74e{-}05$
CReW	2166.03 ± 0.86	$9.88e{-}04 \pm 8.70e{-}05$	$2.52e{-}05 \pm 4.39e{-}05$

Table C.3: Results on the Auto Imports dataset.

Model	MAE	Training Time	Prediction Time
knn	36.51 ± 23.82	$2.55e{-}04 \pm 3.63e{-}05$	$4.82e{-}04 \pm 5.94e{-}05$
dt	41.90 ± 15.59	$2.36e{-}04 \pm 1.78e{-}05$	$6.92e{-}05 \pm 5.61e{-}06$
rf	39.74 ± 16.30	$0.08 \pm 3.10e{-}03$	$5.97e{-}03 \pm 2.17e{-}04$
svr	40.73 ± 26.80	$4.29e{-}04 \pm 3.04e{-}05$	$9.15e{-}05 \pm 7.60e{-}06$
mlp	37.99 ± 25.73	$0.02 \pm 4.50e{-}03$	$9.97e{-}05 \pm 4.25e{-}05$
gb	41.89 ± 15.60	$0.02 \pm 1.27e{-}03$	$1.91e{-}04 \pm 1.35e{-}05$
xgb	40.74 ± 17.22	1.38 ± 2.09	$5.35e{-}03 \pm 4.48e{-}03$
ReW	$\mathbf{34.88} \pm 19.58$	$2.97e{-}05 \pm 6.37e{-}06$	$1.36e{-}05 \pm 6.75e{-}07$
CReW	37.60 ± 0.86	$2.06e{-}05 \pm 8.70e{-}05$	$6.34e{-}04 \pm 4.39e{-}05$

Table C.4: Results on the Auto Insurance dataset.

Model	MAE	Training Time	Prediction Time
knn	6.01 ± 0.97	$4.86e{-}04 \pm 9.09e{-}05$	$9.45e{-}04 \pm 1.69e{-}04$
dt	3.67 ± 0.47	$2.57e{-}03 \pm 4.56e{-}04$	$1.17e{-}04 \pm 4.29e{-}05$
rf	2.99 ± 0.59	$0.18 \pm 9.10e{-}03$	$6.97e{-}03 \pm 2.83e{-}04$
svr	5.65 ± 1.18	$7.74e{-}03 \pm 1.91e{-}04$	$1.88e{-}03 \pm 1.16e{-}04$
mlp	4.53 ± 0.78	0.80 ± 0.59	$3.49e{-}04 \pm 2.92e{-}05$
gb	3.34 ± 0.67	$0.26 \pm 7.76e{-}03$	$7.32e{-}04 \pm 1.38e{-}04$
xgb	$\mathbf{2.82} \pm 0.44$	1.09 ± 0.82	$3.95e{-}03 \pm 3.72e{-}03$
ReW	4.08 ± 0.86	$1.60e{-}03 \pm 8.70e{-}05$	$4.35e{-}04 \pm 4.39e{-}05$
CReW	4.61 ± 0.86	$5.41e{-}03 \pm 8.70e{-}05$	$1.34e{-}04 \pm 4.39e{-}05$

Table C.5: Results on the House Prices dataset.

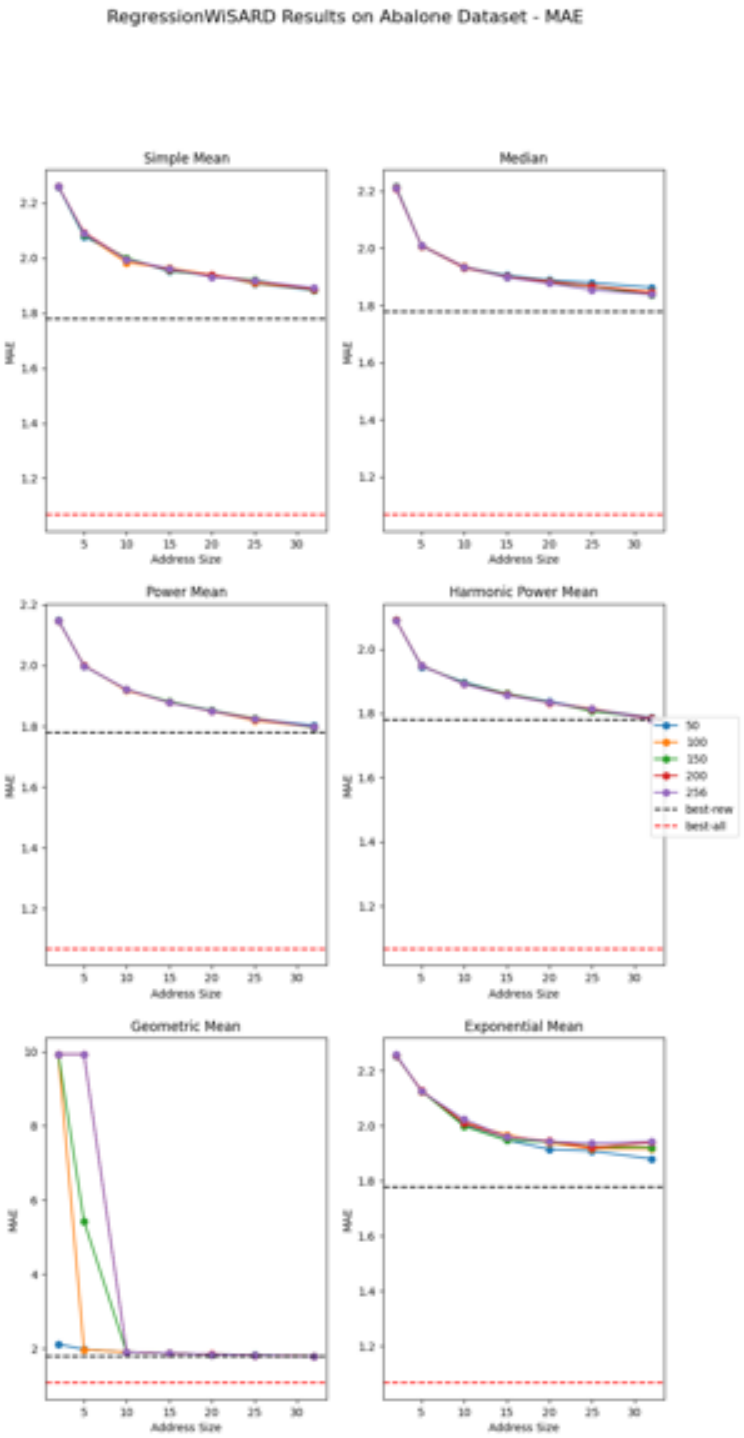

Figure C.1: MAE measures - Hyperparameter exploration on the Abalone Dataset.

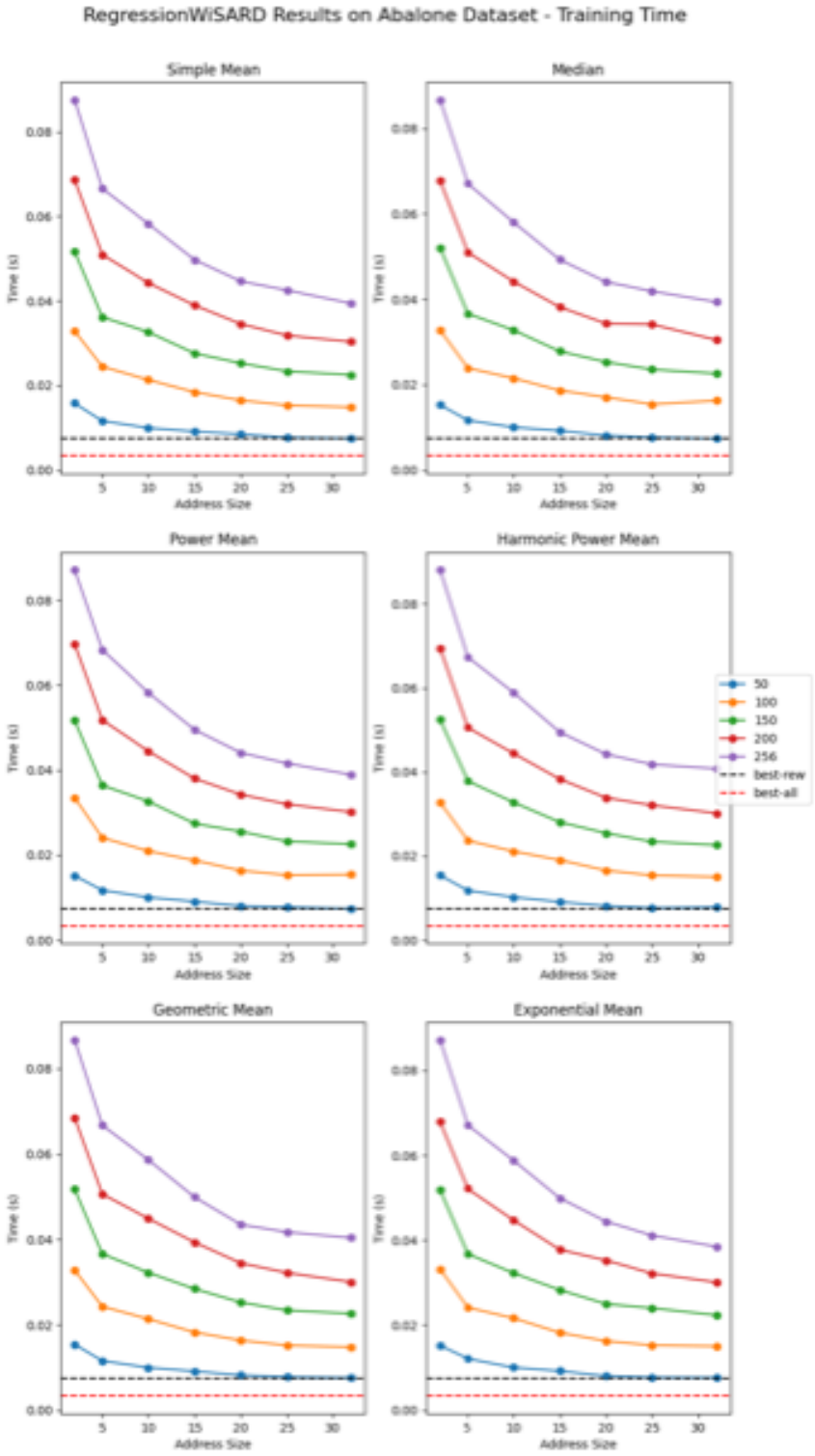

Figure C.2: Training time measures - Hyperparameter exploration on the Abalone Dataset.

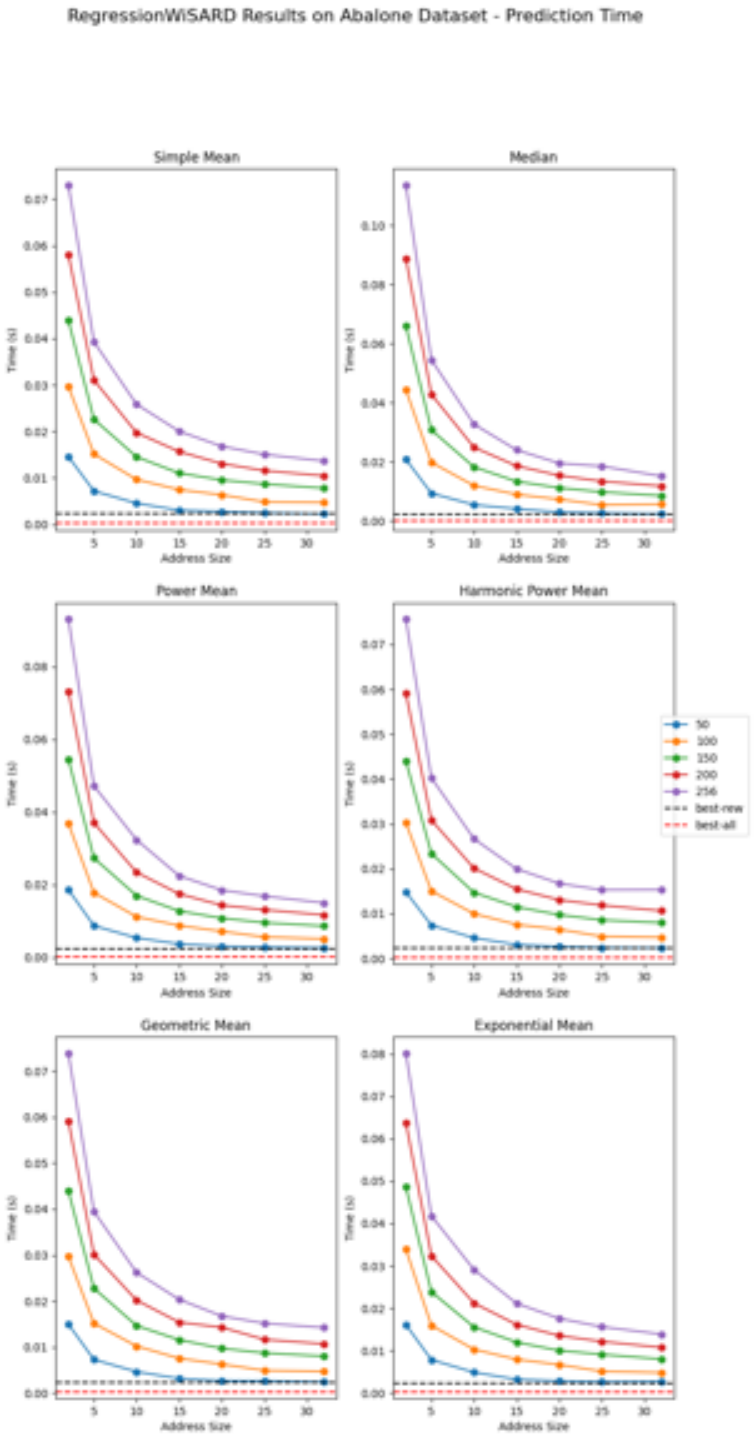

Figure C.3: Prediction time measures - Hyperparameter exploration on the Abalone Dataset.

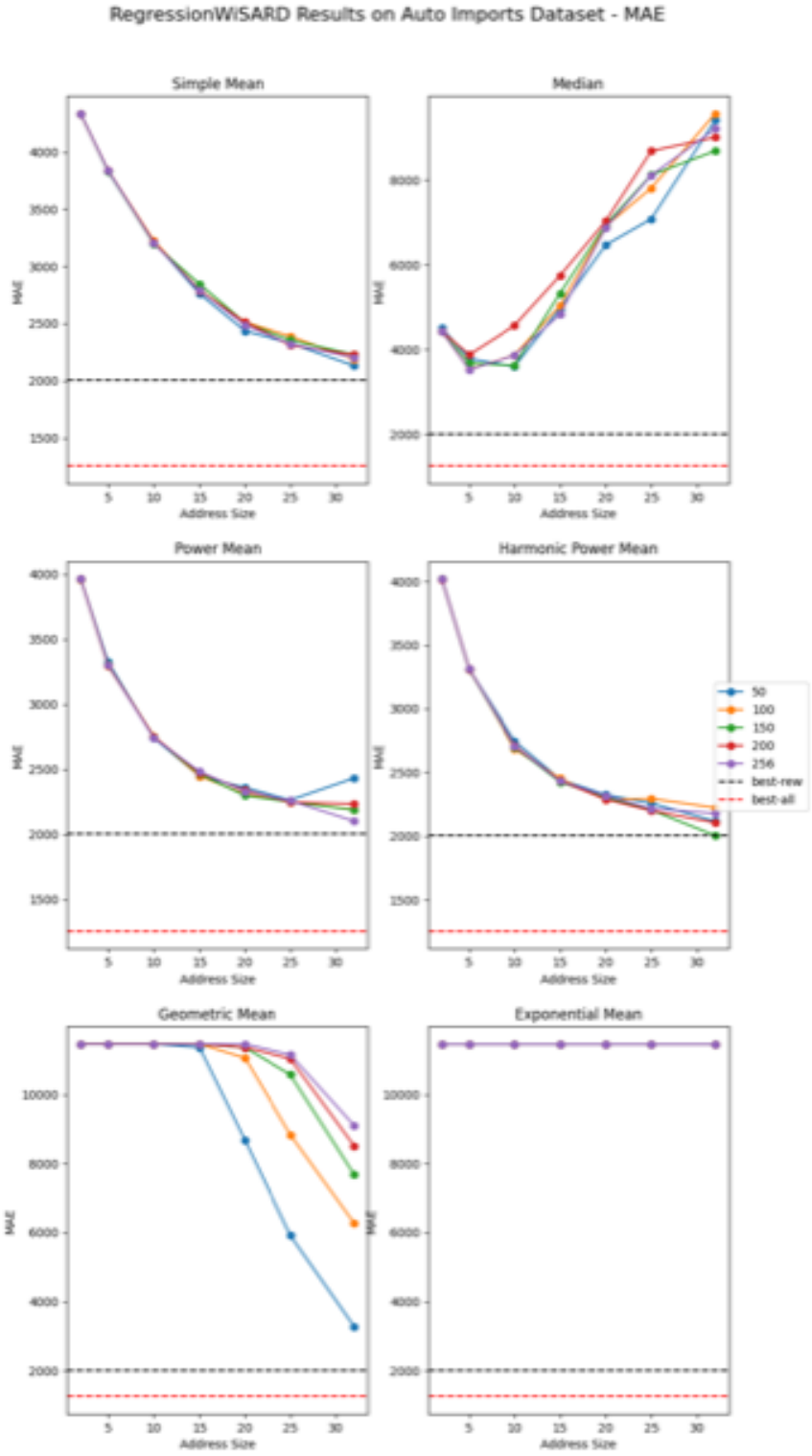

Figure C.4: MAE measures - Hyperparameter exploration on the Auto Imports Dataset.

