Master's thesis

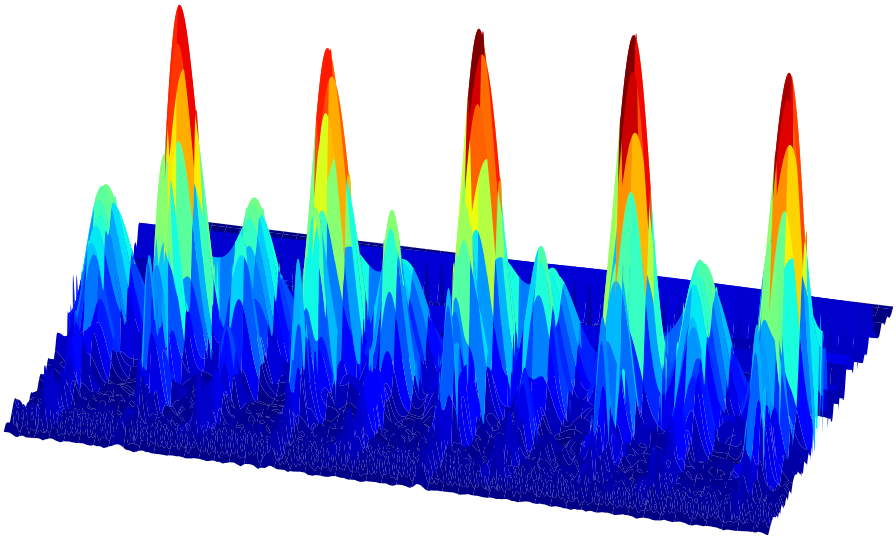
NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Øystein Stavnes Sletta

Mental Stress Classification from Phonocardiogram Signals

A wavelet Scattering Approach on Asynchronously Segmented PCG

Master's thesis in Cybernetics and Robotics
Supervisor: Marta Molinas
Co-supervisor: Amandeep Cheema
June 2023



3D spectrogram of recording snippet from self-collected Mental Stress Dataset.

NTNU
Norwegian University of
Science and Technology

Øystein Stavnes Sletta

Mental Stress Classification from Phonocardiogram Signals

A wavelet Scattering Approach on Asynchronously Segmented PCG

Master's thesis in Cybernetics and Robotics
Supervisor: Marta Molinas
Co-supervisor: Amandeep Cheema
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Preface

This thesis presents the work completed as a part of the course TTK4900 - Engineering Cybernetics, Master's Thesis written for the Department of Engineering Cybernetics at the Norwegian University of Science and Technology. My supervisor was Professor Marta Molinas (NTNU), and my co-supervisor was Dr. Amandeep Cheema (TIET).

I would like to thank my supervisors for their support throughout my thesis work, for answering questions, for helpful discussions, and for providing clarity. I want to thank NTNU high-performance computing group for giving me access to the Idun GPU cluster to run remote machine learning jobs. I want to acknowledge my gratitude to NTNU for funding PCG auscultation equipment and to my co-students in the Mental Health Crew Ida Marie Andreassen, Anne Joo Yun Marthinsen, Christian Sletten, and Ivar Tesdal Galtung for good teamwork while designing the dataset recording protocol, recruiting subjects, and doing the recordings.

Lastly, I want to thank my friends and family for their unconditional support during my time as a master's student at NTNU.

Declaration of Compliance

I, Øystein Stavnes Sletta, declare that this is an independent work according to the exam regulations of the Norwegian University of Science and Technology.

Trondheim, Norway June 12, 2023

Øystein Stavnes Sletta

Abstract

Stress detection from biosignals is a prosperous field of machine learning still in early development. The lack of public datasets with stress labels is a bottleneck for further research progress and the creation of new models that can be used by clinicians for healthcare purposes.

This thesis presents a new Multimodal Mental Stress dataset of time-synchronized phonocardiogram (PCG) data, electroencephalogram (EEG) data, and arithmetic test markers labeled with State-Trait Anxiety Inventory (STAI) and Subjective Stress Assessment (SSA) scores. The dataset's collection protocol and inclusion criteria are discussed, and a dataset analysis is conducted. The dataset is the first public dataset of its kind combining PCG data and stress labels.

Focusing strictly on PCG data and STAI stress labels, four machine-learning models based on wavelet scattering features and Extreme Gradient Boosting (XGB) or K-Nearest Neighbor (KNN) classifiers were made from the Mental Stress Dataset. The dataset recordings were separated into two binary label splits, STAI-30-30 and STAI-35-20, binning STAI-scores into either a high- or low-stress class. Models were optimized with respect to asynchronous segmentation lengths, scattering network parameters, and classifier hyperparameter values and evaluated in performance over a 10-fold cross-validation procedure. The KNN / STAI-30-30 model achieved a sensitivity of 0.8846, specificity of 0.7960, AUC of 0.8403, and accuracy of 0.8412. The XGB / STAI-30-30 model achieved a sensitivity of 0.9000, specificity of 0.8000, AUC of 0.8503, and accuracy of 0.8510. The KNN / STAI-35-20 model achieved a sensitivity of 0.9881, specificity of 0.7694, AUC of 0.8788, and accuracy of 0.9078. Lastly, the XGB / STAI-35-20 model achieved a sensitivity of 0.9796, specificity of 0.8801, AUC of 0.9298, and accuracy of 0.9430.

By only using PCG data and eliminating the use of multimodal recordings, the complexity and cost of using the model are reduced, making it suitable for home care, telemedicine, and rural healthcare centers.

The thesis introduces a new framework for removing systematic noise over specific frequency bands through wavelet scattering from the output feature vectors used in all model pipelines. The technique is referred to as feature exclusion and is reusable on any wavelet scattering feature extraction procedure from 1D signals where systematic noise is present in the data.

From the XGBoost feature importance metric, frequency ranges most important for stress state discrimination have been found. Scattering features connected to wavelets with center frequencies in the ranges 270-290 Hz and 380-390 Hz are found to be most important for the model's predictions. This explains specifically where we can see discrepancies in PCG data for different stress classes and will be very useful for designing similar machine-learning models in future research.

Sammendrag

Deteksjon av stress fra biosignaler er et fremtidsrettet felt innen maskinlæring, men er fremdeles i tidlig utviklingsfase. Mangelen på offentlige datasett med stressmarkeringer er en flaskehals for videre forskningsprogresjon og utvikling av nye modeller som kan brukes av klinikere for helsetjenesteformål.

Denne avhandlingen presenterer et nytt multimodalt Mental Stress datasett med tidsynkroniserte hjertelyd data (PCG), hjernebølge data (EEG) og markører fra aritmetisk test, annotert med State-Trait Anxiety Inventory (STAI) og Subjective Stress Assessment (SSA) poengsummer. Datasettets innsamlingsprotokoll og inklusjonskriterier diskuteres, og en datasettanalyse utføres. Datasettet er det første offentlige datasettet av sin type som kombinerer PCG-data og stressmarkeringer.

Ved å bare bruke PCG-data og STAI-stressmarkeringer ble det laget fire maskinlæringsmodeller fra Mental Stress datasettet. Modellene består av wavelet scattering egenskaper fra PCG dataen sammen med enten en Extreme Gradient Boosting (XGB) eller en K-Nearest Neighbor (KNN) klassifiserer. Datasettopptakene ble delt inn i to binære annoteringssplitter, STAI-30-30 og STAI-35-20, som begge separerte STAI-poengsummene i en høy- og en lav-stressklasse. Modellene ble optimalisert med hensyn til asynkrone segmenteringslengder, scattering-nettverkparametere og klassifiseringshyperparameterverdier, og evaluert med hensyn til ytelse gjennom en 10 gangers kryssvalideringsprosedyre. KNN / STAI-30-30-modellen oppnådde en sensitivitet på 0,8846, spesifisitet på 0,7960, AUC på 0,8403 og nøyaktighet på 0,8412. XGB / STAI-30-30-modellen oppnådde en sensitivitet på 0,9000, spesifisitet på 0,8000, AUC på 0,8503 og nøyaktighet på 0,8510. KNN / STAI-35-20-modellen oppnådde en sensitivitet på 0,9881, spesifisitet på 0,7694, AUC på 0,8788 og nøyaktighet på 0,9078. Til slutt oppnådde XGB / STAI-35-20-modellen en sensitivitet på 0,9796, spesifisitet på 0,8801, AUC på 0,9298 og nøyaktighet på 0,9430.

Ved å eliminere bruken av multimodale opptak ved bare å bruke PCG-data, reduseres kompleksiteten og kostnadene ved bruk av modellen, noe som gjør den egnet for hjemmeomsorg, fjernmedisin og helsetjenester på landsbygda.