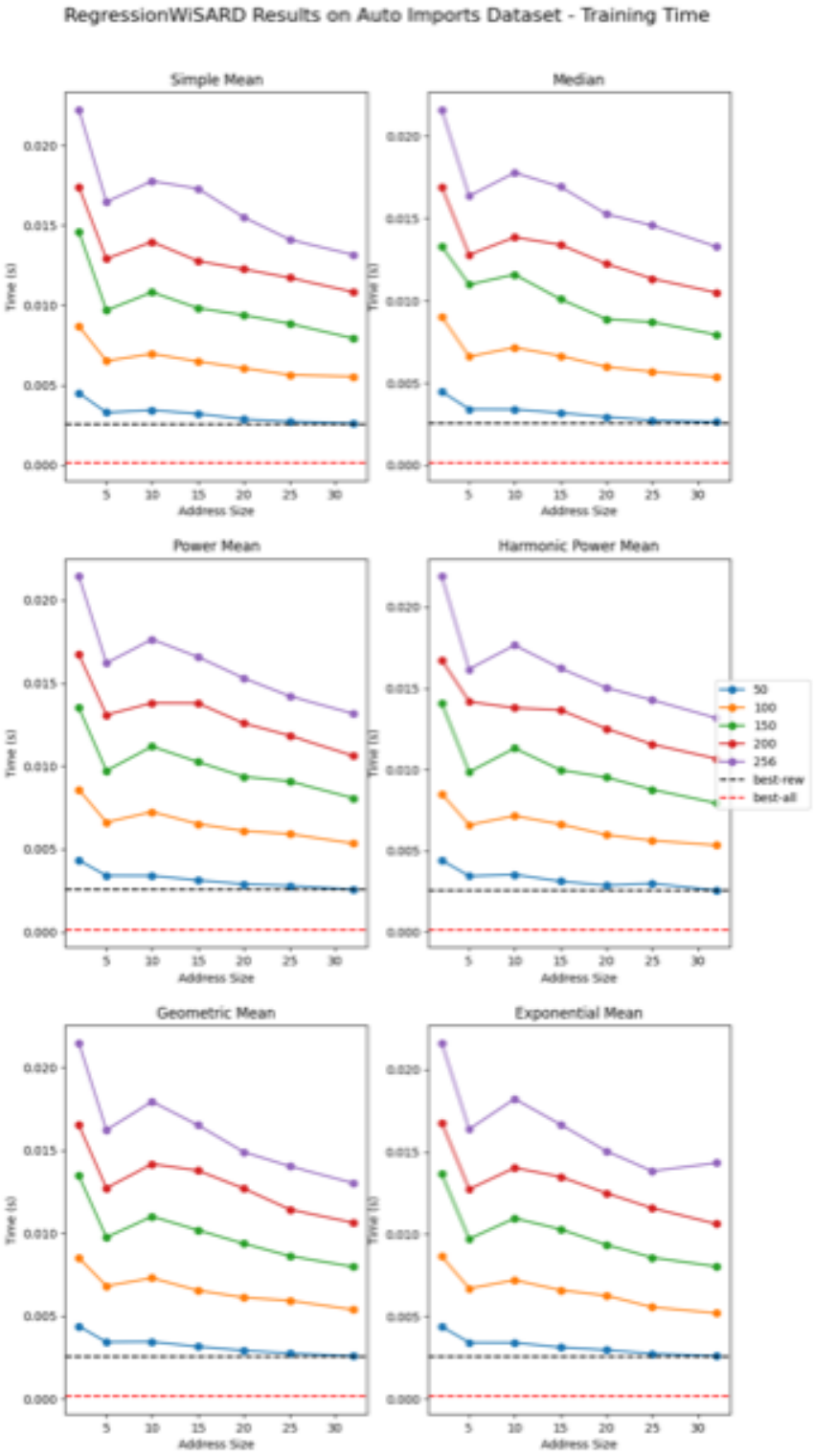

Figure C.5: Training time measures - Hyperparameter exploration on the Auto Imports Dataset.

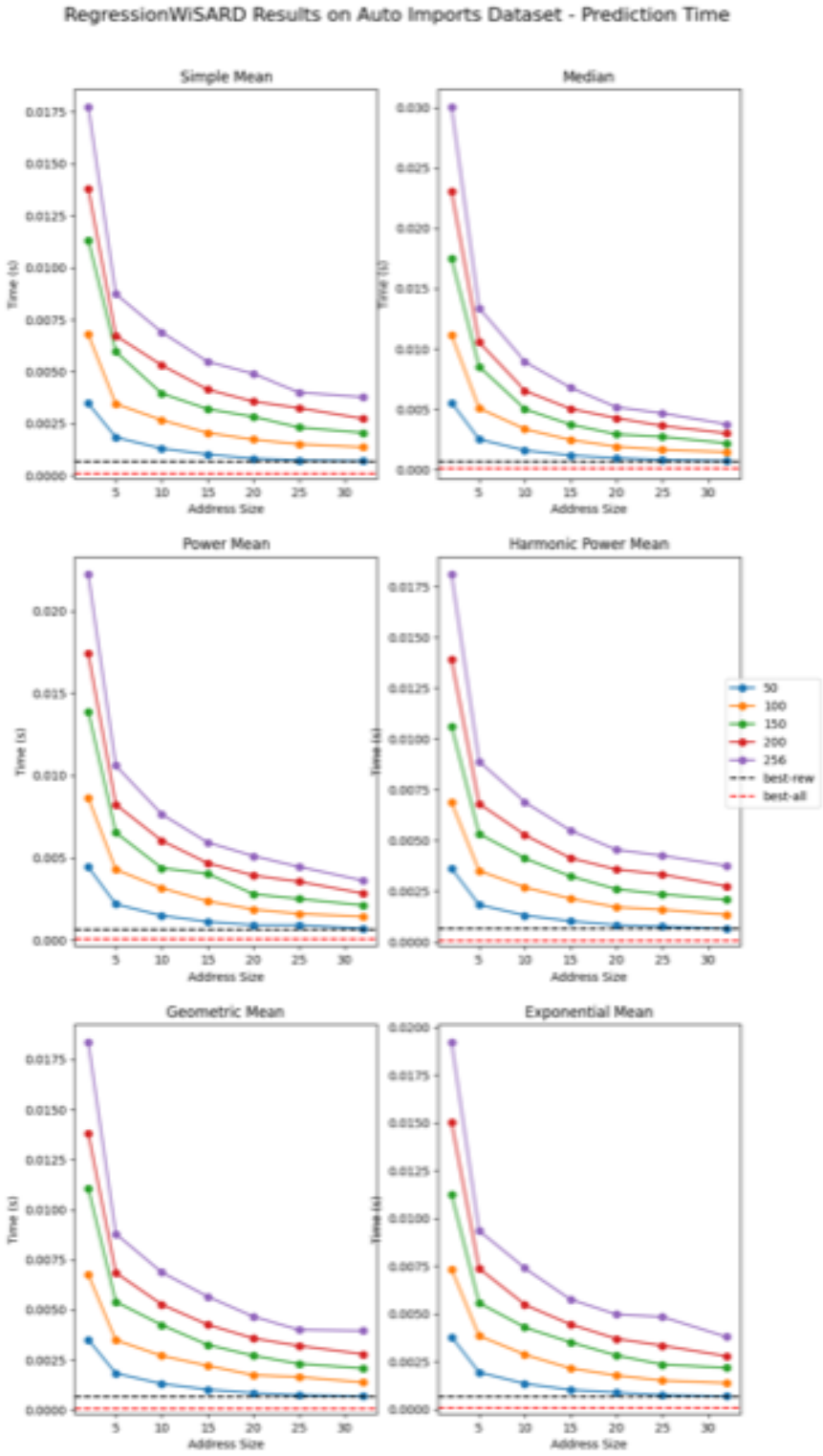

Figure C.6: Prediction time measures - Hyperparameter exploration on the Auto Imports Dataset.

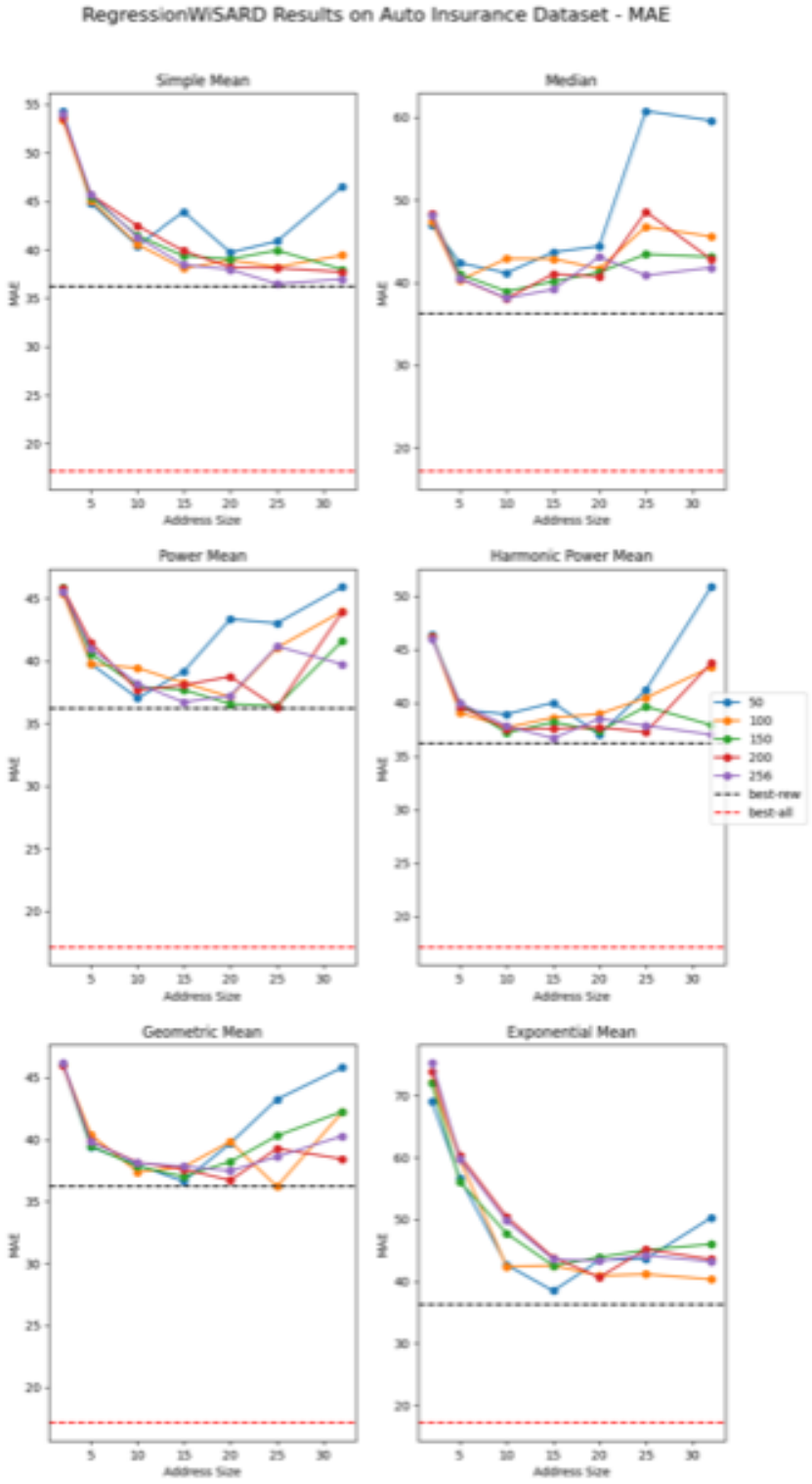

Figure C.7: MAE measures - Hyperparameter exploration on the Auto Insurance Dataset.

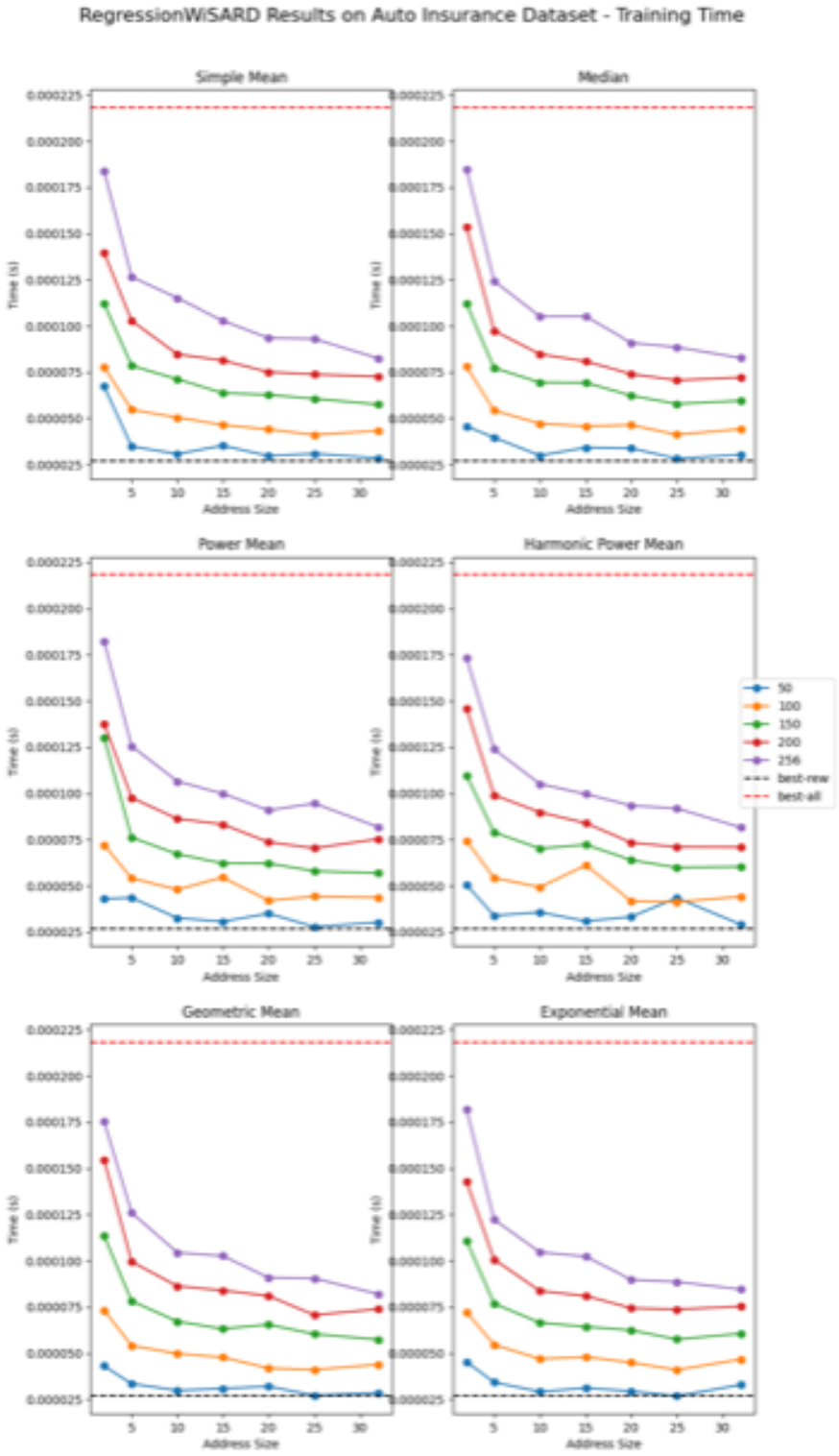

Figure C.8: Training time measures - Hyperparameter exploration on the Auto Insurance Dataset.

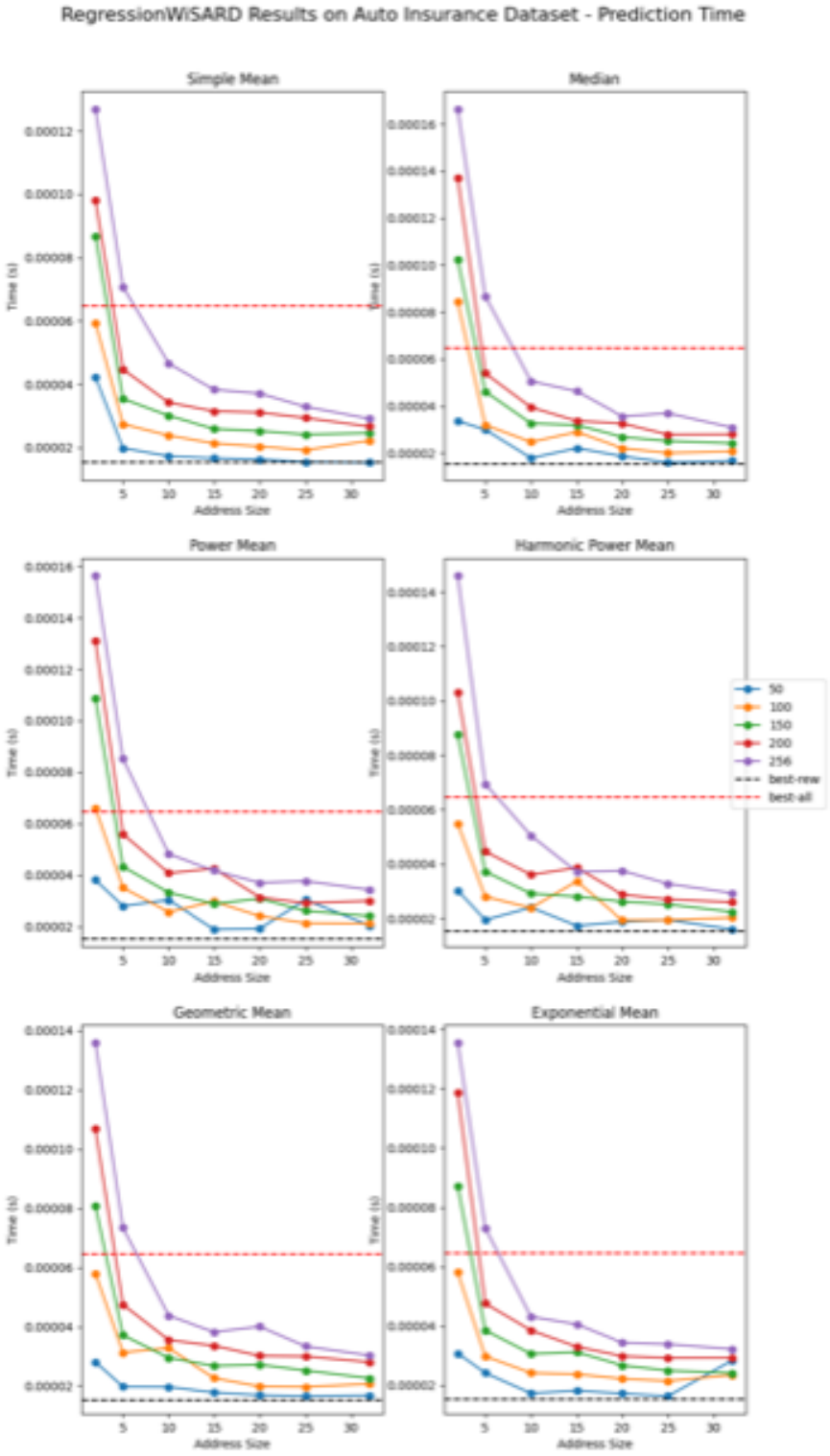

Figure C.9: Prediction time measures - Hyperparameter exploration on the Auto Insurance Dataset.

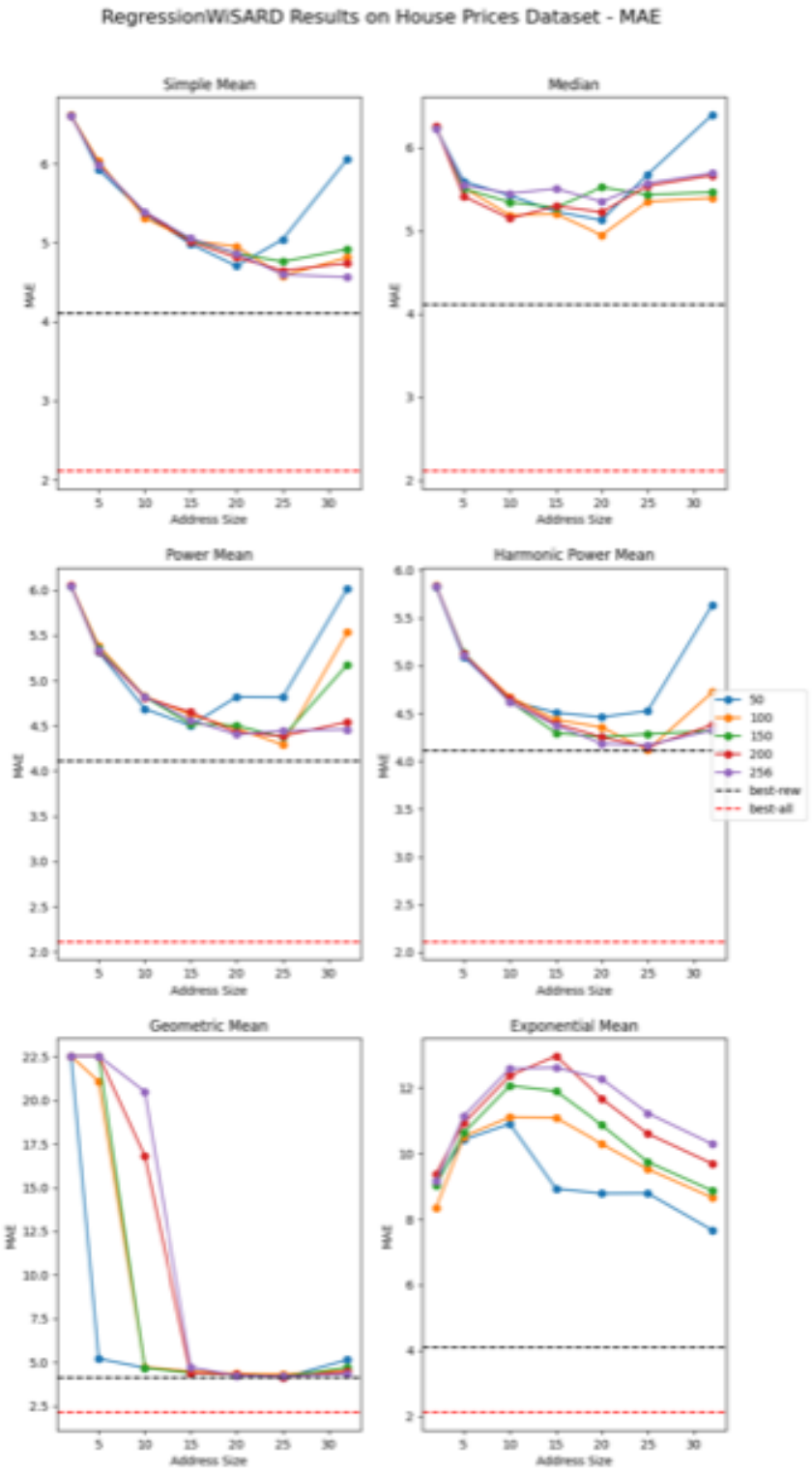

Figure C.10: MAE measures - Hyperparameter exploration on the House Prices Dataset.

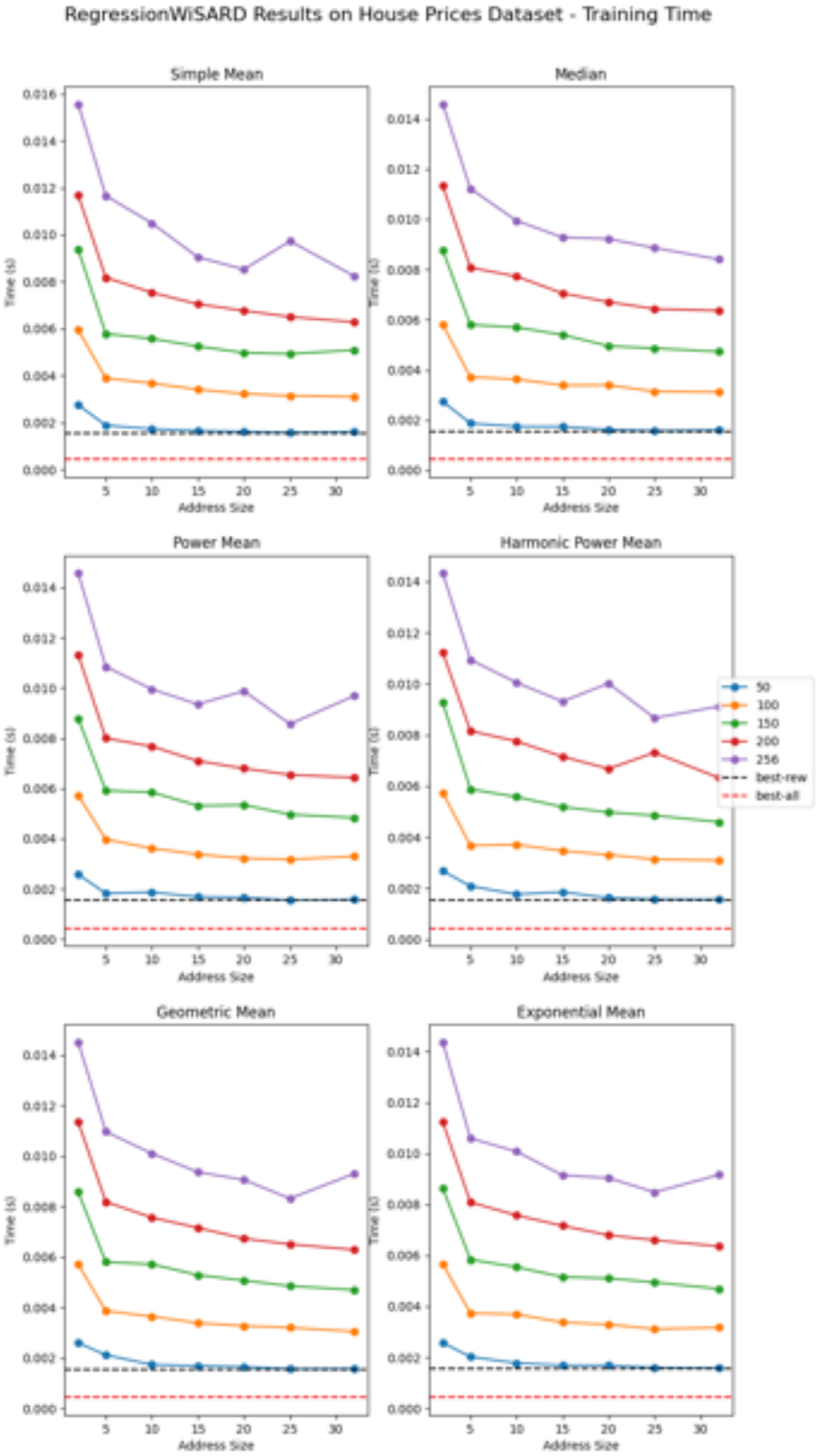

Figure C.11: Training time measures - Hyperparameter exploration on the House Prices Dataset.

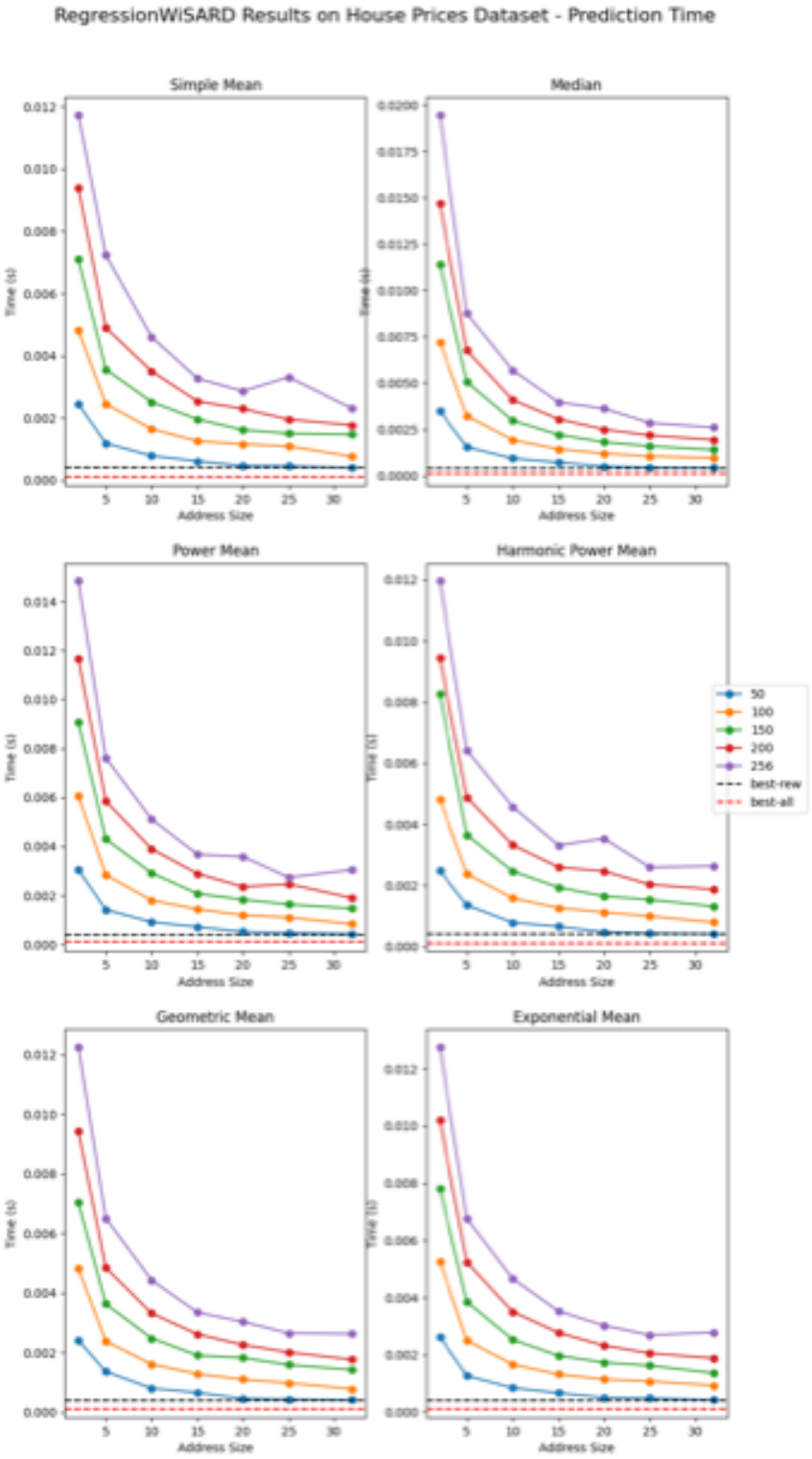

Figure C.12: Prediction time measures - Hyperparameter exploration on the House Prices Dataset.

Appendix D

Heuristics for Selecting Thermometer sizes

This appendix presents an experimental exploration of the proposed heuristic that uses a correlation coefficient to guide the process of choosing the size of the thermometer encoding for each feature. The experiments were conducted with different datasets and a comparison was made between the heuristic and the naïve approach of using the same value for all features. The experimental results show that, in some cases, a better accuracy performance can be obtained by using the proposed heuristic.

D.1 Experimental Framework

D.1.1 Datasets Description

To perform a comparison between approaches, a collection of datasets from the UCI machine learning repository [71] was used. The datasets present various characteristics regarding the number of observations and features, so it might be helpful to understand the effects of the proposed heuristics in each scenario.

Breast Cancer

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe the characteristics of the cell nuclei present in the image. This dataset has 569 instances and 32 attributes.

Ten real-valued features are computed for each cell nucleus: a) radius (mean of distances from center to points on the perimeter); b) texture (standard deviation of gray-scale values); c) perimeter; d) area; e) smoothness (local variation in radius lengths); f) compactness ($perimeter^2/area - 1.0$); g) concavity (severity of concave

portions of the contour); h) concave points (number of concave portions of the contour); i) symmetry; j) fractal dimension ("coastline approximation" - 1).

Diabetes

Diabetes patient records were obtained from an automatic electronic recording device and paper records. The automatic device had an internal clock to timestamp events, whereas the paper records only provided "logical time" slots (breakfast, lunch, dinner, bedtime).

Diabetes files consist of four fields per record: a) date in MM-DD-YYYY format; b) time in XX:YY format; c) code; d) value.

Ionosphere

This dataset has 34 continuous attributes and 351 instances. This radar data was collected by a system in Goose Bay, Labrador. This system consists of a phased array of 16 high-frequency antennas with a total transmitted power on the order of 6.4 kilowatts. See the paper for more details. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere.

Received signals were processed using an auto-correlation function whose arguments are the time of a pulse and the pulse number. There were 17 pulse numbers for the Goose Bay system. Instances in this database are described by 2 attributes per pulse number, corresponding to the complex values returned by the function resulting from the complex electromagnetic signal.

Sonar

This dataset has 60 continuous attributes and 208 instances. This is composed of two main files: the file *sonar.mines* contains 111 patterns obtained by bouncing sonar signals off a metal cylinder at various angles and under various conditions. The file *sonar.rocks* contains 97 patterns obtained from rocks under similar conditions. The transmitted sonar signal is a frequency-modulated chirp, rising in frequency. The data set contains signals obtained from a variety of different aspect angles, spanning 90 degrees for the cylinder and 180 degrees for the rock.

Each pattern is a set of 60 numbers in the range [0.0, 1.0] Each number represents the energy within a particular frequency band, integrated over a certain period. The integration aperture for higher frequencies occurs later since these frequencies are transmitted later during the chirp

Multiclass Classification Datasets

Iris

This dataset has 4 attributes and 150 instances. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are not linearly separable from each other.

Wheat Seed

This dataset has 7 attributes and 210 instances. The examined group comprised kernels belonging to three different varieties of wheat: Kama, Rosa, and Canadian, 70 elements each, randomly selected for the experiment. High-quality visualization of the internal kernel structure was detected using a soft X-ray technique. It is non-destructive and considerably cheaper than other more sophisticated imaging techniques like scanning microscopy or laser technology. The images were recorded on 13x18 cm X-ray KODAK plates. Studies were conducted using combined harvested wheat grain originating from experimental fields, explored at the Institute of Agrophysics of the Polish Academy of Sciences in Lublin.

To construct the data, seven geometric parameters of wheat kernels were measured: a)area; b) perimeter; c) compactness ($\frac{4*\pi*A}{P^2}$); d) length of kernel; e) width of kernel; f) asymmetry coefficient; g) length of kernel groove. All of these parameters are real-valued continuous.

Wine

This dataset is the result of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines and the goal is to use these constituents to determine the type of wine.

Regression Datasets

A single regression dataset was used in these experiments. The House Prices dataset is the most used for regression benchmarks. It has 77 features (both categorical and numerical) and the goal is to predict the selling price of each building. The training set has 973 observations, while the test set has 480.

D.1.2 Experiments Description

Two versions of the heuristic were developed and evaluated: the first one - referred to as *division* - defines a base value for all features. This value is then scaled

according to the absolute value of the correlation coefficient. In this case, unless the features present a full correlation concerning the output, the binary words generated from this heuristic will always be shorter than the words generated from the naïve approach. The second heuristic - referred to as *extend* - defines a fixed value for the final binary word. Each feature receives a fraction of this total amount, according to the magnitude of the coefficient.

For each dataset, a 10-fold cross-validation was performed with a data shuffle. The minimum and maximum values of the features were collected from the training set. The hyper-parameters of both WiSARD and ReW were explored by varying the base thermometer value in the set [50,100,150,200,256], the tuple size in the set [2,5,10,15,20,25,31], and, for ReW, the simple mean, median and quadratic mean. Also, a slightly different correlation coefficient was used for comparison alongside the Pearson correlation coefficient. The Spearman's ρ, or Spearman's rank correlation coefficient [80], is a metric that relates to Pearson's r, but instead of modeling the variables with a linear relationship, it models them using a monotonic function.

This process was repeated 100 times, and the mean values were computed and reported. One aspect to be mentioned is the standard deviation of the results. In all cases reported, regarding the accuracy score, MAE, and training and prediction speed, the standard deviations were significantly low (below 0.1% or 1 second). Then, for the sake of simplicity, the standard deviation values of the experiments will not be shown. Also, although the previously mentioned recommendation is to remove features with a high correlation with each other and a significantly low correlation with the target, such modifications will not be performed. This is due to the goal of comparing the impact between approaches, and not necessarily to improve the model's performance when compared to other models.

D.2 Results and Discussion

Figures D.1-D.24 present the computational results from the experiments. Each figure contains 10 subfigures, each one with three colored lines. The red lines indicate the results from the heuristic using the Pearson correlation coefficient, while the green lines indicate the ones using the Spearman correlation coefficient. The blue lines indicate the naïve approach.

In each figure, the upper five subfigures represent results using the *division* heuristic, while the lower five indicate the results from the *extend* heuristic. Above the graphs, there is a title indicating the base thermometer size used: the subfigures on the far left use thermometers of size 50, and the subfigures on the far right use thermometers of size 250. The x-axis of the subgraphs indicates the variation of tuple size within the same thermometer. The y-axis indicates the metric of the

subfigures, and for each set of five subfigures, the axis is normalized. Each dataset has three associated figures containing its results. The groups of figures are displayed in alphabetical order according to the dataset name. In each group, figures #1, #2, and #3 present the results using the classic thermometer encoding regarding prediction performance, followed by the training and prediction times respectively.