Avhandlingen introduserer en ny metode for å fjerne systematisk støy over spesifikke frekvensbånd ved hjelp av wavelet scattering fra utdata-egenskaps-vektorene som brukes i alle modell-pipelinene. Teknikken kalles egenskapseksklusjon og kan gjenbrukes i enhver wavelet scattering-egenskaps ekstraksjonsprosedyre fra 1D-signaler der systematisk støy er til stede i dataene.

Basert på XGBoosts mål for viktighet av ulike features, har frekvensområder som er viktigst for å skille stressnivåer blitt funnet. Scattering-egenskaper knyttet til wavelets med senterfrekvenser i områdene 270-290 Hz og 380-390 Hz anses som mest betydningsfulle for modellens prediksjoner. Dette forklarer spesifikt hvor vi kan observere avvik i PCG-data for forskjellige stressklasser, og vil være svært nyttig for utformingen av lignende maskinlæringsmodeller i fremtidig forskning.

Table of Contents

Preface i

Abstract ii

Sammendrag iii

List of Tables vii

List of Figures xii

Abbreviations xiii

1 Introduction 1
- 1.1 Problem description . . . 1
- 1.2 Delimitations . . . 2
- 1.3 Contributions . . . 2

2 Background and Theory 4
- 2.1 Heart Sound Auscultation . . . 4
  - 2.1.1 The Heart Sounds . . . 5
  - 2.1.2 Cardiac Auscultation Foci . . . 6
  - 2.1.3 Synchronous and Asynchronous Segmentation . . . 6
- 2.2 Measuring Mental Stress . . . 7
  - 2.2.1 Self-report Measures . . . 7
  - 2.2.2 Physiology of Stress . . . 8
  - 2.2.3 Stress Detection from Biosignals . . . 9
- 2.3 Wavelets . . . 9
  - 2.3.1 Continuous Wavelet Transform . . . 10
  - 2.3.2 Scalograms . . . 11
  - 2.3.3 Scattering Networks . . . 12

2.4 Classification . . . 19
2.4.1 K Nearest Neighbor . . . 19
2.4.2 XGBoost . . . 20
2.4.3 Cross-validation . . . 21
2.4.4 Performance Measures . . . 22

3 Multimodal Mental Stress Dataset **24**
3.1 Delimitations . . . 24
3.2 Inclusion Criteria and Data Collection Protocol . . . 25
3.2.1 Heart Sound Auscultation . . . 25
3.3 Dataset Reformatting . . . 27
3.4 Dataset Analysis . . . 28
3.4.1 Labels . . . 28
3.4.2 Frequency Analysis . . . 30

4 PCG Machine Learning Methods and Classification Pipeline **35**
4.1 Label Augmentation . . . 35
4.2 Data Augmentation . . . 37
4.2.1 Resampling and Normalization . . . 37
4.2.2 Normalized Scattering Transform . . . 37
4.2.3 Noise Related Feature Exclusion . . . 38
4.3 Classification Pipeline . . . 40
4.3.1 The Stress Heuristic . . . 41
4.3.2 Feature Standardization . . . 41
4.3.3 Cross-Validation . . . 41
4.3.4 Final Model Evaluation . . . 44

5 Cross-validation Results and Final Evaluation **45**
5.1 Segmentation Length and Scattering Network Optimization . . . 45
5.1.1 STAI-30-30 (L, J, Q) Optimization . . . 45
5.1.2 STAI-35-20 (L,J,Q) Optimization . . . 47
5.1.3 (L, J, Q) Optimization Summary . . . 49
5.2 Hyperparameter Optimization . . . 50
5.2.1 STAI-30-30, KNN Classifier . . . 50
5.2.2 STAI-30-30, XGBoost Classifier . . . 51
5.2.3 STAI-35-20, KNN Classifier . . . 52
5.2.4 STAI-35-20, XGBoost Classifier . . . 53
5.2.5 Hyperparameter Optimization Summary . . . 55
5.3 Final Model Evaluations . . . 56
5.3.1 Feature Importance . . . 58

6 Discussion **62**
6.1 Mental Stress Dataset . . . 62
6.1.1 Inclusion Criteria . . . 62
6.1.2 Two Runs per Session . . . 63
6.1.3 PCG Auscultation setup and Systematic Noise . . . 64

6.1.4 Dataset Size . . . 64
6.1.5 STAI Binary Label Split . . . 64
6.2 Machine Learning . . . 65
6.2.1 Segmentation Choices . . . 65
6.2.2 Inter-subject Classifier . . . 66
6.2.3 Normalized Scattering Features . . . 67
6.2.4 Feature Exclusion . . . 68
6.2.5 Hyperparameter Optimization . . . 69
6.2.6 Final Model . . . 70
6.2.7 ROC AUC . . . 70
6.2.8 Important Frequencies for Mental Stress Detection from PCG . . . 71
6.3 Future Work . . . 71
6.3.1 Potential Pipeline Improvements . . . 71
6.3.2 Automatic Relabeling . . . 71
6.3.3 Joint Time-Frequency Scattering . . . 72
6.3.4 Heart-Brain Coherence . . . 72

7 Conclusion **73**

Bibliography **75**

Appendix **80**
A Mental Stress Dataset Consent Form . . . 80
B Arithmetic Test Equations with Answers . . . 83
C Default Hyperparameters for XGBoost, KNN and SVM Classifiers. 84
D Hyperparameter Optimization Trials . . . 86

List of Tables

2.1 Confusion matrix. 22

5.1 L,J,Q optimization summary for STAI-30-30 and STAI-35-20. 50
5.2 KNN classifier hyperparameter selections for each binary dataset split. . . 55
5.3 XGBoost classifier hyperparameter selections for each binary dataset split. 55
5.4 Final 10-fold cross validation model performances. Performance measures obtained from concatenated predictions on all 10 fold's validation sets. . . 56

7.1 Arithmetic test equation numbers, problems, and answers. 83
7.2 KNN Classifier — 50 hyperparameter combinations with Hyperopt for the STAI-30-30 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue. 86
7.3 XGBoost Classifier — 100 hyperparameter combinations with Hyperopt for the STAI-30-30 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue. 88
7.4 KNN Classifier — 50 hyperparameter combinations with Hyperopt for the STAI-35-20 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue. 92
7.5 XGBoost Classifier — 100 hyperparameter combinations with Hyperopt for the STAI-35-20 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue. 94

List of Figures

2.1 Normalized raw phonocardiogram. A random sample of recording 'P025_S002_002' from the self-collected Multimodal Mental Stress dataset (see chapter 3). S_1 and S_2 sounds are indicated. 5
2.2 Cardiac ausculation points: A=Aortic, P=Pulmonic, E=Erb, T=Tricuspid, M=Mitral [44]. 6
2.3 Scalogram $|W(a,b)|$ of a 3 second PCG recording displayed as a 2D heatmap (left) and a 3D surface plot (right). Random segment from recording 'P025_S002_002' in the self-collected Mental Stress Dataset (see chapter 3). 12
2.4 Different TF tiling methods. STFT-based tiling with linear center frequency distribution and fixed bandwidth (left), and wavelet tiling using a combination of STFT and CQT tiling (right). 14
2.5 Low pass filter $\hat{\phi}(\omega)$ (red) and wavelet sets $\hat{\lambda_1}(\omega)$ and $\hat{\lambda_2}(\omega)$ (blue) for $Q = (8,1)$, $J = 5$. The frequency axis is normalized, i.e cycles/sample. [44] 14
2.6 Example scattering output on one low-stress labeled recording (left, 'P017_S002_001') and one high-stress labeled recording (right, 'P009_S001_002') from the self-collected Multimodal Mental Stress Dataset (see chapter 3). Network invariance scale is set to $T = 2^J = 2^8 = 0.256$s at 1000Hz and octave frequency resolution set to $(Q_1, Q_2) = (10,1)$. For the second-order scattering coefficients, λ_1 and λ_2 indices are mixed along the vertical axis to provide a 2D image of the 3D tensor of coefficients. 16
2.7 Wavelet scattering network structure displayed with zeroth, first and second order scattering coefficients S_0x, S_1x, S_2x and first and second order modulus coefficients U_1x, U_2x. [44] 17
2.8 Scattering Coefficient values before (left) and after (right) normalization for an averaging window of size $T = 2^J$, 1000Hz PCG data. Scattering network parameters are $J = 13$, $Q = (20,1)$. 19
2.9 K-fold cross-validation procedure with the holdout test set. [44] 21