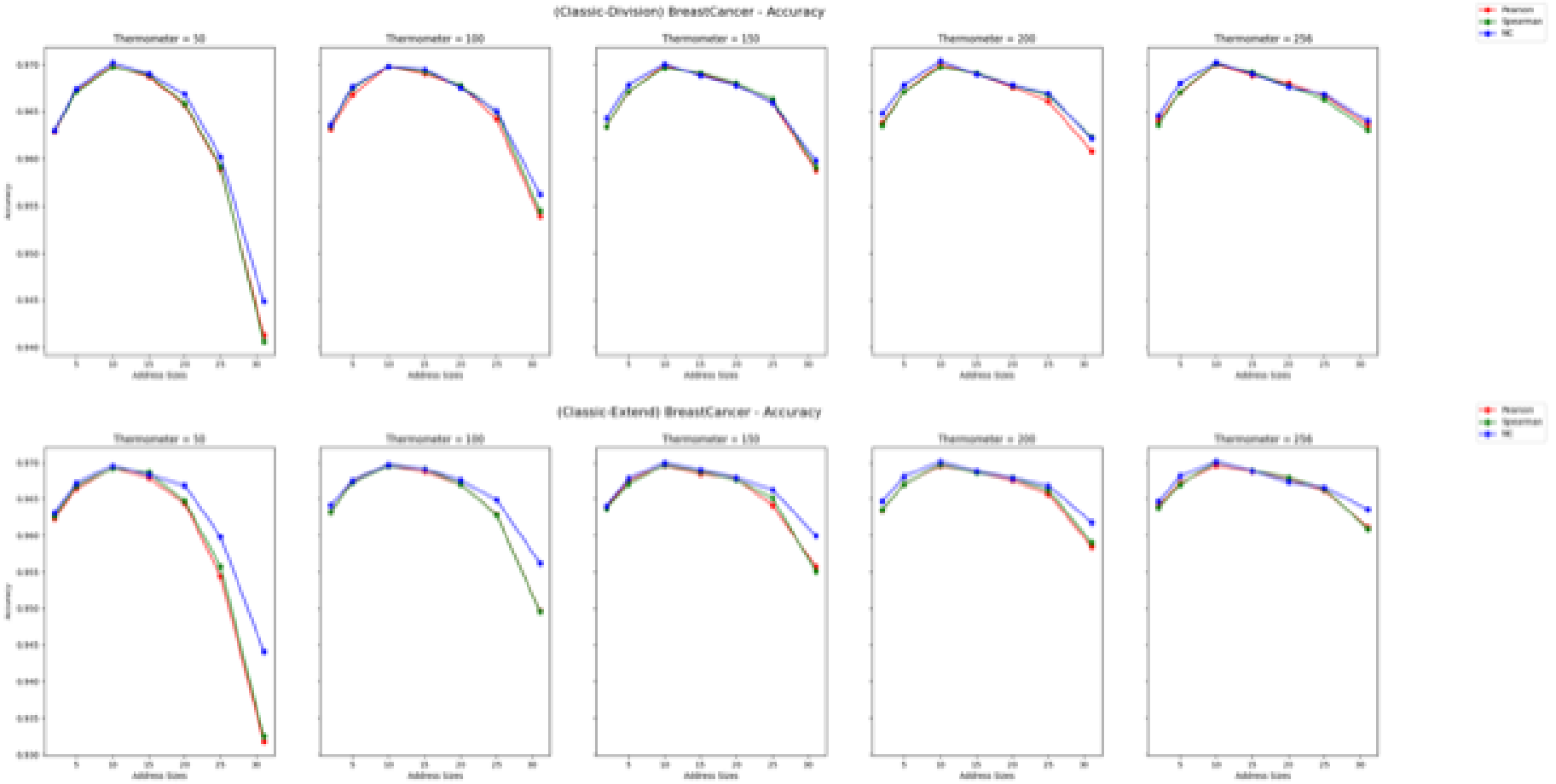

Figure D.1: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Breast Cancer dataset.

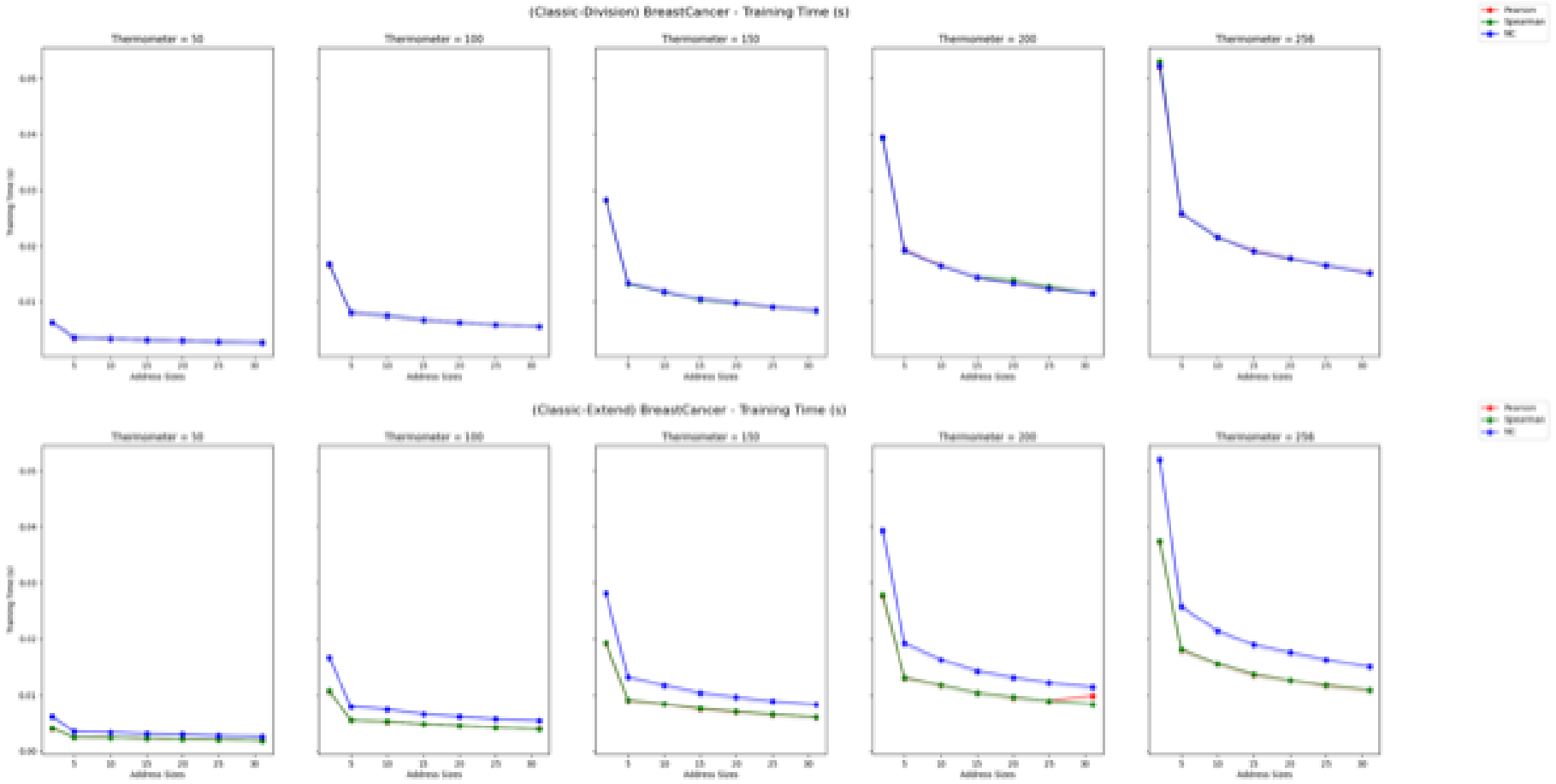

Figure D.2: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Breast Cancer dataset.

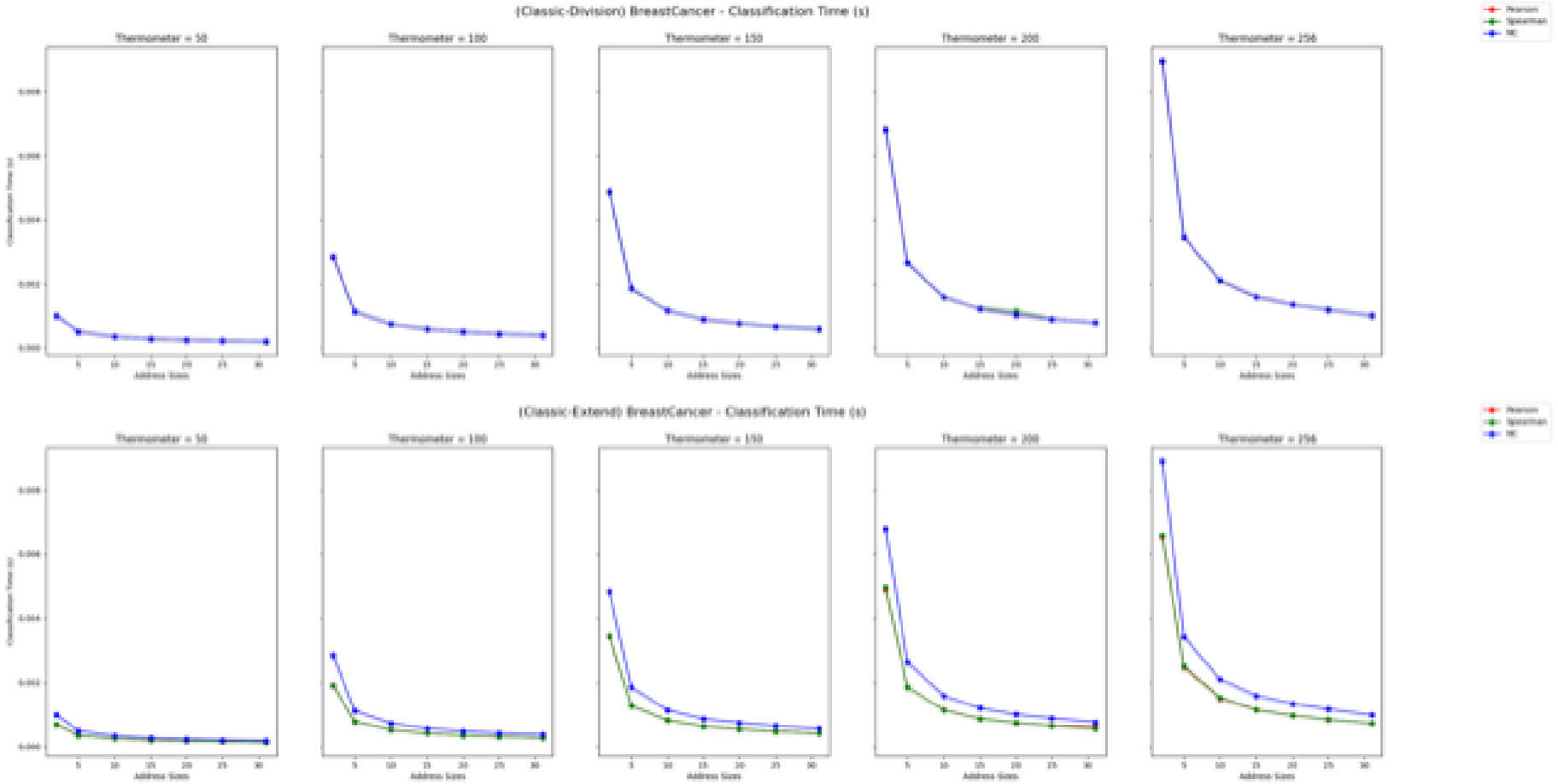

Figure D.3: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Breast Cancer dataset.

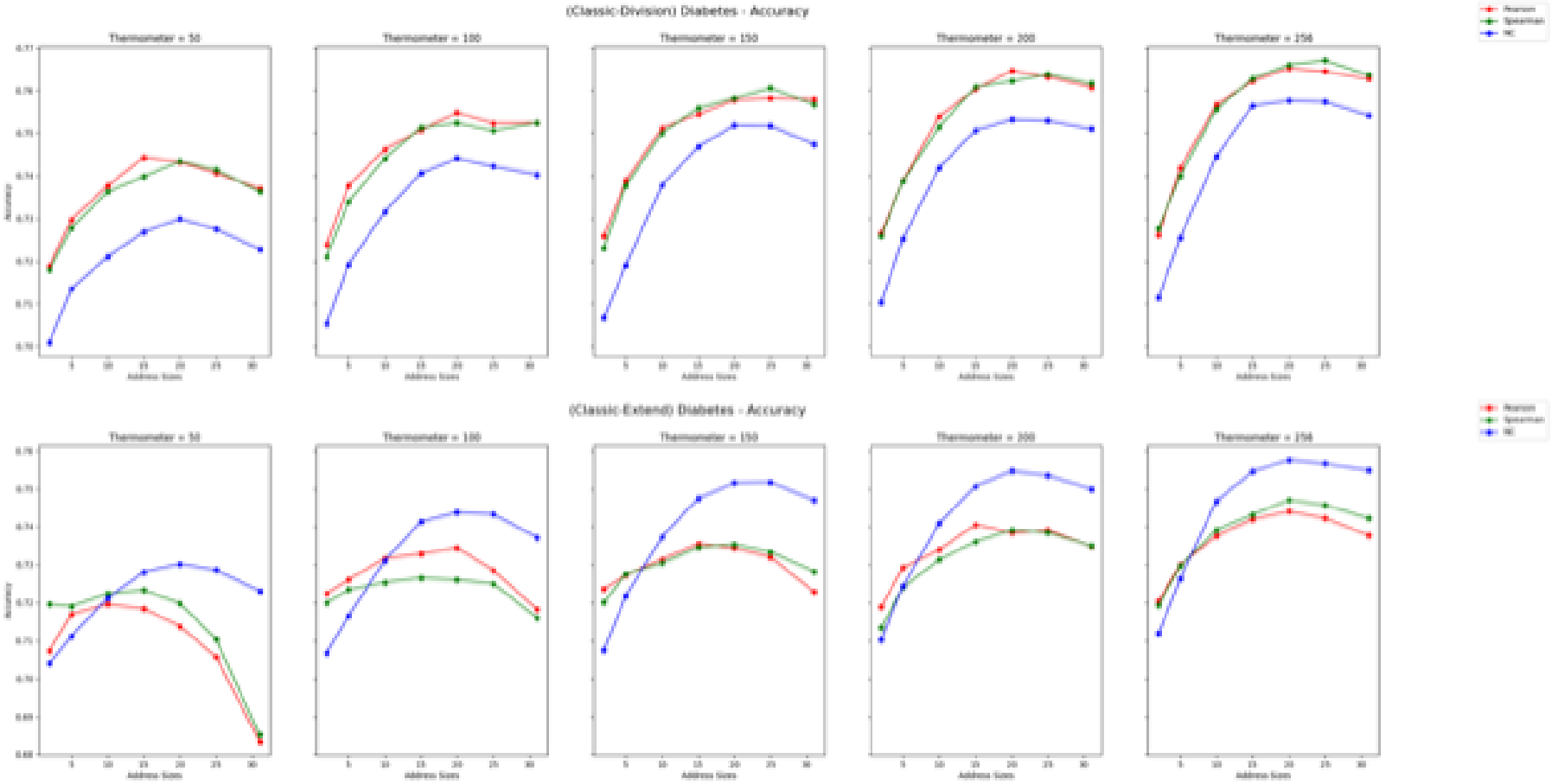

Figure D.4: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Diabetes dataset.

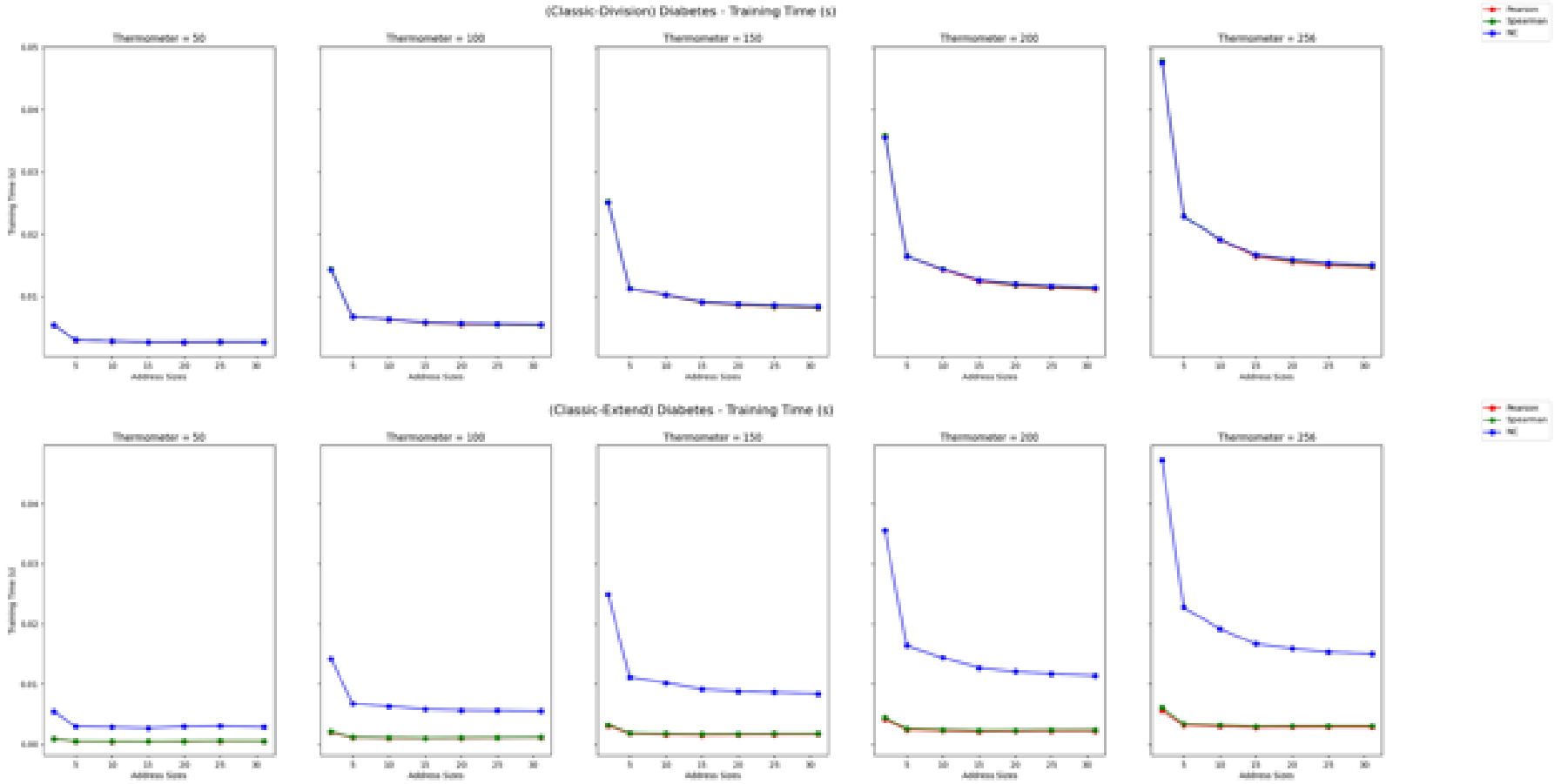

Figure D.5: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Diabetes dataset.

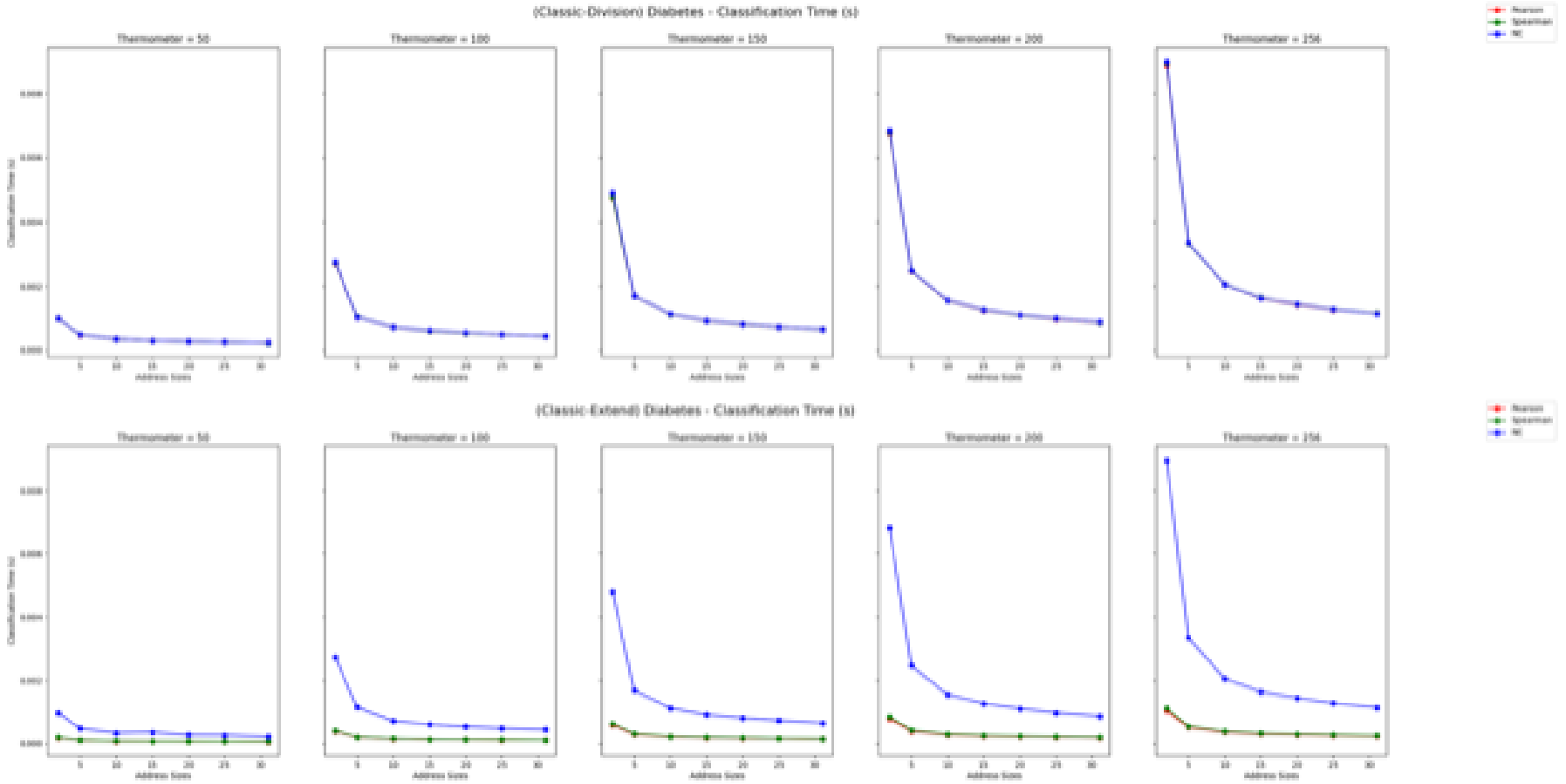

Figure D.6: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Diabetes dataset.

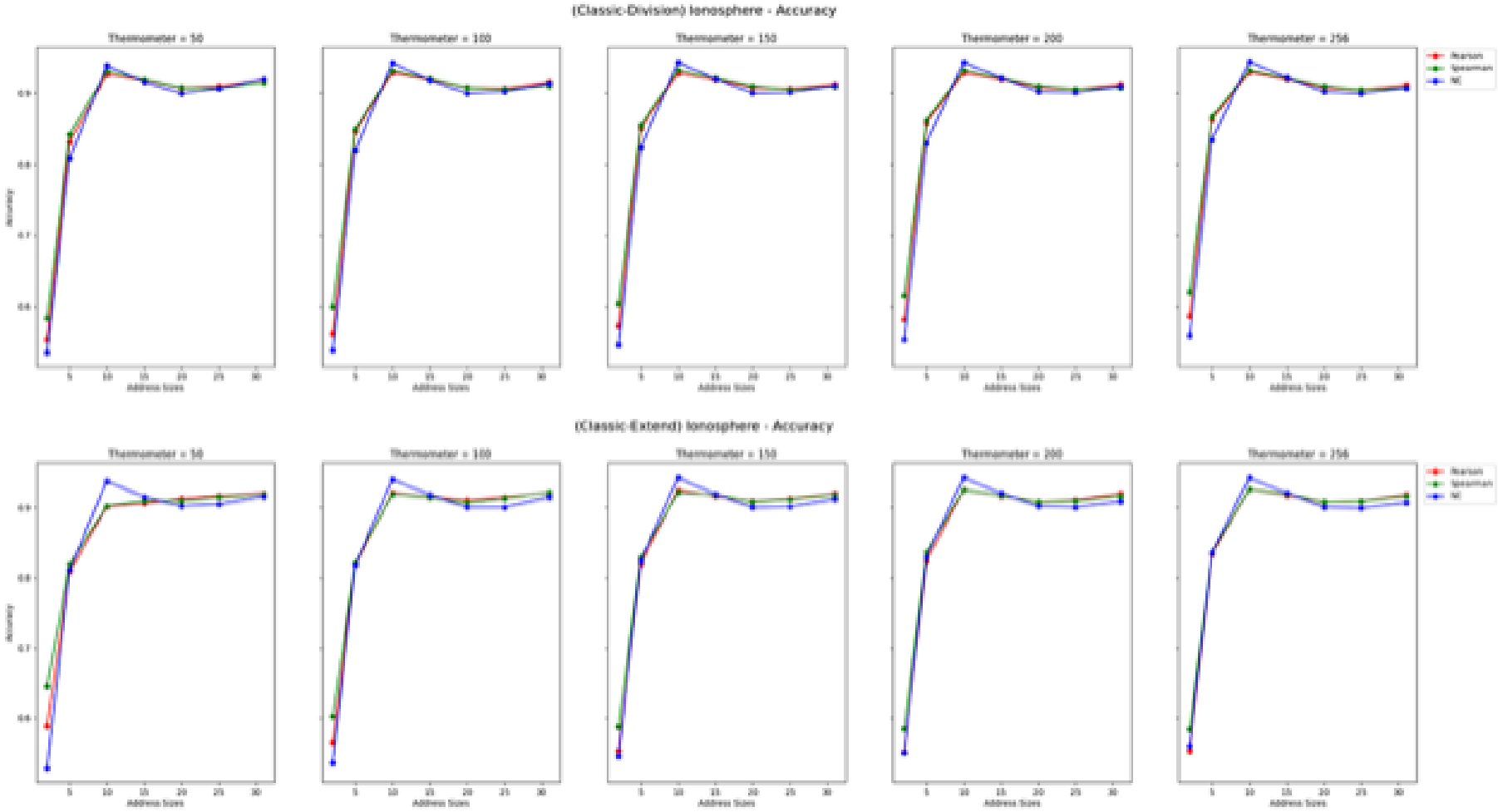

Figure D.7: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Ionosphere dataset.

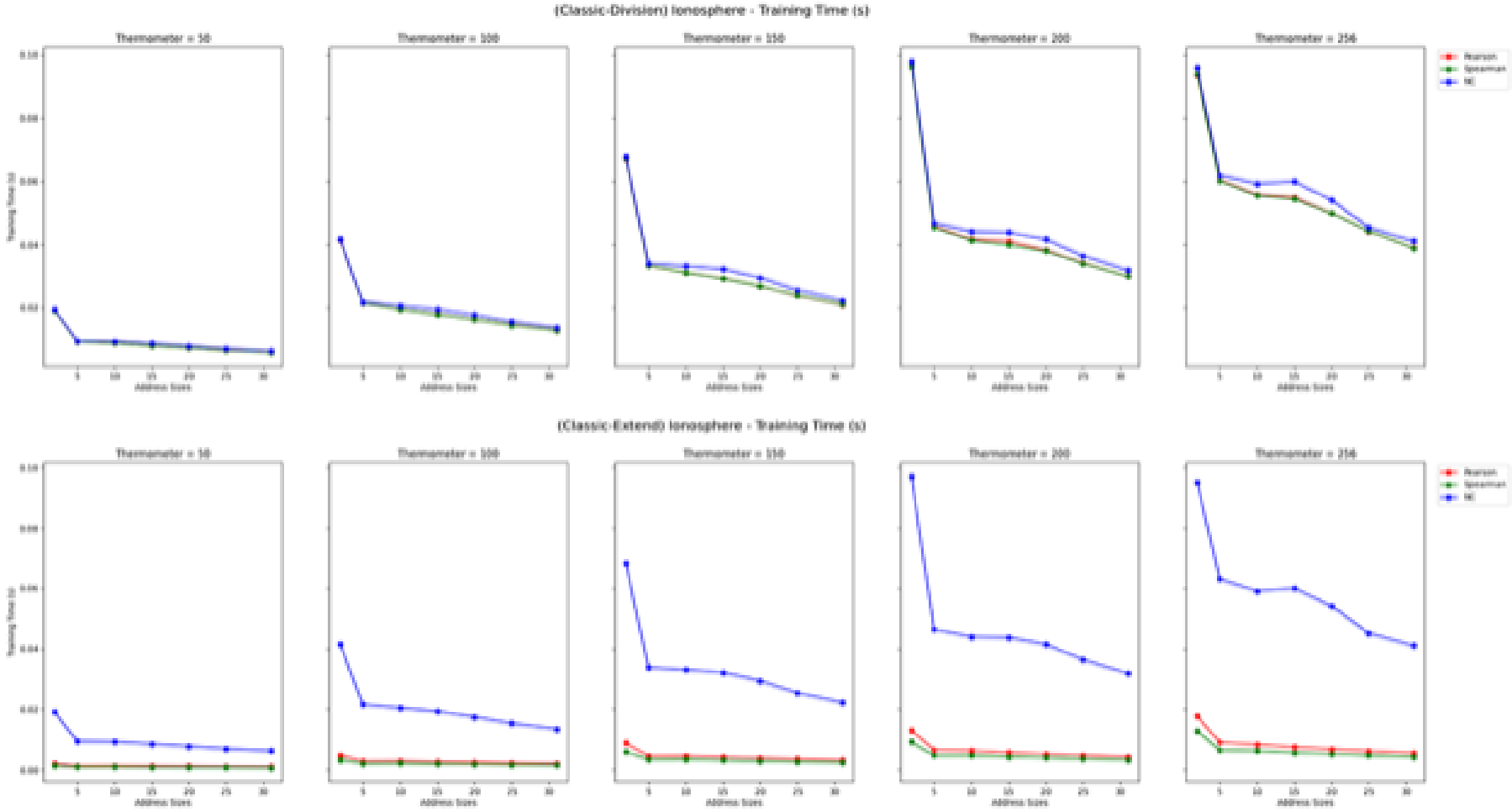

Figure D.8: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Ionosphere dataset.

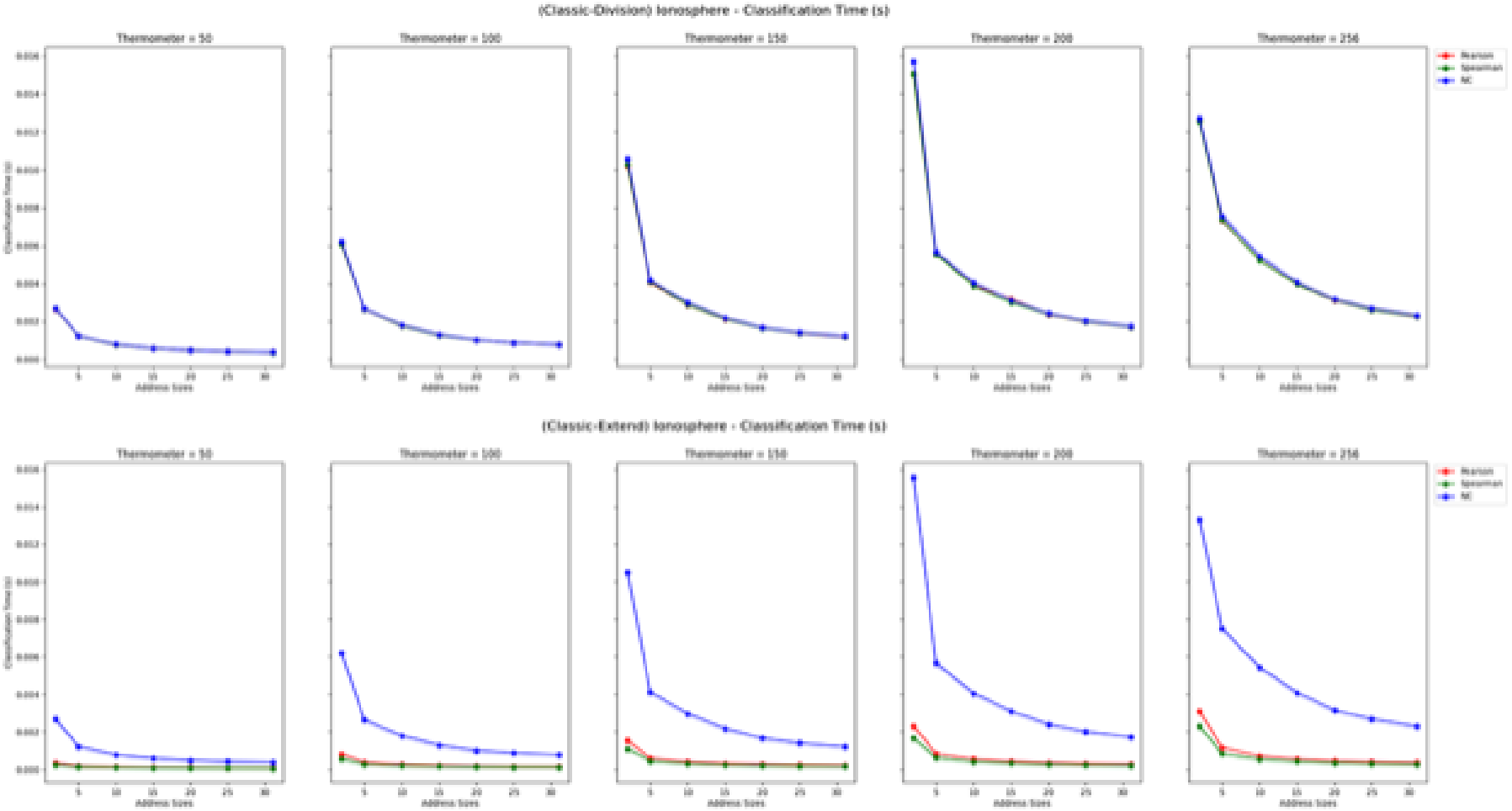

Figure D.9: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Ionosphere dataset.

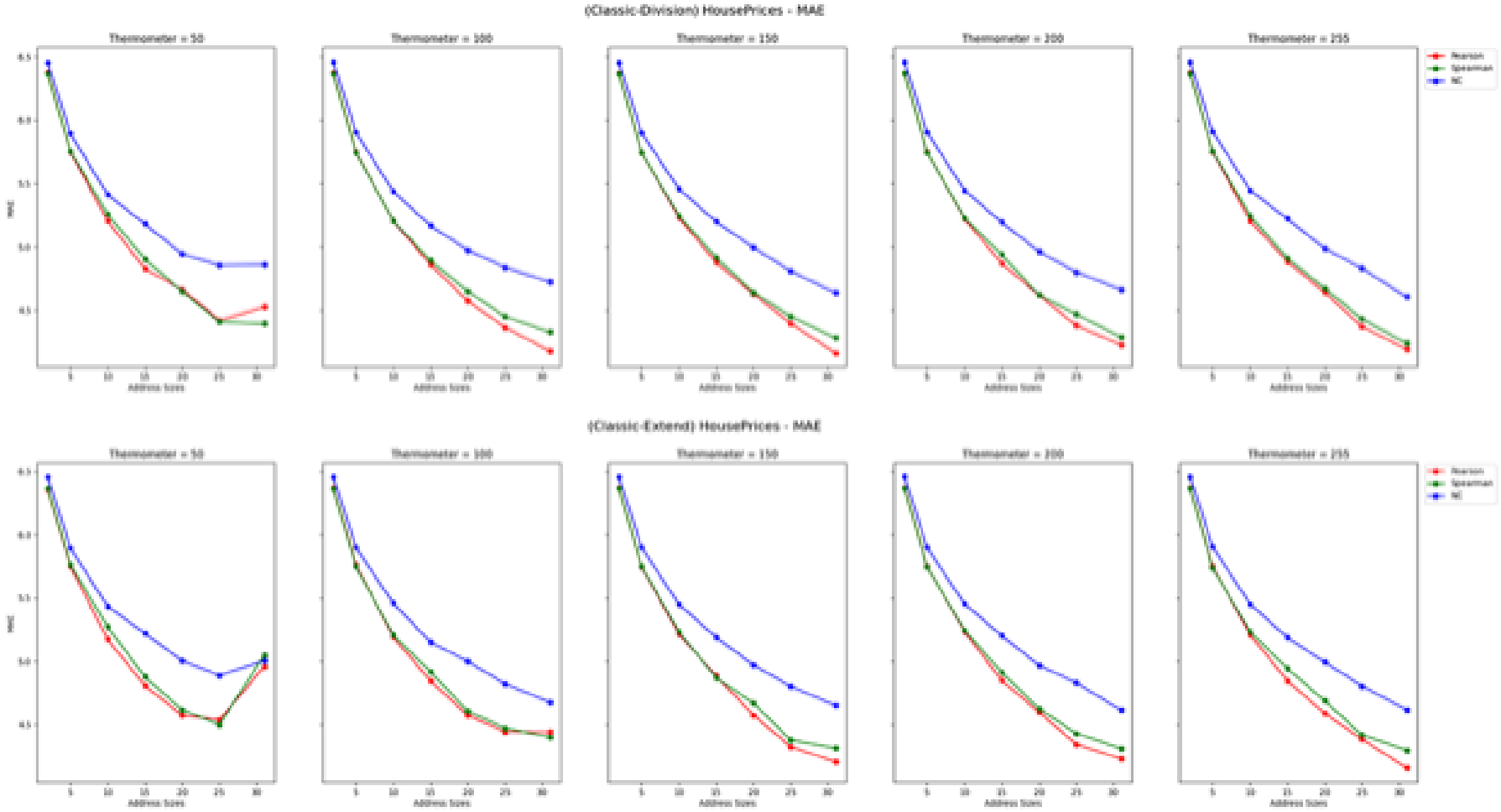

Figure D.10: Mean Absolute Error score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the House Prices dataset.