2.10 ROC curve with AUC shaded in gray. The dotted 'chance' line indicates the performance of a random guess. [44] 23

3.1 Lab Streaming Layer receiving stream "MyAudioStream" from AudioCapture. Mic directly connected to the computer's sound card [44]. 26
3.2 Eko Duo with connected aux cable [44]. 26
3.3 STAI score box plot displaying stress score distribution over the session and run numbers. S001 represents all recordings done in December 2022 (intended stressed state), and S002 represents all recordings done in January 2023 (intended low-stress/baseline state). 28
3.4 STAI score box plot displaying stress score distribution over session number and gender. 29
3.5 Age distribution over all recordings. 29
3.6 Joint plot comparing age to STAI score distribution for all recordings. Male recordings are marked in blue, and female recordings are marked in orange. 30
3.7 Power Spectral Density of all data in the dataset concatenated. Data is downsampled from 22050Hz to 1000Hz and normalized between [-1,1] before computing the PSD. 31
3.8 Wireless unmodified example recording through Eko's app interface. 120 seconds of 1000 Hz PCG data. The 50/60 Hz and harmonic peaks are non-existent. 32
3.9 Power Spectral Density comparing normal heart function Physionet dataset recordings to Mental Stress dataset recordings. All recordings are downsampled to 1000Hz before computing the PSD. An 8-second window of each signal is used. 33
3.10 Power Spectral Density comparing concatenated data from low (blue) and high (orange) stress-labeled classes with the STAI-35-20 split. All data concatenated is also included in the plot (green). 34

4.1 STAI score distribution pre-augmentation. 36
4.2 Dataset reduction from STAI labels to binary STAI labels. STAI-30-30 binary labels (left) and STAI-35-20 binary labels (right). 36
4.3 Binary label distribution of reduced datasets. STAI scores pre-augmentation (left), STAI-30-30 binary labels (left), and STAI-35-20 binary labels (right). 37
4.4 Frequency bands in the PCG data where scattering features are excluded for the machine learning pipeline ($f_\epsilon = 7$Hz). The graph (blue) shows the Power Spectral Density of all data in the Mental Stress Dataset concatenated. 39
4.5 Classification pipeline used for STAI-30-30 and STAI-35-20. The figure shows the parameter optimization steps, the 10-fold cross-validation procedure, and the final model evaluation with results. 40

5.1 (L, J, Q) optimization STAI-30-30 for XGBoost Classifier with default hyperparameters except tree_method='hist'. Average accuracy over 10-fold cross-validation. Chosen parameter combination for further optimization is marked in red. A hopeful combination for larger datasets is marked in pink. 46
5.2 Scattering representation of $(L, J, Q) = (30000, 4, 12)$. The original signal and Normalized scattering coefficients before and after feature exclusion are shown for 10-second segments from a low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress recording in STAI-35-20. Feature exclusion effectively removes the noise bands of the powerline frequencies visible as light blue horizontal lines in $\widetilde{S}_1$ before exclusion. Every L/T column spectral vector in $\widetilde{S}_1$ and $\widetilde{S}_2$ after feature exclusion is used as one classification data sample. 47
5.3 (L, J, Q) optimization STAI-35-20 for XGBoost Classifier with default hyperparameters except tree_method='hist'. Average accuracy over 10-fold cross-validation. Chosen parameter combination for further optimization is marked in red. A hopeful combination for larger datasets is marked in pink. 48
5.4 Scattering representation of $(L, J, Q) = (10000, 3, 12)$. The original signal and Normalized scattering coefficients before and after feature exclusion are shown for 10-second segments from a low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress recording in STAI-35-20. Feature exclusion effectively removes the noise bands of the powerline frequencies visible as light blue horizontal lines in $\widetilde{S}_1$ before exclusion. Every L/T column spectral vector in $\widetilde{S}_1$ and $\widetilde{S}_2$ after feature exclusion is used as one classification data sample. 49
5.5 STAI-30-30 Hyperopt parameter search illustrated over 50 model evaluations (trials). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $p = 1$ and $n\_neigbors = 18$. . 50
5.6 STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing number of trees ($n\_estimators$) and learning rate (eta). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $n\_estimators = 131$ and $eta = 0.598764$. 51
5.7 STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing regularization parameter $gamma$ and maximal tree depth (max_depth). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $gamma = 4.999442$ and $max\_depth = 15$. 51
5.8 STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing L1 regularization term reg_alpha and L2 regularization term reg_lambda. The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $reg\_alpha = 5$ and $reg\_lambda = 1.467703$. 52

5.9 STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing the subsample ratio of columns when constructing each tree ($colsample\_bytree$) and the minimum sum of instance weight (hessian) needed in a child (min_child_weight). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $colsample\_bytree = 0.990046$ and $min\_child\_weight = 0$. 52
5.10 Hyperopt parameter search illustrated over 50 model evaluations (trials). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $p = 2$ and $n\_neigbors = 4$. 53
5.11 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing number of trees ($n\_estimators$) and learning rate (eta). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $n\_estimators = 81$ and $eta = 0.829672$. 53
5.12 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing regularization parameter $gamma$ and maximal tree depth (max_depth). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $gamma = 2.005672$ and $max\_depth = 7$. 54
5.13 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing L1 regularization term reg_alpha and L2 regularization term reg_lambda. The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $reg\_alpha = 52$ and $reg\_lambda = 0.631727$. 54
5.14 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing the subsample ratio of columns when constructing each tree ($colsample\_bytree$) and the minimum sum of instance weight (hessian) needed in a child (min_child_weight). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $colsample\_bytree = 0.858025$ and $min\_child\_weight = 3$. 55
5.15 Final evaluation confusion matrices for the tested binary labels and classifiers. KNN, STAI-30-30 (a). XGBoost, STAI-30-30 (b). KNN, STAI-35-20 (c). XGBoost, STAI-35-20 (d). 57
5.16 ROC AUC results for the tested binary labels and classifiers. KNN, STAI-30-30 (a). XGBoost, STAI-30-30 (b). KNN, STAI-35-20 (c). XGBoost, STAI-35-20 (d). For STAI-35-20, fold 4 is removed since the validation fold contains no high-stress samples and would therefore be irrelevant for ROC AUC analysis. 58
5.17 Barplot showing feature/frequency importance relationship for STAI-30-30. The scattering coefficients' importance is shown with underlying first and second-order wavelet center frequencies are shown in Hz as (ξ_{λ_1}) for first-order coefficient features and $(\xi_{\lambda_1}, \xi_{\lambda_2})$ for second-order coefficient features. The most important frequencies for discerning mental stress states in STAI-30-30 are 287.76 Hz, 271.61 Hz, (271.61,43.75) Hz, and (384.11,43.75) Hz. 59

5.18 Barplot showing feature/frequency importance relationship for STAI-35-20. The scattering coefficients' importance is shown with underlying first and second-order wavelet center frequencies are shown in Hz as (ξ_{λ_1}) for first-order coefficient features and $(\xi_{\lambda_1}, \xi_{\lambda_2})$ for second-order coefficient features. The most important frequencies for discerning mental stress states in STAI-35-20 are 384.11Hz, 271.91Hz, (271.91,43.75)Hz, and (384.11,43.75)Hz. 59
5.19 Normalized scattering after feature exclusion $(L, J, Q) = (30000, 4, 12)$. The data displayed is: low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress from STAI-30-30. The four most important frequencies for STAI-30-30 are marked as dashed horizontal lines. 60
5.20 Normalized scattering after feature exclusion $(L, J, Q) = (10000, 3, 12)$. The data displayed is: low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress from STAI-35-20. The four most important frequencies for STAI-35-20 are marked as dashed horizontal lines. 61

Abbreviations

Abbreviation	Description
AC	Audio Capture
ACTH	Adrenocorticotropin
ADHD	Attention Deficit Hyperactivity Disorder
AUC	Area Under Curve
BVP	Blood Volume Pressure
CQT	Constant-Q Transform
CRH	Corticotropin
CWT	Continuous Wavelet Transform
DBP	Diastolic Blood Pressure
ECG	Electrocardiogram
EDA	Electro-dermal Activity
EEG	Electroencephalogram
EMD	Empirical Mode Decomposition
EMG	Electromyogram
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
GCN	Graph Convolutional Network
HPA	Hypothalamus-Pituitary-Adrenal
HR	Heart Rate
HRV	Heart Rate Variability
IFT	Inverse Fourier Transform
IMF	Intrinsic Mode Function
JTFS	Joint Time-Frequency Scattering
KNN	K Nearest Neighbor
LORO	Leave One Recording Out
LOSO	Leave One Subject Out
LSL	Lab Streaming Layer
LSTM	Long Short Term Memory
MAN	Manhattan (distance)
ML	Machine Learning
PCG	Phonocardiogram
PSD	Power Spectral Density
RBF	Radial Basis Function
ROC	Receiver Operating Characteristic

SBP	Systolic Blood Pressure
SCL	Skin Conductance Level
SCR	Skin Conductance Response
SF	Serialized Features
STAI	State-Trait Anxiety Inventory
STFT	Short-Time Fourier Transform
SVM	Support Vector Machine
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
WSC	Wavelet Scattering Coefficient
XGB	Extreme Gradient Boosting

1

Introduction

Mental stress is a prevalent issue impacting millions of people worldwide, and its adverse effects on mental health can manifest in various disorders, including depression, anxiety, attention deficit hyperactivity disorder (ADHD), and even severe neurological conditions. Early identification of mental stress is crucial to initiate prompt interventions and prevent long-term health consequences. However, detecting stress accurately can be challenging, as it is often subjective and difficult to quantify.