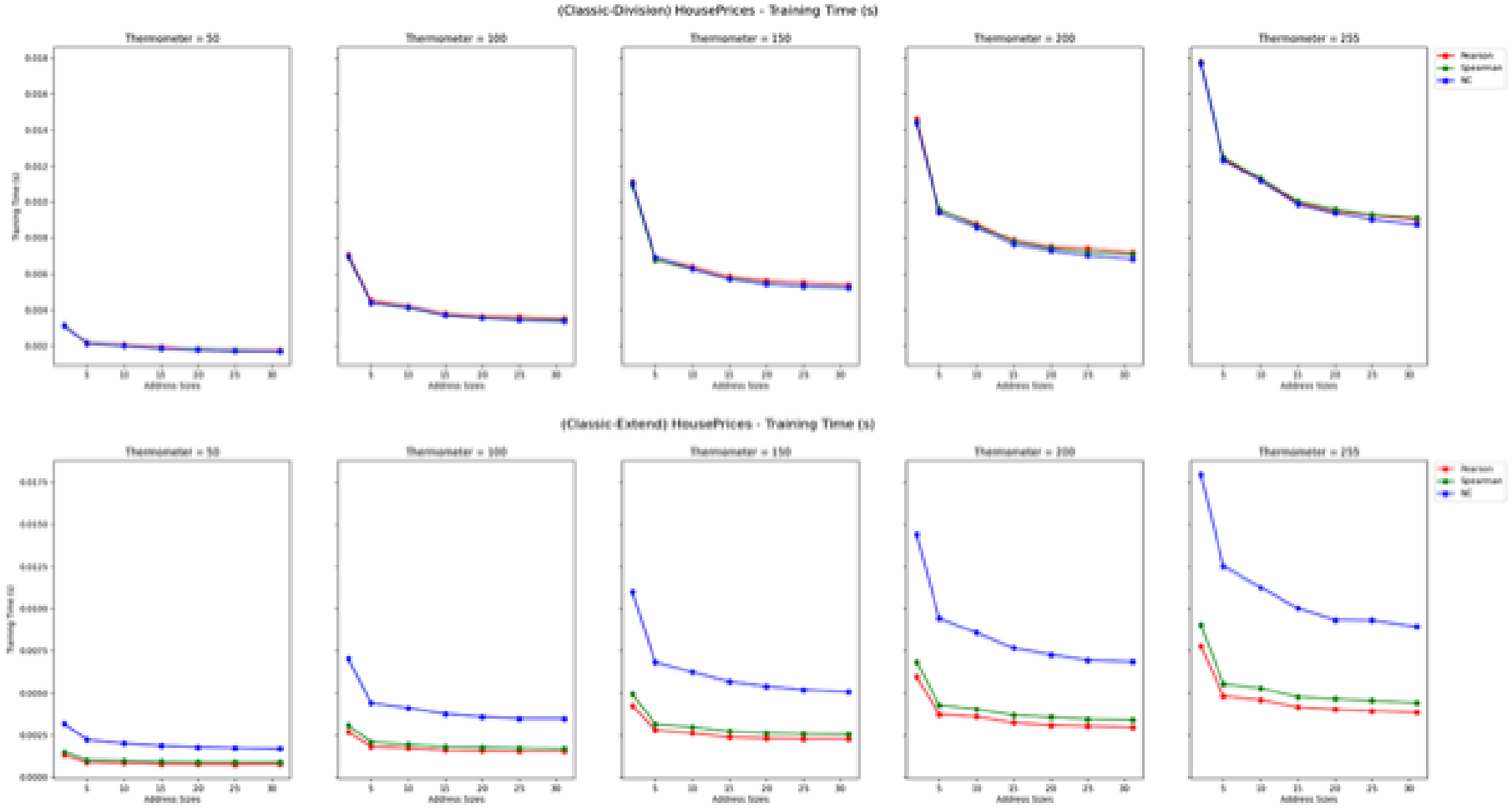

Figure D.11: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the House Prices dataset.

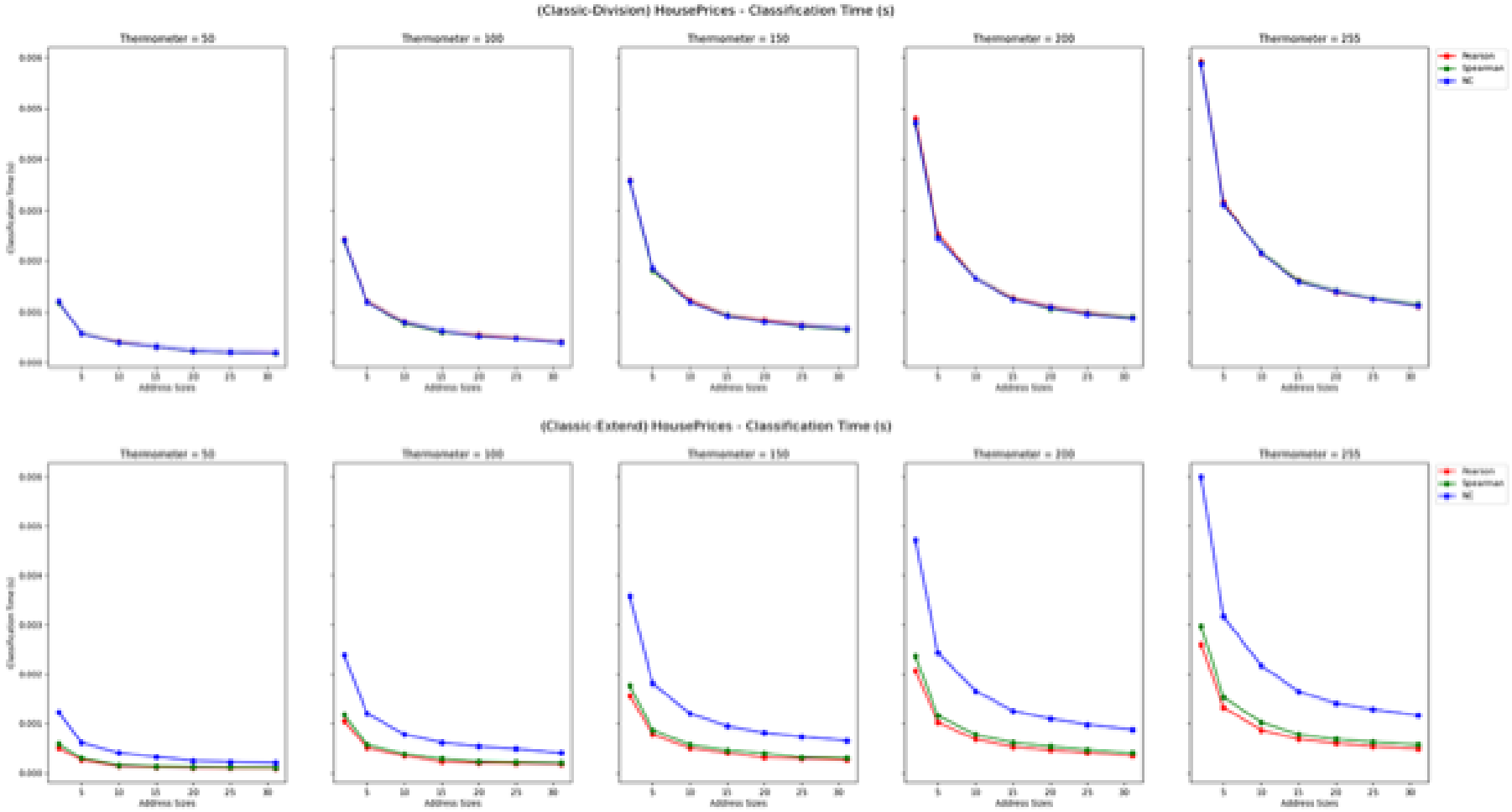

Figure D.12: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the House Prices dataset.

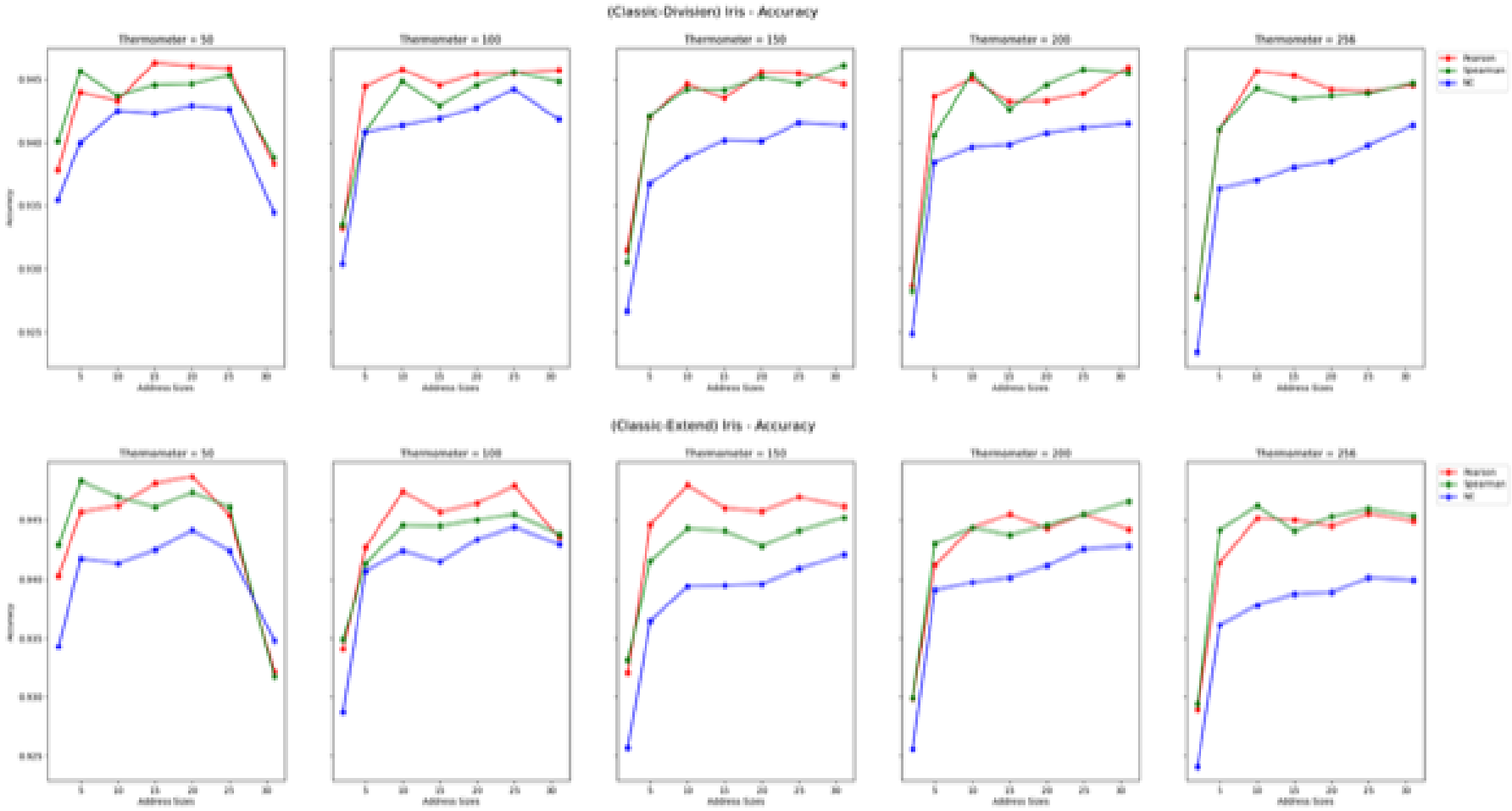

Figure D.13: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Iris dataset.

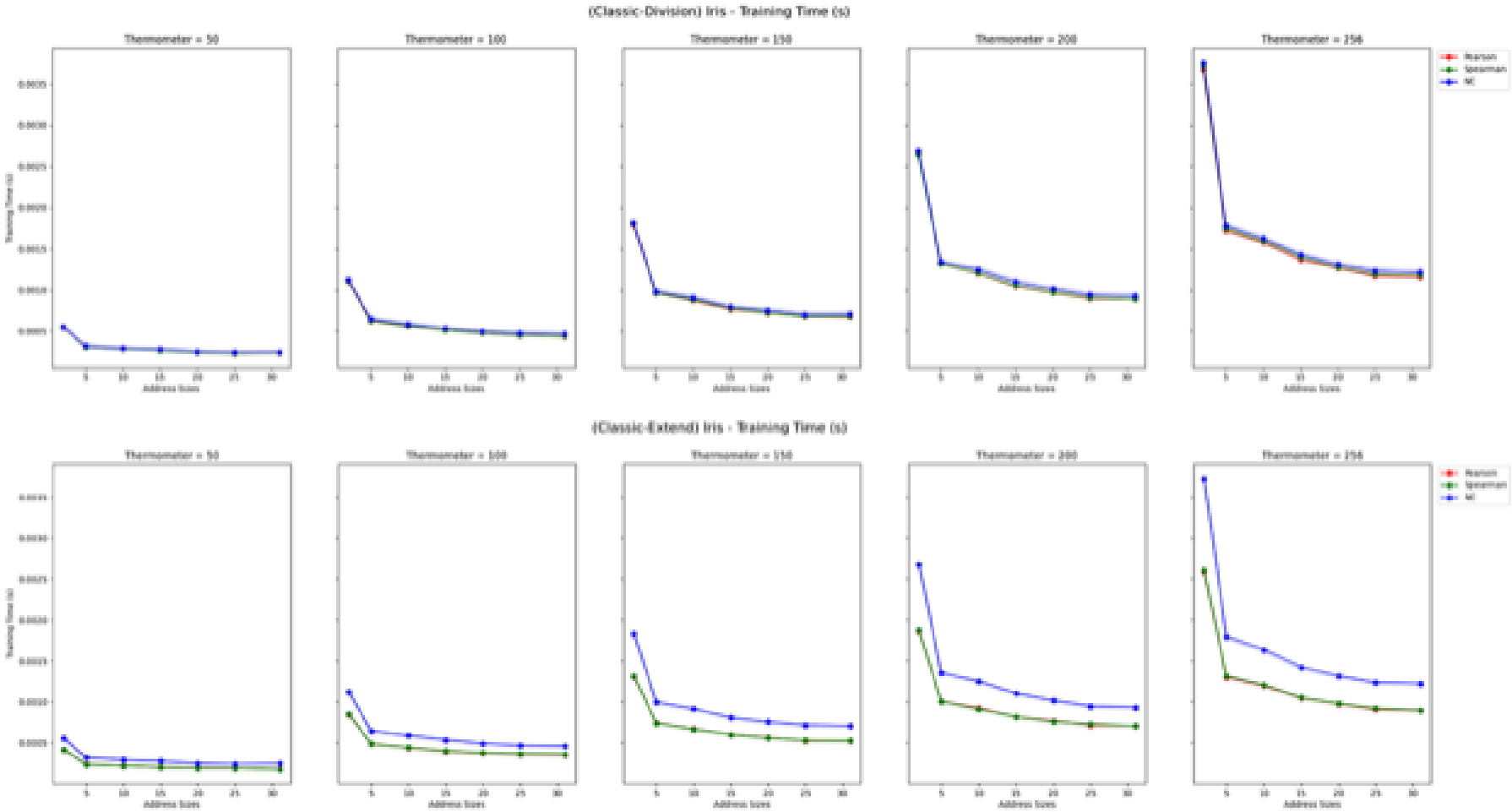

Figure D.14: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Iris dataset.

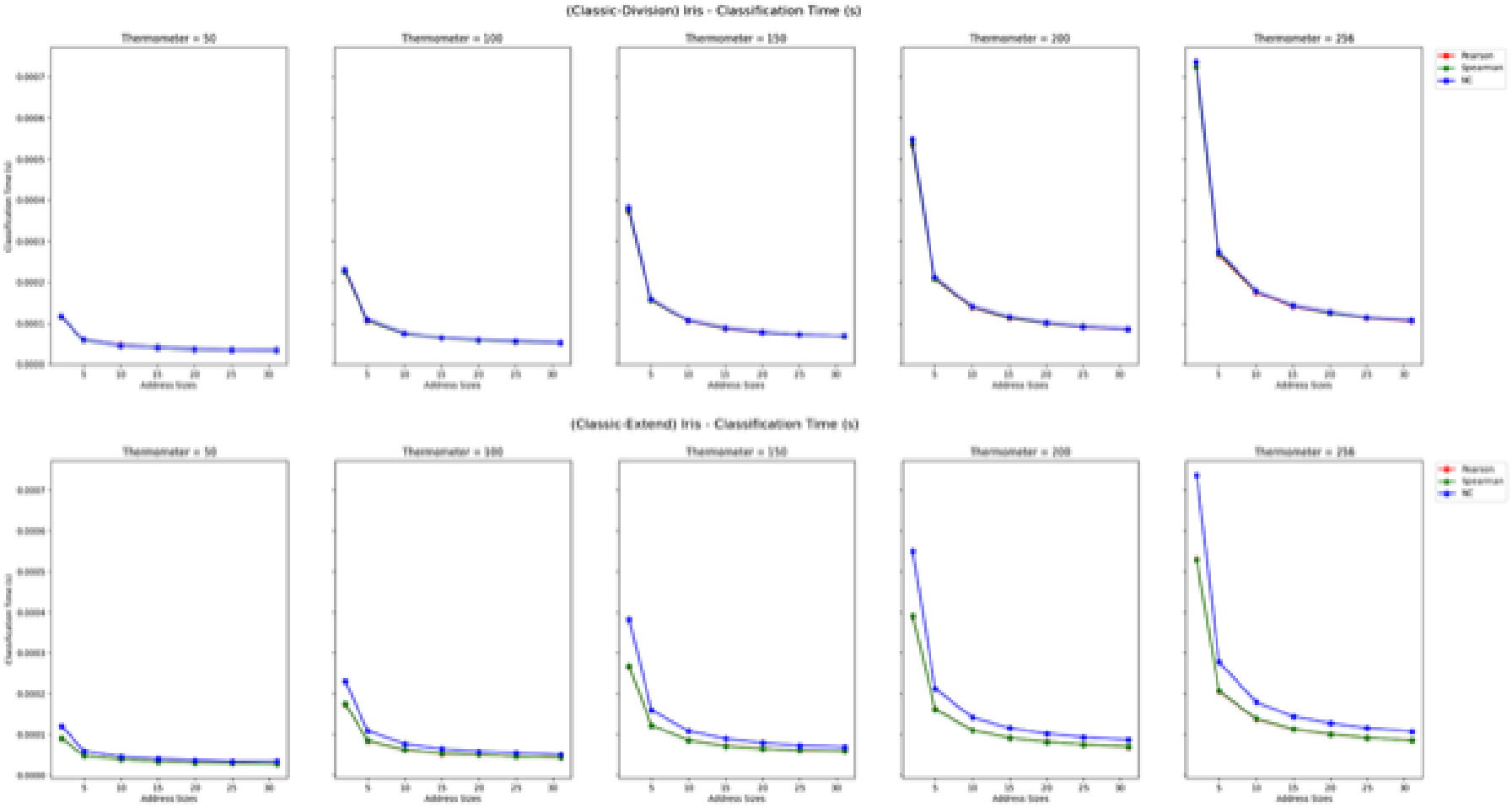

Figure D.15: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Iris dataset.

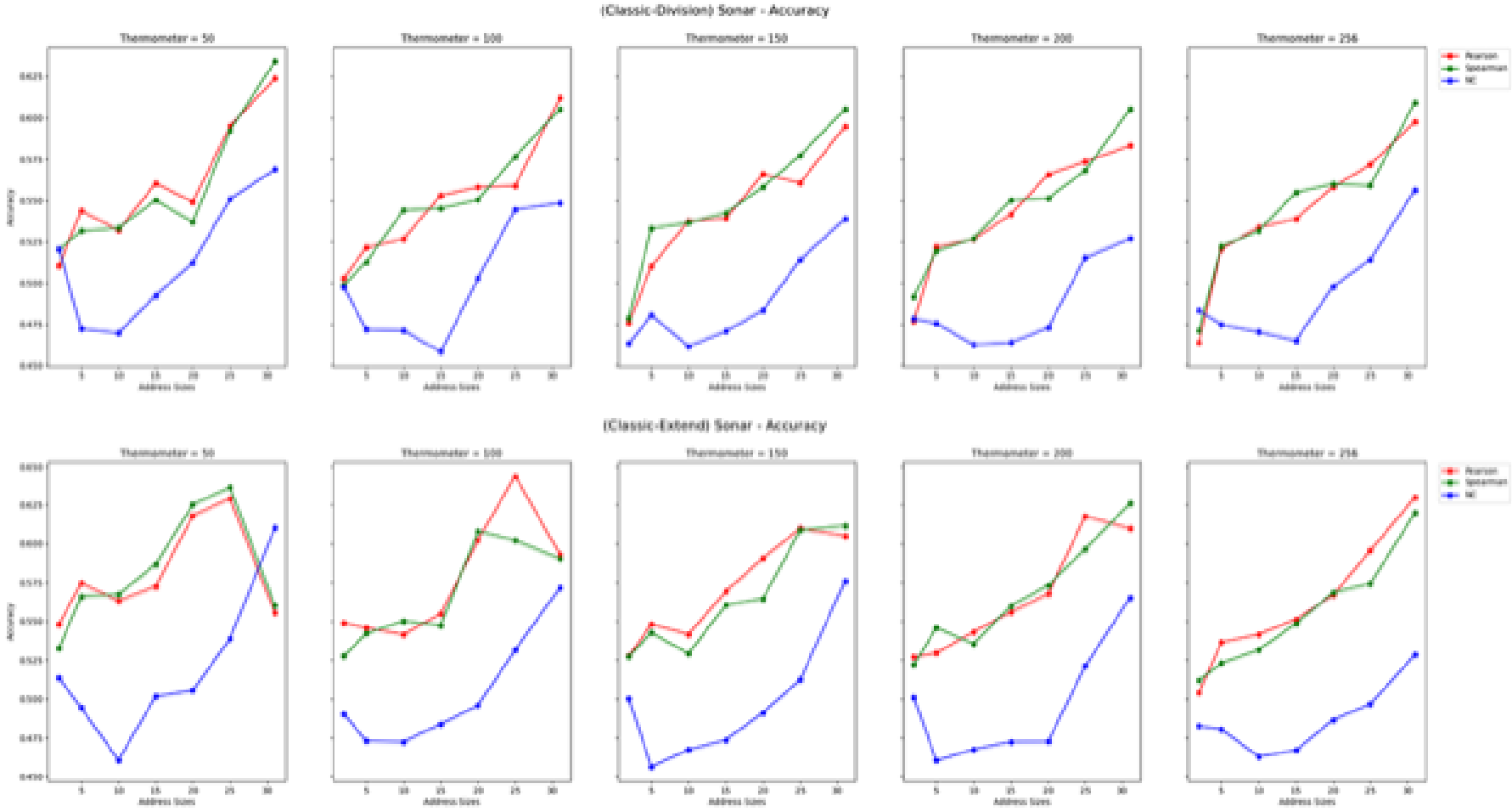

Figure D.16: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Sonar dataset.

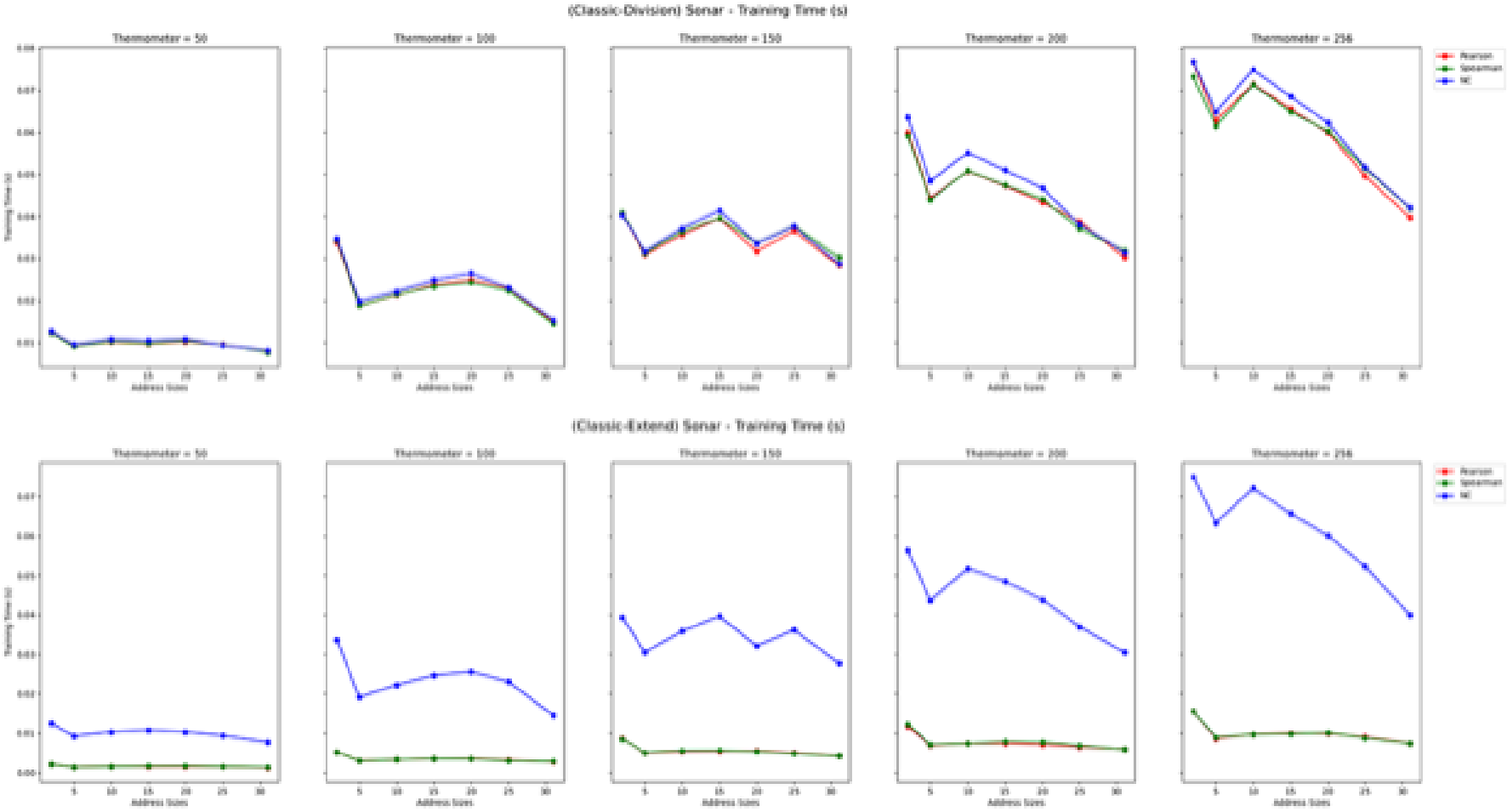

Figure D.17: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Sonar dataset.

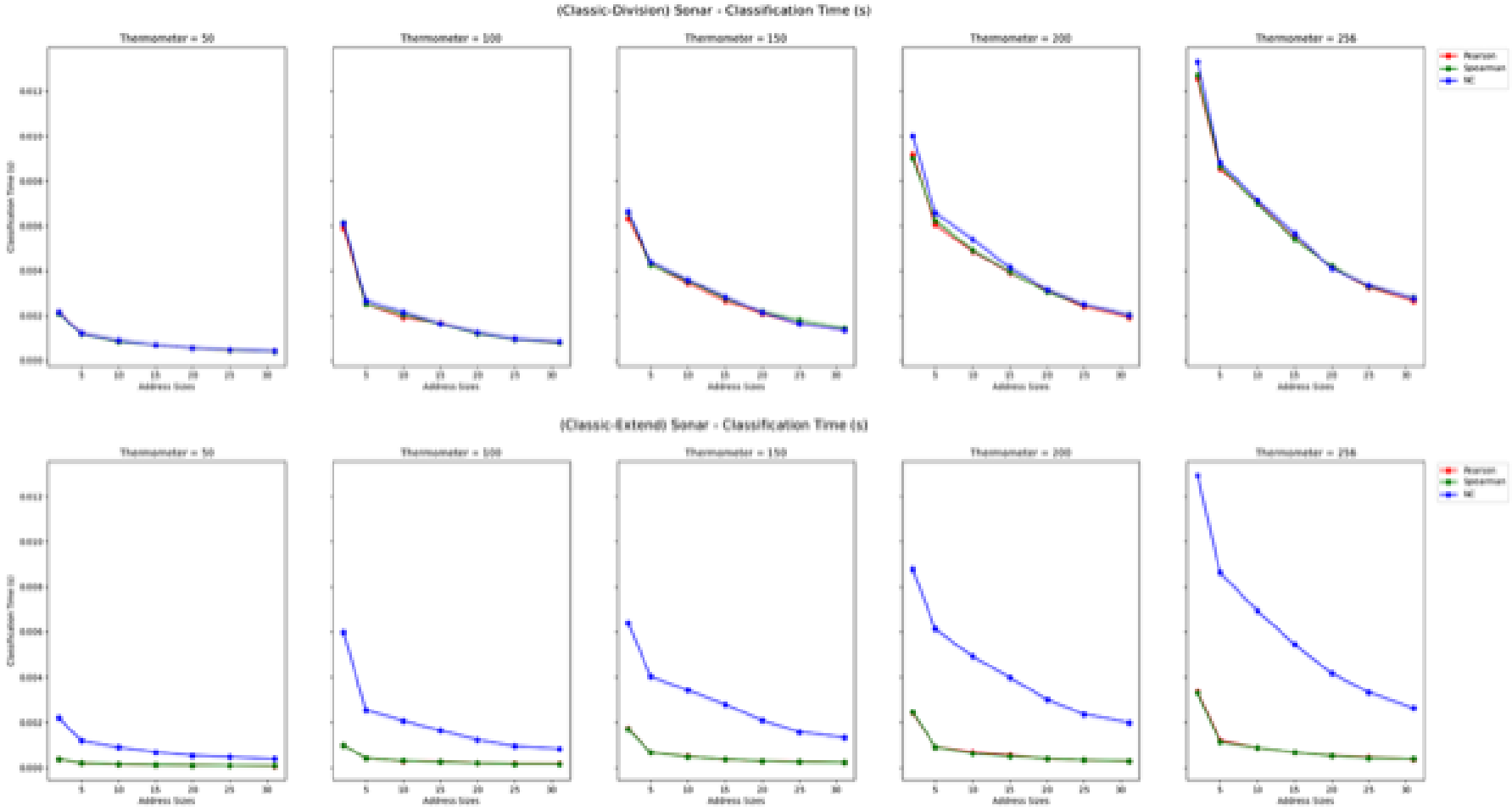

Figure D.18: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Sonar dataset.

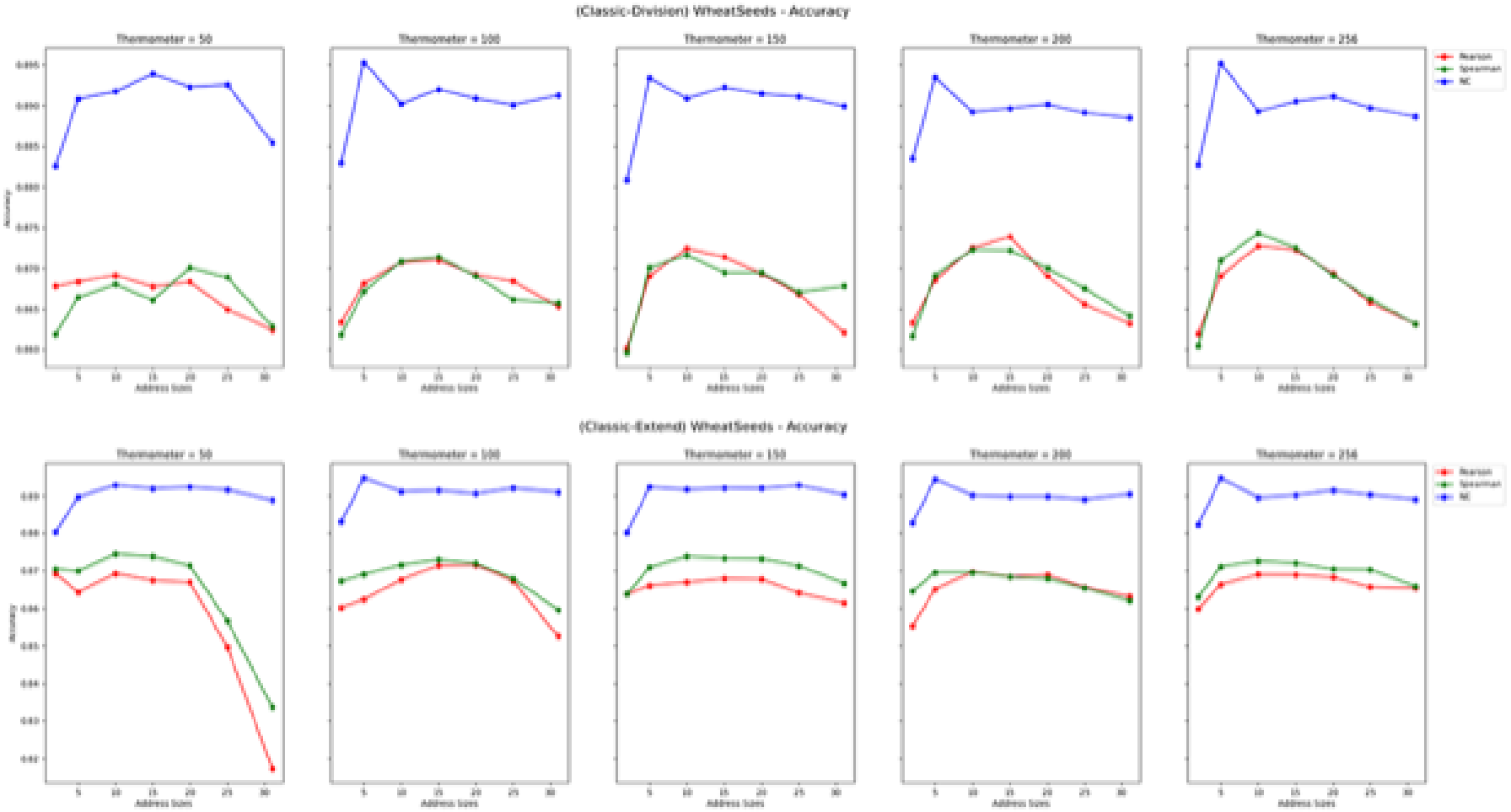

Figure D.19: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wheat Seeds dataset.

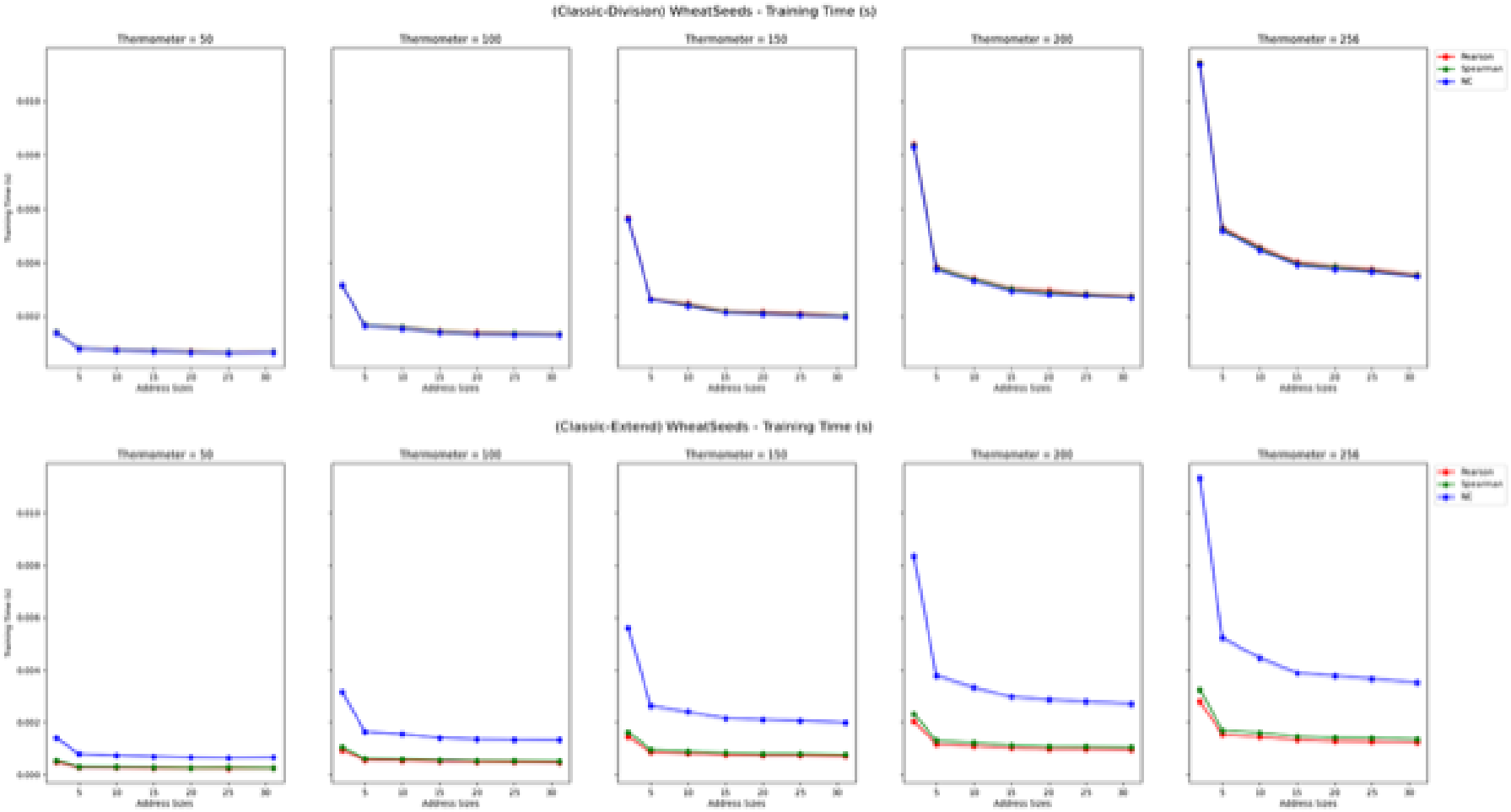

Figure D.20: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wheat Seeds dataset.