In recent years, machine learning and computing technology development have propelled rapid advancements in clinical application of the phonocardiogram (PCG). Originally used as a non-invasive diagnostic tool for detecting and monitoring cardiovascular diseases, PCG appears to have more diagnostic potential. This results from the underlying biological processes seen and estimated from the PCG. As this thesis will show, by leveraging machine learning techniques, PCG can now serve as a potential means to measure mental stress. For quick assessments, this can provide valuable insights into patients' well-being and stress levels for a medical professional, and in a long-term perspective, can significantly improve patient outcomes by early detection and thus prevent the development of more severe stress-induced medical conditions.

1.1 Problem description

This thesis is part of a larger project to design a multimodal automated stress detection system using electroencephalogram (EEG) signals and cardiac sound phonocardiography (PCG) signals. The work in the larger project is split between five master students. All the students have cooperated in collecting a new dataset for mental stress detection comprised of time-synchronized PCG, EEG, and Arithmetic test marker recordings from students attending NTNU labeled with the subject's stress state, age, and gender (see Chapter 3 for details). The dataset characteristics make it applicable for a wide range of analysis points, connecting PCG and EEG biosignals and bodily stress response. The time-synchronized

nature of the recordings also makes the dataset applicable for heart-brain coherence analysis.

The work related to machine learning in this thesis strictly covers PCG signals and stress response. It expands on the work of my specialization project covering feature extraction on PCG signals and cardiovascular disease classification [44]. Some background information and figures from the specialization project are reused in the thesis as they serve a similar context and are referenced. The objectives of the thesis are:

1. Explain and analyze state-of-the-art signal decomposition and processing techniques for feature extraction on PCG data.
2. Collect the remaining data of the Mental Stress Dataset consisting of PCG data from a baseline (not stressed) group and a stressed state group. Only the non-stressed state data is collected as part of the thesis.
3. Give a detailed explanation and analysis of the collected Mental Stress dataset.
4. Reformat the dataset. Make it easier to load and develop a protocol for binary labeling.
5. Design a reliable binary classification model using wavelet scattering features on PCG data from the self-collected Mental Stress dataset.

1.2 Delimitations

PCG recordings of the Mental Stress dataset unfortunately have systematic noise at 50/60Hz and their harmonics, making the analysis more limited. This is due to the nature of the recording setup as explained in chapter 3. A workaround method has been developed to aid the systematic noise explained in chapter 4.

Analysis of EEG, arithmetic test markers, and their relationship to the stress response is not discussed in this thesis. For research on this, please refer to my fellow student's thesis work [45].

1.3 Contributions

This thesis presents a novel framework, eliminating the need for Electrocardiography (ECG) signals-based referencing of Phonocardiography (PCG) signals for mental stress detection. This stand-alone PCG-based model uses a wavelet scattering approach on the data acquired from 26 healthy male and female subjects to detect mental stress.

For the best-performing model, PCG data is segmented asynchronously with 10-second segmentation lengths. An XGBoost classifier is trained on normalized wavelet scattering features from the segments and achieves a 94.30% 10-fold cross-validation accuracy on a 10-second input.

The elimination of multi-modal data acquisition and analysis makes this approach cost-efficient, and suitable for homecare, telemedicine, rural healthcare centers, and developing

economies.

Frequency ranges of the PCG signal, most important for discrimination of stress states have been found. The frequency ranges 270-290 Hz and 380-390 Hz has the highest importance in XGBoost's predictions. AI prediction models are often seen as a black box, however, knowing what features are most important for the classifier's decisions gives a partial explanation of why we get good predictions. This explainability and knowledge of important frequency ranges in the PCG for stress state discriminability makes designing future models less difficult.

A wavelet scattering systematic noise elimination method (feature exclusion) has also been implemented and is used as a mandatory step in the model pipeline. This method reduces feature dimensions, removing the frequency bands contaminated with noise from the feature output of the wavelet scattering procedure. The method is applicable for any wavelet scattering-based feature extraction method for 1D-signal data.

2

Background and Theory

When designing a binary classification pipeline for mental stress labeled phonocardiogram data, multiple factors must be considered. PCG signals are multi-component non-stationary signals highly susceptible to interference from background noise. The spectral content and timing of the different heart sounds give insight into various pathological conditions. This implies that a machine learning pipeline processing PCG data needs to construct robust and noise-resilient features without losing important time-frequency analysis information. Thus, the machine learning process involves carefully considering pre-processing steps, feature extraction techniques, classifier selection, and performance evaluation metrics.

This chapter aims to give a clear understanding of PCG signals, how they are measured, and their spectral content. Different stress detection metrics and previous successes with measuring mental stress from biomedical signals will be discussed. Thereafter topics relevant to obtain features rich in analysis information from PCG data will be introduced, including wavelet theory and wavelet scattering networks. Lastly, the theory provides information on some classifiers and important performance measures in biomedical research.

2.1 Heart Sound Auscultation

Phonocardiography (PCG) is a non-invasive diagnostic technique that records heart sounds and murmurs during the cardiac cycle. Using a digital stethoscope, the sounds made by the heart are converted into a phonocardiogram. A typical phonocardiogram displays four audible heart sounds labeled S1, S2, S3, and S4, produced by the snapping of closing heart valves. However, only S1 and S2 are easily identified in a phonocardiogram. Figure 2.1 shows a typical phonocardiogram with labeled heart sounds. PCG data can be measured from multiple locations around the heart. Five specific heart auscultation foci are normally used. Below, the heart sounds, and auscultation foci are explained in detail.

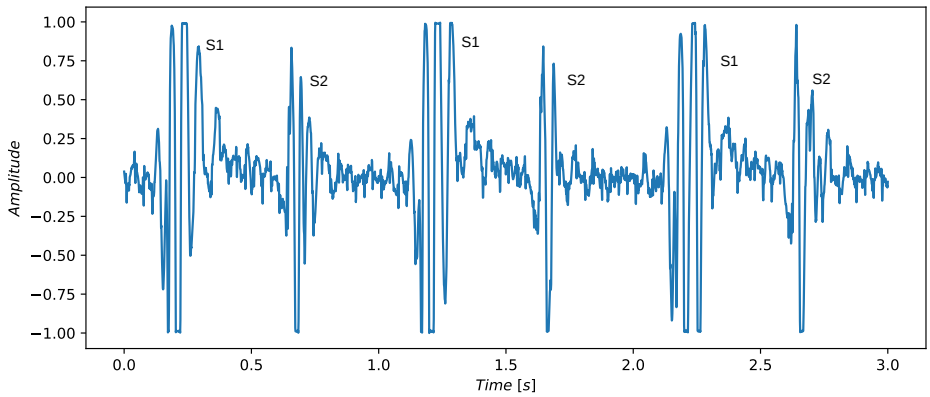


Figure 2.1: Normalized raw phonocardiogram. A random sample of recording 'P025_S002_002' from the self-collected Multimodal Mental Stress dataset (see chapter 3). S_1 and S_2 sounds are indicated.

2.1.1 The Heart Sounds

The S_1 sound occurs at the beginning of systole when the ventricles contract to pump blood into the aorta and pulmonary artery. This sound is due to vibrations caused by the mitral and tricuspid valves closing, preventing blood from flowing back into the atria. The closing occurs when the pressure in the ventricles becomes higher than in the atria. The most important abnormalities of S_1 relate to its intensity, where the sound can be abnormally loud or vary in intensity from beat to beat [32]. The power spectra of S_1 have been shown to have dominant frequencies in the 24-104 Hz range [5].

When the heart relaxes at the end of systole, the aortic and pulmonary valves close due to the ventricular pressure being lower than the arterial pressure. The closing of the two valves creates the S_2 sound, which has a minimum duration of 80ms and generally contains more high-frequency content than S_1. The power spectra of S_2 has been shown to have high values in the frequency range of 24-144Hz [5]. The most prominent diagnostic feature of S_2 is its splitting, which refers to how the aortic and pulmonic components of S_2 vary in timing during the cardiac cycle [32].

The third and fourth heart sounds S_3 and S_4, arise from the ventricle during diastole. The period when the heart is relaxed and filling with blood. S_3 occurs in early diastole from the rapid filling of blood in the ventricle and is typically heard in young healthy individuals. However, if heard in adults over 40 years of age, S_3 may indicate severe heart failure [32]. On the other hand, S_4 is heard at the end of diastole when ventricular filling suddenly slows down during the atrial contraction. S_4 is associated with an abnormally stiff ventricle due to either fibrosis or hypertrophy [32] and is not normally present at any age. Both the S_3 and S_4 sounds have a low pitch, ranging from 20 to 70 Hz, and are typically low in intensity.

2.1.2 Cardiac Auscultation Foci

During a physical examination, healthcare professionals use a diagnostic technique known as auscultation to listen to body sounds. Cardiac auscultation focuses on evaluating heart sounds and murmurs that may indicate abnormal cardiac function. The heart is auscultated at five foci or areas, which include the aortic, pulmonic, tricuspid, and mitral areas, as well as Erb's point. The first four foci are located above the corresponding heart valve positions, while Erb's point is located in the third intercostal space, close to the sternum below the pulmonary valve. Erb's point is named after Wilhelm Heinrich Erb, a German neurologist who identified this optimal point to auscultate the second heart sound (S_2). Figure 2.2 depicts the five auscultation foci.

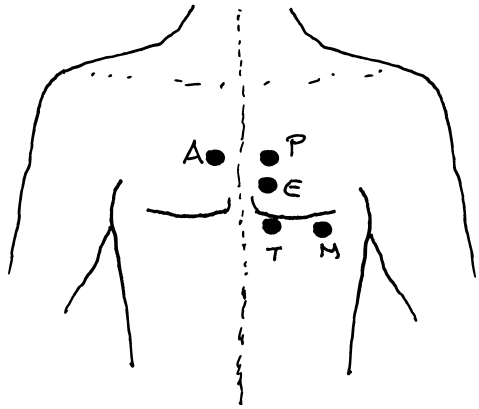


Figure 2.2: Cardiac ausculation points: A=Aortic, P=Pulmonic, E=Erb, T=Tricuspid, M=Mitral [44].

2.1.3 Synchronous and Asynchronous Segmentation

Typically, to do machine learning on data, the data samples need to be uniform in size, and for time series data, similar in structure to discern patterns. This task is challenging for PCG data since it is highly non-linear, and the timing varies between different cardiac sound events. However, much effort has been put into making algorithms for heart sound segmentation. Specifically, these methods most commonly aim to find S1–S1 intervals in the PCG time series data, making each segment entail one heart cycle. The segmentation procedure based on cardiac events is called synchronous segmentation and is achieved in multiple ways. Ricke et al. [40] uses a parallel ECG recording and a hidden Markov model to segment the PCG signal. Other studies focus on segmenting the signal with only the signal itself available. This has been done with an entropy and envelogram method in [35], a zero frequency filtering method in [38], and with bidirectional LSTM networks with attention in [17]. Binary classifiers designed for synchronously segmented PCG data from the PhysioNet CinC 2016 challenge dataset have become highly accurate, with a newer model based on spectrograms, transfer learning, and graph convolutional networks (GCN) introduced by Rezaee et al. [39] achieving a remarkable 99.4% accuracy.

While synchronous segmentation is very effective, it involves an extra preprocessing step in a prediction model, which can be computationally expensive and introduce unnecessary model complexity. Synchronously segmented data also results in segments of different lengths, inducing the need for an extra preprocessing step involving padding signals to equal lengths or using statistical or spectral features yielding uniform feature sizes for a machine learning interface. Therefore, newer research often focuses on models that can handle unsegmented data utilizing the simpler approach of asynchronously segmenting the signals. This involves splitting the PCG recordings into sections of similar lengths without considering the timing of underlying cardiac events. As a result, the structure and timing of underlying cardiac events will differ for all segments, and subsequently, features constructed for these segments will have to be somewhat invariant to these differences to find important class characteristics. With recent breakthroughs in feature extraction techniques, models for asynchronously segmented data have become very accurate. Li et al. [25] proposed a wavelet, entropy, and fractal feature-based method with a Twin SVM classifier and achieved a classification accuracy of 90.4% on the PhysioNet CinC 2016 challenge dataset. An ensemble model using cepstral, spectral, and envelogram features was proposed in [42], achieving an accuracy of 92.47%, and a spectrogram and transfer learned CNN-based model was proposed in [23] achieved an accuracy of 95,75% on the same dataset. The most recent articles significantly improve on this by using a new feature extraction technique based on deep scattering networks. Singh et al. [41] proposes a wavelet scattering and KNN classifier model reaching an accuracy of 97.82% on the PhysioNet dataset. This shows that asynchronous segmentation models are catching up to the performance previously seen in state-of-the-art synchronous segmenting models.

2.2 Measuring Mental Stress

Stress is a complex and multi-dimensional construct that manifests in the body as a threat from adverse intrinsic or extrinsic forces, real or imagined, termed stressors. These forces are counteracted by physiological and behavioral responses known as the stress response, aiming to maintain body equilibrium (homeostasis) [14]. Stress is challenging to measure accurately. While stress is a universal experience, how it manifests and affects individuals can vary widely, making it difficult to develop a standardized measurement tool. Measuring stress is essential for understanding its negative impact on mental health, including anxiety, depression, and other neurological disorders.

2.2.1 Self-report Measures

Capturing stress through self-report measures alone can be challenging, as individuals may not accurately report their stress levels due to various reasons such as social desirability bias, lack of self-awareness, or difficulty in identifying and expressing emotions. One commonly used tool for measuring non-disorder-specific anxiety is the State-Trait Anxiety Inventory (STAI) questionnaire, which assesses both state (S-anxiety) and trait anxiety (T-anxiety) levels. Understanding the theoretical framework and psychometric properties of the STAI is crucial for researchers and clinicians in assessing stress levels and designing effective interventions for stress-related disorders.

The State-Trait Anxiety Inventory (STAI) is a widely recognized self-report questionnaire. The STAI is based on the theoretical framework that anxiety is a multidimensional construct that consists of both situational (state) and dispositional (trait) components. The STAI was developed by Charles Spielberger in 1970 [51] and has been widely used in clinical and research settings, with the first publication surpassing 45000 citations. The inventory consists of two 20-item subscales, one assessing S-anxiety and the other T-anxiety. Each item on the STAI asks respondents to rate the intensity of anxiety symptoms such as "I am tense", "I feel calm", or "I feel content" on a 4-point Likert scale. The final state and trait anxiety scores are calculated by summing the point scores from each of the 20 questions, giving a final score from 20-80 for each form.

The STAI has demonstrated good psychometric properties, including high internal consistency and test-retest reliability. The inventory has been translated into several languages and used in numerous studies to assess anxiety levels in various populations, including children, adolescents, and adults [22]. Research has shown that the STAI is a valid and reliable measure of anxiety levels. For instance, a meta-analysis conducted by Barnes et al. 2002 found that the STAI had good diagnostic accuracy in identifying anxiety disorders. Similarly, a study conducted by Vitasari et al. 2011 on engineering students in Malaysia showed strong statistical reliability and validity of both S-anxiety and T-anxiety measures.

2.2.2 Physiology of Stress

The stress response is a physiological process that occurs when a person perceives an upcoming threat. This response involves a cascade of physiological processes that help the body adapt to the stressor. These processes eventually serve homeostasis by regulating body functions such as temperature, heart activity, blood pressure, respiration, and glucose levels, which are essential for survival through a range of environmental conditions.

The physiological response to stress involves two primary pathways, namely the Hypothalamus-Pituitary-Adrenal (HPA) axis and the Sympathetic-Adrenal Medullary (SAM) pathway [18]. The HPA axis combines neural and endocrine components involving the Hypothalamus, Pituitary gland, and Adrenal cortex, while the SAM pathway includes the sympathetic component of the autonomic nervous system and the adrenal medulla.

The HPA axis is activated in the hypothalamus, either directly or through neurotransmitters sent by the amygdala. Assessing the severity of stimulus, the hypothalamus releases corticotropin hormone (CRH) into the anterior lobe of the pituitary gland. This triggers the release of adrenocorticotropin (ACTH) into the bloodstream. ACTH facilitates the synthesis and secretion of the adrenaline, noradrenaline, and cortisol hormones in the adrenal cortex (small organ on top of each kidney). These three stress hormones increase glucose levels, providing an immediate energy resource for muscles and nerve cells in order to serve adaptation to stressors.