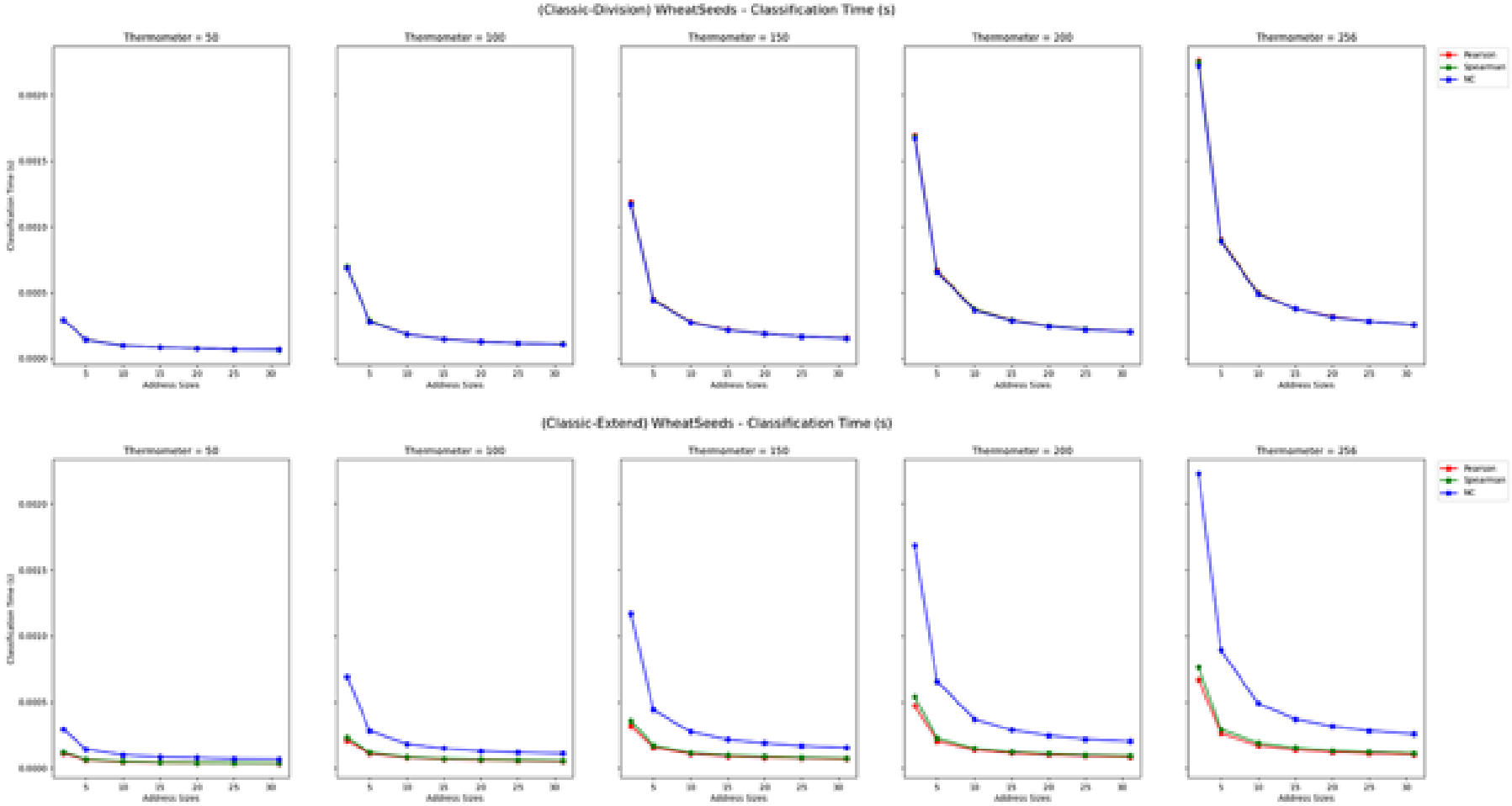

Figure D.21: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wheat Seeds dataset.

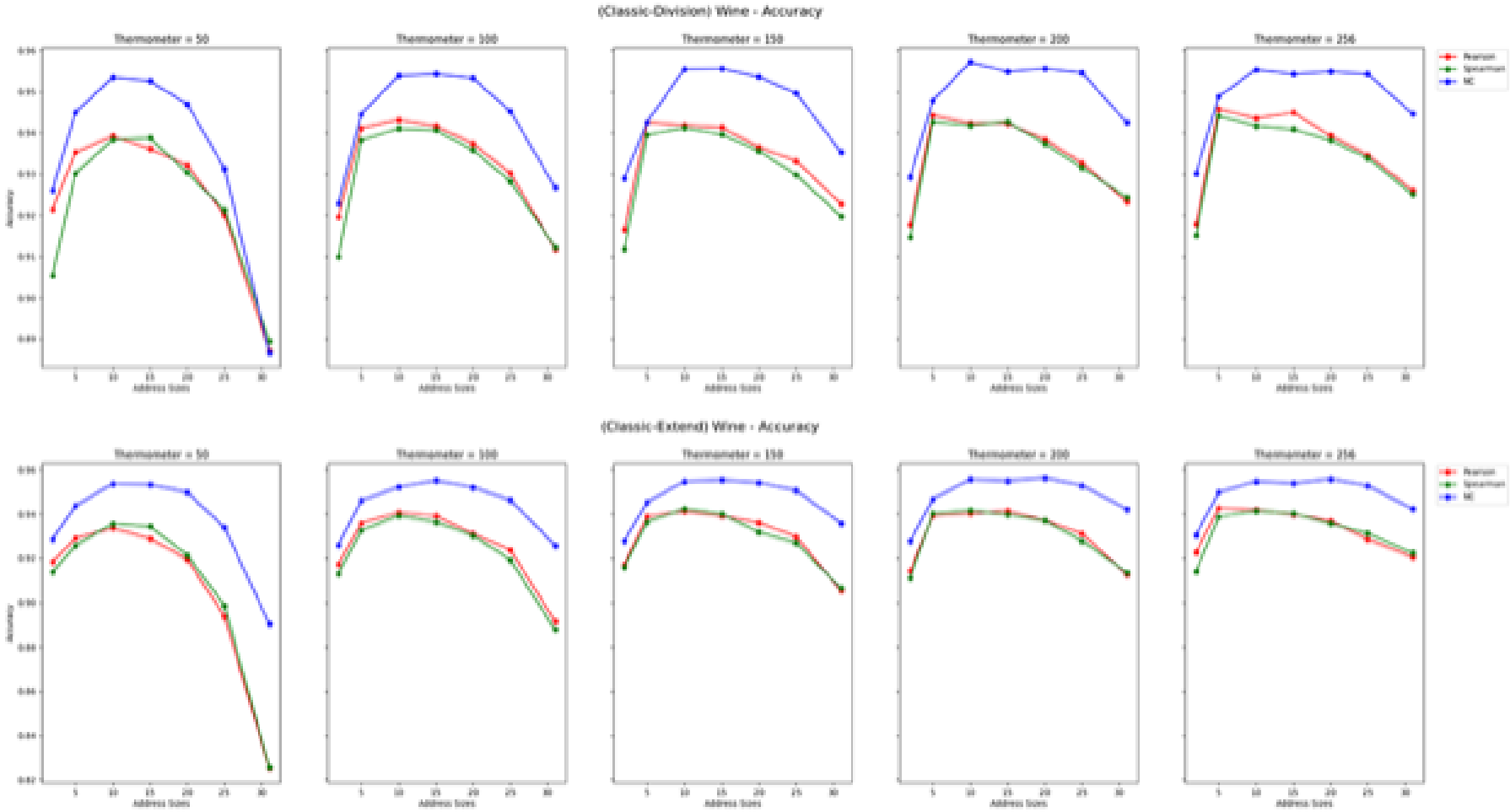

Figure D.22: Accuracy score comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wine dataset.

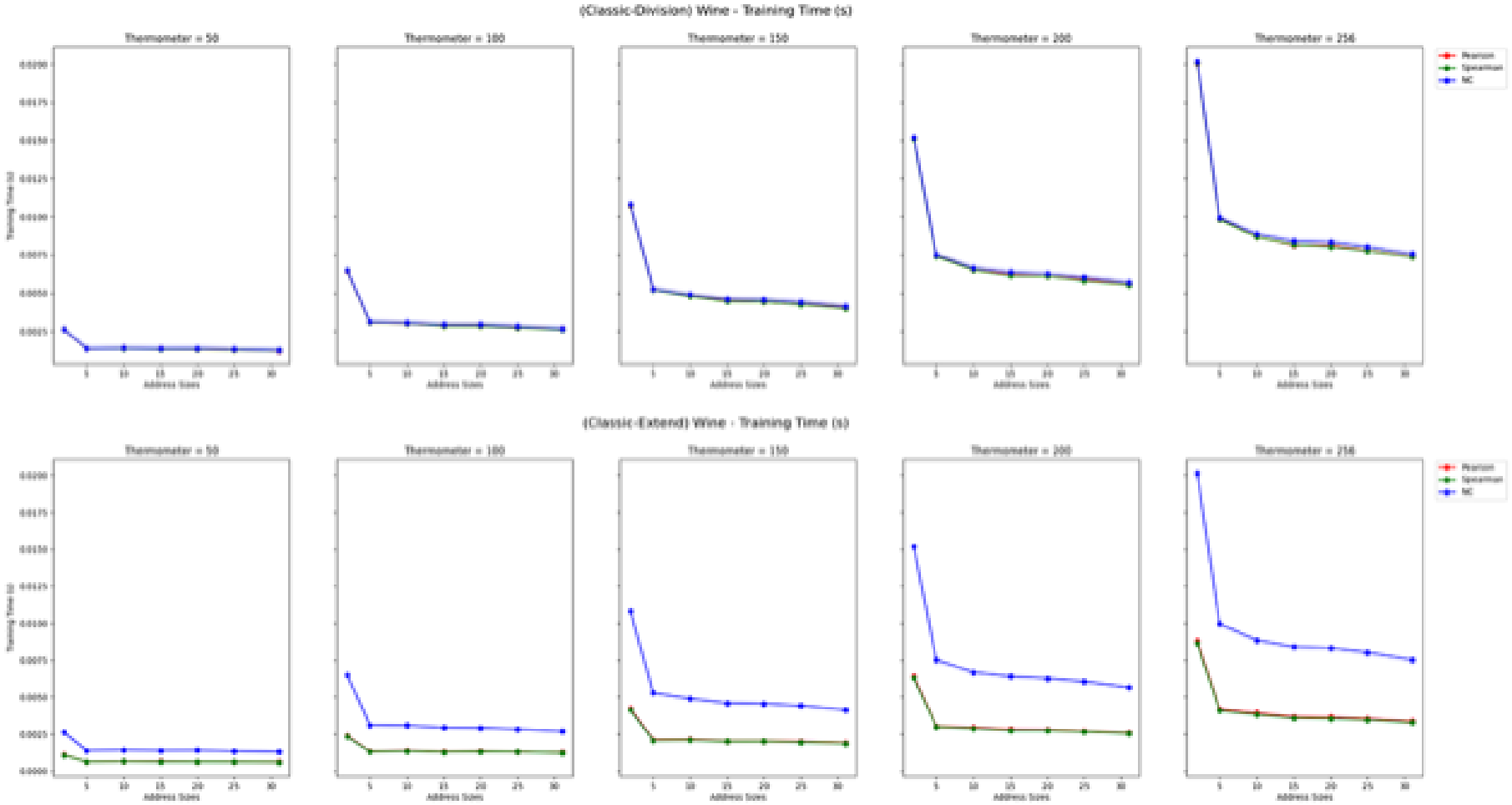

Figure D.23: Training time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wine dataset.

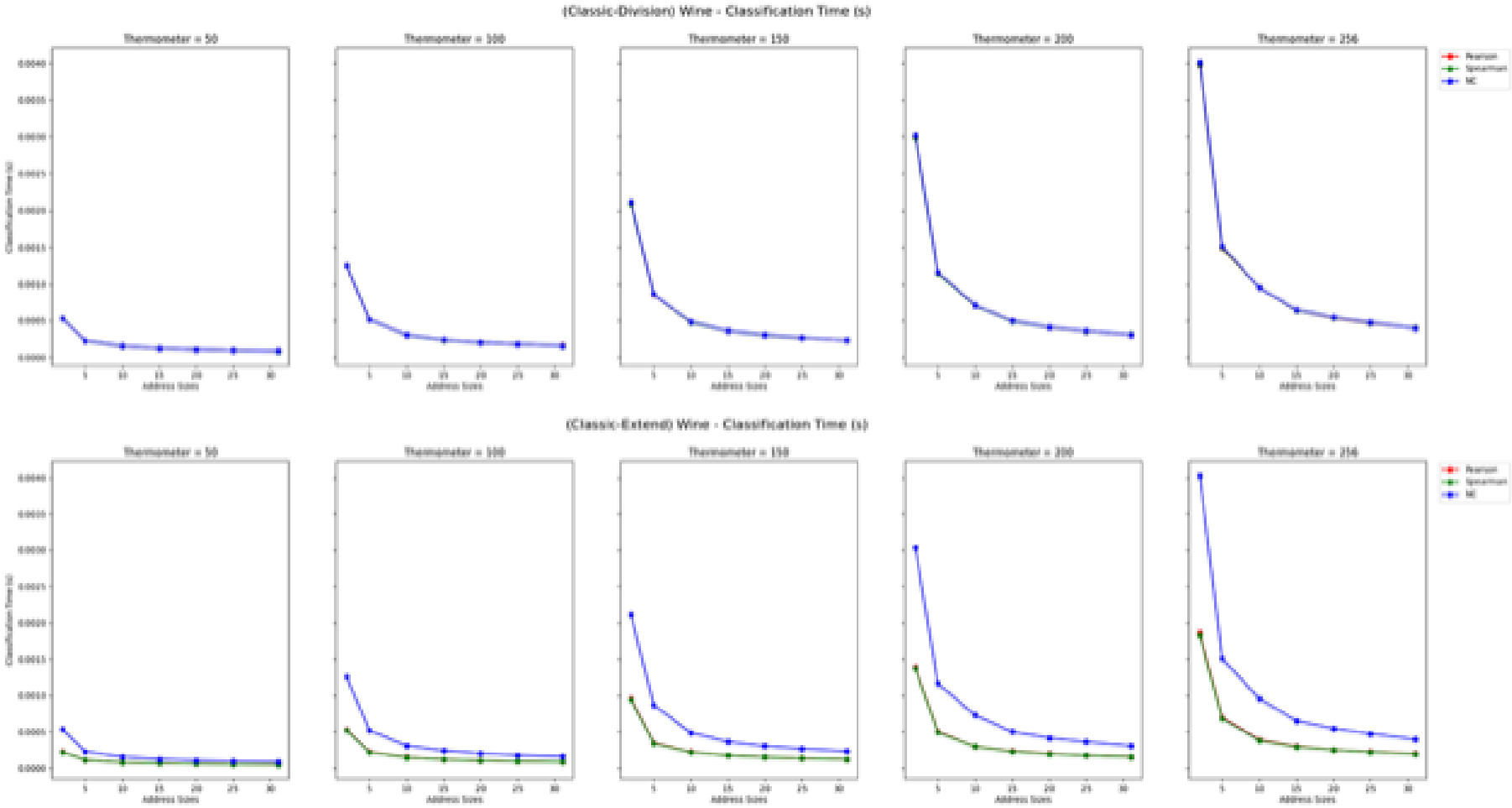

Figure D.24: Prediction time comparison between *division* (top) and *extend* (bottom) approaches using classic thermometer for the Wine dataset.

A first look at the results shows an undefined conclusion around the benefits of the proposed approaches. From the eight datasets, four presented better prediction performance, two presented no apparent difference and two had a decrease in accuracy. Table D.1 shows a summary of statistics regarding the datasets and the metrics of the correlation coefficients computed for them. The column *status* indicates whether the prediction performance increased (+), decreased (-), or did not show a significant difference (=). The table also indicates no apparent reason to detect when the approaches should be used or not.

Dataset	**Status**	**Lines**	**Features**	**Classes**	**Sum**	**Mean**	**StD**	**Min**	**Max**
Diabetes	+	768	8	2	1.66	0.21	0.12	0.07	0.47
House	+	506	13	-	5.59	0.43	0.15	0.18	0.74
Iris	+	150	4	3	3.10	0.78	0.22	0.41	0.96
Sonar	+	208	60	2	10.89	0.18	0.12	0.00	0.48
Cancer	=	683	9	2	6.73	0.75	0.10	0.53	0.87
Ionos.	=	351	33	2	6.07	0.18	0.14	0.00	0.52
Seed	-	210	7	3	2.49	0.36	0.17	0.02	0.58
Wine	-	178	13	3	6.40	0.49	0.23	0.05	0.85

Table D.1: Statistics regarding the datasets used in the experiments.

Considering the results for classification and prediction speed, it is worth noticing that by using the *division* approach there is no significant difference in using or not a correlation coefficient, since the inputs have always the same final size. That does not occur in the *extend* approach, where inputs that uses the correlation coefficients usually present shorter inputs which will result in faster processing.