The SAM pathway is activated parallel to the HPA axis, where the brain directs the sympathetic part of the autonomic nervous system, increasing arousal, alertness, and body mobilization. The core process in this pathway involves the secretion of the hormones adrenaline and noradrenaline from the adrenal medulla. These hormones bind to specialized receptors throughout the body and brain and trigger the rapid mobilization of car-

diovascular, musculoskeletal, gastrointestinal, nervous, and endocrine systems comprising the "fight-or-flight" response. The main physiological effects of SAM activation involve increased heart rate, respiratory rate, blood pressure, muscle tension, diversion of blood flow from the internal organs to the brain and muscles, perspiration, and pupil dilation [16].

2.2.3 Stress Detection from Biosignals

Using known physiological responses to stress, there have been many successful attempts at identifying stress from biosignals. In a comprehensive review conducted by Giannakakis et al. 2019, a wide variety of biosignal patterns caused during stress conditions was explored. Several biosignal measures such as electrocardiography (ECG), electroencephalography (EEG), blood volume pressure (BVP), electromyography (EMG), electrodermal activity (EDA), respiration, skin temperature, pupil diameter, eye activity, and speech recordings were reviewed. According to the review, several biosignals present consistent patterns in their efficiency at discriminating stress conditions. Heart rate (HR), typically measured with ECG, was the most prominent feature, increasing significantly during stress. Heart rate variability (HRV) also appeared to give prominent features, with a consistent low-frequency to high-frequency ratio increase with stress. Skin conductance response (SCR) and level (SCL), measured from EDA, also typically increased with stress. For brain activity, the EEG alpha symmetry index value had a consistent pattern of reducing during high stress. Other consistent variations with high stress across studies include increased systolic blood pressure (SBP) and diastolic blood pressure (DBP), increased respiration rate, and increased voice pitch.

Existing research on stress detection from PCG signals is limited; however, the available research shows promising results. Cheema and Singh 2019 presents a framework for psychological stress detection using entropy-based features from intrinsic mode functions (IMFs) derived from synchronously segmented PCG signals decomposed with empirical mode decomposition (EMD). The synchronous segmentation is achieved by using ECG gating to find S1 peaks and is thus dependent on having a multimodal recording of time-synchronized PCG and ECG data. Using this method, a 10-fold cross-validation accuracy of 93.14% is achieved with a least squares support vector machine classifier. A recent article, [12] 2023 proposes a similar approach, using IMFs of synchronously segmented PCG data, where in contrast, the features are derived from 2D and 3D phase space reconstructions of the IMFs. This results in a five-fold cross-validation accuracy of 97.14% with a radial basis function support vector machine classifier. These results provide optimism for PCG-based stress detection.

2.3 Wavelets

Wavelets are a class of mathematical functions characterized by their short duration, zero mean, and finite energy. They serve as a good set of basis functions from which many other waveforms can be generated. Modern wavelet theory was introduced in the 1980s and 1990s by mathematicians such as Jean Morlet, Yves Meyer, and Ingrid Daubechies;

however, the first mentions of the word wavelet date back to 1910, with Haar orthogonal basis functions [34]. A framework was developed for decomposing signals into a series of wavelets, each representing a different frequency range and time duration. Since then, wavelets have become an important tool in signal processing and have found widespread use in fields such as audio and image compression, medical signal processing [21], and financial analysis.

There are several wavelet families, including the Haar wavelet, the Daubechies wavelet, and the Morlet wavelet, among others. Each type has its own unique properties and is suited for specific applications.

Recently, wavelet techniques combined with machine learning have significantly impacted the biomedical research industry. This includes utilizing continuous wavelet transform (CWT) and synchrosqueezing-based EEG, ECG, and PCG signal features in conjunction with deep convolutional networks. Additionally, there has been a recent breakthrough with wavelet scattering networks, which construct shift-invariant features that are stable to time-warping deformations. This section will focus on the necessary wavelet theory and provide a detailed description of wavelet scattering networks. It is assumed that the reader has a fundamental understanding of the Fourier transform and its time-frequency windowed variant, the short-time Fourier transform (STFT).

2.3.1 Continuous Wavelet Transform

Like the STFT, the continuous wavelet transform (CWT) utilizes inner products to measure the correlation between a signal and an analyzing function. While the STFT utilizes windowed complex exponentials ($f(t) = w(t)e^{j\omega t}$) as analyzing functions, the CWT uses wavelets ($f(t) = \psi(t)$). By comparing the signal with the wavelet at different scales and time shifts, the CWT generates a function of two variables, much like the STFT generates a function of frequency and time. "Continous" in CWT does not refer to the type of signals the transform can handle but to the continuous scaling and transformation of the analyzing wavelet. Defining the scale $a > 0$ and translation b, the CWT is expressed analytically as:

$$W_\psi x(a,b) = |a|^{-1} \int_{-\infty}^{\infty} x(t)\bar{\psi}\left(\frac{t-b}{a}\right) dt, \tag{2.1}$$

where the bar over the mother wavelet function indicates the complex conjugate. By continuously varying a and b, the CWT coefficients, $W_\psi x(a,b)$, are given. It is worth noting that the 2-D representation of a 1-D signal is redundant. The CWT outputs a real-valued function of scale and position for a real-valued signal and a complex-valued function of scale and position for a complex-valued wavelet.

When implemented digitally, a discrete form of (2.1) is used:

$$W_\psi x[a,b] = |a|^{-1} \sum_{n=0}^{N-1} x[n]\bar{\psi}\left[\frac{n-b}{a}\right], \tag{2.2}$$

which is mathematically equivalent to convolving the input signal with an array of wavelet filters possessing varying lengths. However, implementing the CWT like this would be computationally inefficient because of the fine time-frequency tiling expected from the CWT. Thankfully, by using Parseval's theorem, the computational complexity of the CWT can be vastly reduced.

Applying Parseval's theorem to (2.1) gives

$$W_\psi x(a,b) = \frac{1}{2\pi}\int \hat{x}(\xi)\overline{\widehat{\psi_{a,b}}(\xi)}\mathrm{d}\xi. \tag{2.3}$$

Now utilizing the timed-scaling and time-shifting properties of the wavelet in the Fourier domain:

$$\widehat{\psi_{a,b}}(\xi) = \hat{\psi}(a\xi)\mathrm{e}^{-ib\xi} \tag{2.4}$$

substitution of (2.4) into (2.3) gives:

$$W_{\psi,a}x(b) = \frac{1}{2\pi}\int \hat{x}(\xi)\overline{\hat{\psi}(a\xi)}\mathrm{e}^{ib\xi}\mathrm{d}\xi \tag{2.5}$$

or written in discrete form

$$W_{\psi,a}x[b] = \frac{1}{K}\sum_{k=0}^{K-1} \hat{x}[k]\overline{\hat{\psi}[ak]}\mathrm{e}^{i2\pi bk/K}, \tag{2.6}$$

which describes the CWT as an inverse Fourier transform (IFT) of $\hat{x}[k]\overline{\hat{\psi}[ak]}$. Thus, the translation parameter b is no longer a transform dependency, drastically reducing computational complexity since we can compute an entire row of the CWT at the time. As $\hat{x}[k]$ can be calculated beforehand, only three steps are needed per scale a:

1. Generate $\overline{\hat{\psi}[ak]}$
2. Calculate $\hat{x}[k]\overline{\hat{\psi}[ak]}$
3. Evaluate IFT to get $W_{\psi,a}x[b]$

Now, since the CWT can be expressed as an IFT one can use FFT and IFFT algorithms to achieve an $O(Nlog_2N)$ complexity [6].

2.3.2 Scalograms

A scalogram is a visual representation of the continuous wavelet transform (CWT) coefficients. It provides a valuable tool for analyzing the time-frequency content of non-stationary signals, as it allows for the visualization of the signal's energy distribution over different scales and times. Scalograms are particularly useful for analyzing signals with varying frequency content over time, such as speech signals or biomedical signals like

EEG, ECG, and in this case, PCG. By analyzing the scalogram, one can identify important features in the signal, such as peaks, ridges, and oscillatory patterns, which can provide valuable insights into the underlying physiological or biological processes.

A common way of displaying the scalogram is by taking the modulus of the CWT, $|W_\psi(a,b)|$. This removes any complex function values. Figure 2.3 displays the scalogram in 2D and 3D of a 3-second PCG recording downsampled to 1000Hz and normalized. The S1 and S2 peaks and their frequency ranges are clearly visible.

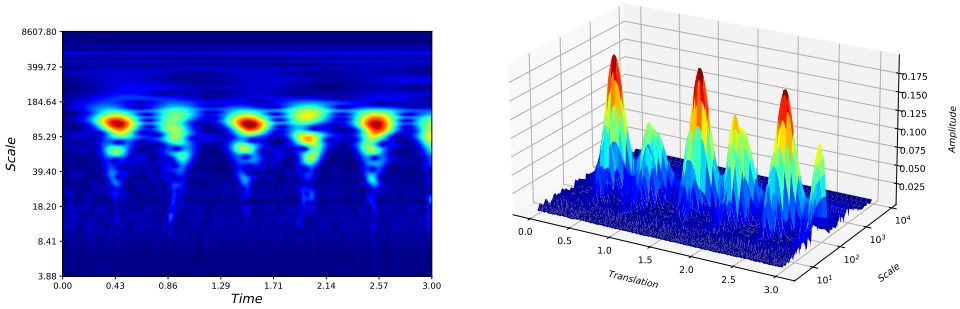


Figure 2.3: Scalogram $|W(a,b)|$ of a 3 second PCG recording displayed as a 2D heatmap (left) and a 3D surface plot (right). Random segment from recording 'P025_S002_002' in the self-collected Mental Stress Dataset (see chapter 3).

2.3.3 Scattering Networks

Wavelet scattering networks are equivalent deep networks consisting of a cascade of convolution, modulus, and low-pass operators. Scattering networks generate representations that exhibit time-shift invariance, noise resilience, and stability against time-warping deformations [4]. For machine learning purposes in classification tasks like musical genre classification, acoustic scene classification [26], spoken digit recording classification [49], and EEG seizure classification [21], scattering coefficients have proven useful, obtaining state-of-the-art performance on limited datasets. In scattering networks, the most commonly used wavelets are Morlet wavelets. These are formed by modulating a Gaussian function with a complex exponential.

As the goal of this thesis is to analyze PCG signals, only 1D scattering will be covered in this section, neglecting the rotational properties of the transform present in higher dimensions. However, scattering generalizes beautifully to higher dimensions and has been used in applications like image texture classification [48], and chemistry [15].

Filter Bank Implementation

The process of dilation, which is the key step in constructing wavelet filter banks, involves scaling the mother wavelet's size with respect to the hyperparameters chosen for the network. A set of filters with varying center frequency tiles the part of the time-frequency (TF) plane relevant to the input signal. TF plane is defined by the frequencies ranging

from zero to the Nyquist frequency (half the sampling frequency of the signal), and the length of the signal being analyzed.

Complete tiling of the TF plane can be achieved in multiple ways. Kymatio [3], a Python package for scattering networks, implements filter banks with two hyperparameters. The number of octaves J, which sets the invariance scale of the network $T = 2^J$, and the number of wavelets per octave Q_i per layer i, which sets the quality factor. By scaling the mother Morlet wavelet, a filter bank Λ_i for each network layer i is constructed with Q_i wavelets per octave. The wavelets in the filter banks have a center frequency ξ and a center frequency index λ. The set of center frequency indices of filterbank i is referred to as the index set $\lambda_i \in \Lambda_i$. Wavelets are bandpass filters with $\hat{\psi}(0) = 0$. For center frequency $\xi > 0$, a scaled wavelet is written in time and frequency domain as

$$\psi_\xi(t) = \xi\psi(\xi t) \text{ and hence } \hat{\psi}_\xi(\omega) = \hat{\psi}(\frac{\omega}{\xi}). \tag{2.7}$$

The scale sequence $\{a^j\}_{1 \leq j \leq J}$, for $a > 1$ is chosen based on the given quality factor Q as $a = 2^{1/Q}$. Audio processing applications require higher frequency resolution with typically $Q \geq 8 \implies a \leq 2^{1/8}$ [29]. The center frequency of each wavelet has inverse properties with scale as $\xi_{\lambda_{i,j}} = a^{-j}$ for a given scale index j. Thus we can rewrite (2.7) as:

$$\psi_{\xi_{\lambda_{i,j}}}(t) = a^{-j}\psi(a^{-j}t) \text{ and hence } \hat{\psi}_{\xi_{\lambda_{i,j}}}(\omega) = \hat{\psi}(\frac{\omega}{a^{-j}}), \quad j \in \mathbb{Z} \tag{2.8}$$

The highest frequencies of the filterbank ($\xi_{\lambda_{i,j}} \geq 2\pi Q/T$) are tiled by constant-Q transform (CQT) with Q wavelets per octave, which implies that the center frequencies defined as

$$\xi_{\lambda_{i,j}} = a^{-j}, \quad j \in \mathbb{Z}, \quad a = 2^{1/Q}. \tag{2.9}$$

The frequential support of $\hat{\psi}_{\xi_{\lambda_{i,j}}}(\omega)$ is centered in $\xi_{\lambda_{i,j}}$ with a bandwidth in the order of Q^{-1}. The energy of $\psi_{\xi_{\lambda_{i,j}}}(t)$ is concentrated around 0 within an interval with size $2\pi Q/\xi_{\lambda_{i,j}}$. To prevent this interval from overreaching T, the lowest frequencies of the TF plane, $\xi \in [0, 2\pi Q/T)$, are tiled by short-time Fourier transform (STFT). STFT tiling makes wavelet filters with linear center frequency distribution and fixed frequency bandwidth $2\pi/T$. This contrasts the CQT tiling, which produces wavelet filters with logarithmic center frequency spacing with a fixed center frequency to bandwidth ratio [20]. This split enables complete tiling with a finite amount of filters and avoids discretization artifacts. Figure 2.4 shows the time-frequency tiling discrepancy between the STFT and the wavelet STFT/CQT tiling methods.

The Kymatio tiling method makes a time/frequency resolution tradeoff by favoring frequency resolution at low frequencies (i.e over larger time supports) and temporal resolution at high frequencies (i.e lower time supports).

A low pass filter $\phi_J(t) = 2^{-J}\phi(2^{-J}t)$ with size 2^J, frequency bandwidth $2\pi/2^J$, and center frequency $\lambda = 0$ is also created. This is used for averaging in the network. Time shifts under the invariance scale ($T = 2^J$) are time-averaged in the scattering coefficient output, making a time-shift invariant representation. Figure 2.5 shows an example two-layer filterbank constructed with with Kymatio.

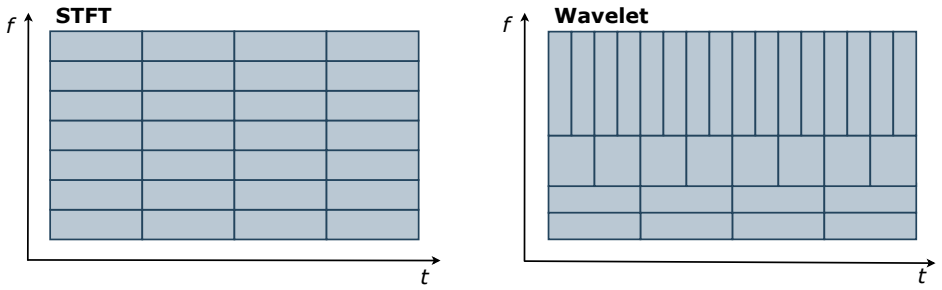


Figure 2.4: Different TF tiling methods. STFT-based tiling with linear center frequency distribution and fixed bandwidth (left), and wavelet tiling using a combination of STFT and CQT tiling (right).

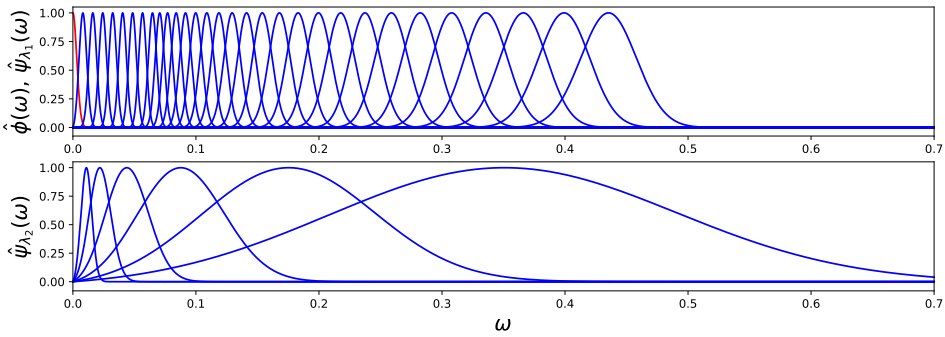


Figure 2.5: Low pass filter $\hat{\phi}(\omega)$ (red) and wavelet sets $\hat{\lambda_1}(\omega)$ and $\hat{\lambda_2}(\omega)$ (blue) for $Q = (8, 1)$, $J = 5$. The frequency axis is normalized, i.e cycles/sample. [44]

Scattering

With the filterbanks constructed, the scattering coefficients can be calculated. The scattering transform is calculated of an input signal with non-overlapping time windows of size 2^J, centered at time points $t = k2^{J-1}$, $k \in \{2n-1, n \in \mathbb{N}\}$. This implies that the temporal length of the scattering output of an input signal x with length L is $L/2^J$. Define the wavelet modulus transform W_i of order i and input x as:

$$W_i x = (x * \phi_T(t), |x * \psi_{\xi_{\lambda_i}}(t)|) = (x * \phi_T(t), U_i x), \quad \lambda_i \in \Lambda_i \tag{2.10}$$

where λ_i are the center frequency indices of the set Λ_i, the filter bank of network layer i. Zeroth-order scattering coefficients are found by convolving a low-pass filter with the input signal

$$S_0 x = x * \phi_T. \tag{2.11}$$

These represent the lowest frequency energy of the original signal and, in energy analysis

terms, contain little information. The higher frequencies lost are recovered by a wavelet modulus transform

$$W_1 x = (x * \phi_T(t), |x * \psi_{\xi_{\lambda_1}, Q_1}(t)|) = (x * \phi_T(t), U_1 x), \quad \lambda_1 \in \Lambda_1. \tag{2.12}$$

The wavelet operator performs a CWT on x with wavelet filters described by Λ_i and octave frequency resolution Q_1. Taking the modulus produces the scalogram $U_1 x$, which removes the complex phase but does not lose information because the temporal variation of the multiscale envelopes is kept [4]. As a result of the redundancy of the wavelet transform, the wavelet modulus transform is invertible for multiple wavelet types, which indicates that the signal is entirely reconstructable from the scalogram $U_1 x$ [29].

The First-order scattering coefficients $S_1 x$ and the second-order modulus coefficients are obtained by cascading the wavelet modulus transform on the previous-order scalogram

$$W_2 U_1 x = (U_1 x * \phi_T, |U_1 x * \psi_{\xi_{\lambda_2}, Q_2}|) = (S_1 x, U_2 x), \quad \lambda_2 \in \Lambda_2, \tag{2.13}$$

where $S_1 x(t, \xi_{\lambda_1})$ is a time-averaged scalogram representing the first-order scattering coefficients with octave frequency resolution Q_1. The first-order scattering coefficients are invariant to time shifts under the scale T by averaging all U_1 coefficients over this time duration. The averaging operation also improves stability against time-warping deformations. $U_2 x(t, \xi_{\lambda_1}, \xi_{\lambda_2})$ is a three-dimensional tensor representing the second order modulus coefficients indexed at time t, acoustic log-frequency $\xi_{\lambda_{1,j}}$, and modulation frequency $\xi_{\lambda_{2,j}}$. The tensor is the result of convolving every row of the scalogram $U_1 x$ with every second-order filter $\psi_{\xi_{\lambda_{2,j}}}$ and taking the modulus. By convolving with unaveraged wavelet modulus coefficients, high-frequency information lost by averaging S_1 is recovered.

Scattering coefficients of second order are given by

$$S_2 x = U_2 x * \phi_T = |U_1 x * \psi_{\xi_{\lambda_2}, Q_2}| * \phi_T. \tag{2.14}$$

The convolution in time between the tensor $U_2 x(t, \xi_{\lambda_1}, \xi\lambda_2)$ and low pass filter $\phi_T(t)$ produces a time averaged tensor $S_2 x(t, \xi_{\lambda_1}, \xi_{\lambda_2})$ of second order scattering coefficents with octave resolution Q_2.

In theory, the scattering transform can be computed at infinite depth by cascading the wavelet modulus transform, yet the number of scattering coefficients increases combinatorially with depth. As explored experimentally in [4], for smaller invariance scales $T = 23$ms, $99,3\%$ of the input signal fractional energy is absorbed by the first- and second-order scattering coefficients. However, with $T = 1.5$s, order three coefficients carried 24.7% of the fractional energy. Thus, higher-order coefficients carry more energy with increasing averaging scales and become more important.

By further cascading wavelet modulus operators, higher-order coefficients can be found. The nth order scattering coefficients can be calculated as:

$$S_n x = |||x * \psi_{\xi_{\lambda_1}}| * \cdots| * \psi_{\xi_{\lambda_n}}| * \phi_J, \quad \lambda_i \in \Lambda_i, \quad i = 1, 2, \ldots, n. \tag{2.15}$$

An example scattering transform of a PCG signal is shown in figure 2.6. We see that S_0 contains marginal information for the chosen scale as it simply represents a low-pass

filtered version of the input. S_1 coefficients contain the most energy whilst providing high-frequency resolution because the chosen $Q_1 = 10$ implies the use of wavelet filters with low-frequency bandwidth and large temporal support. S_2 restores a lot of the temporal context lost by averaging for S_1 by employing a smaller octave resolution of $Q_2 = 1$, which implies wavelet filters with smaller temporal support and larger frequency bandwidths.

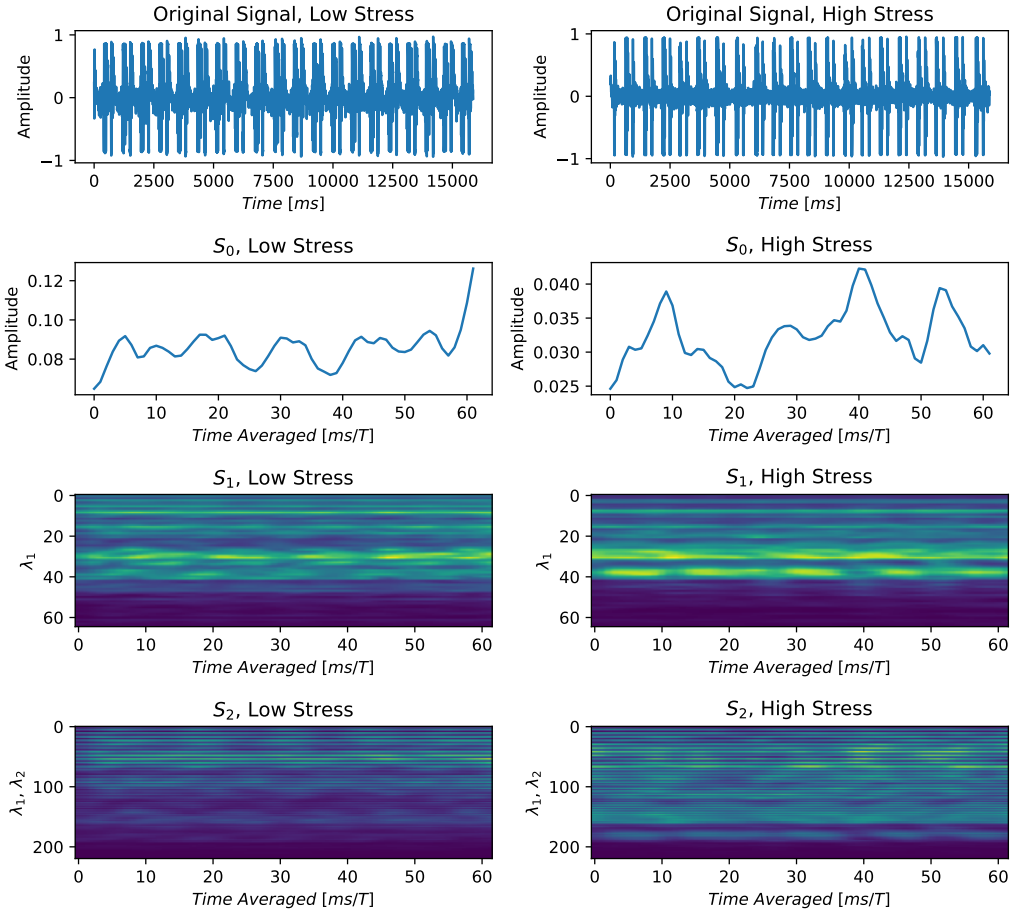


Figure 2.6: Example scattering output on one low-stress labeled recording (left, 'P017_S002_001') and one high-stress labeled recording (right, 'P009_S001_002') from the self-collected Multimodal Mental Stress Dataset (see chapter 3). Network invariance scale is set to $T = 2^J = 2^8 = 0.256$s at 1000Hz and octave frequency resolution set to $(Q_1, Q_2) = (10, 1)$. For the second-order scattering coefficients, λ_1 and λ_2 indices are mixed along the vertical axis to provide a 2D image of the 3D tensor of coefficients.

The structure of the wavelet scattering network is summarized in figure 2.7. A scattering

network shares many similarities with convolutional neural networks by having convolving weights, nonlinearities, and averaging operations. The difference is that the weights of the scattering network are predefined, non-trained wavelets, and the features are extracted from every network layer. Having pre-trained weights makes the wavelet scattering transform fast, as it only requires one forward pass through the network to extract all features.

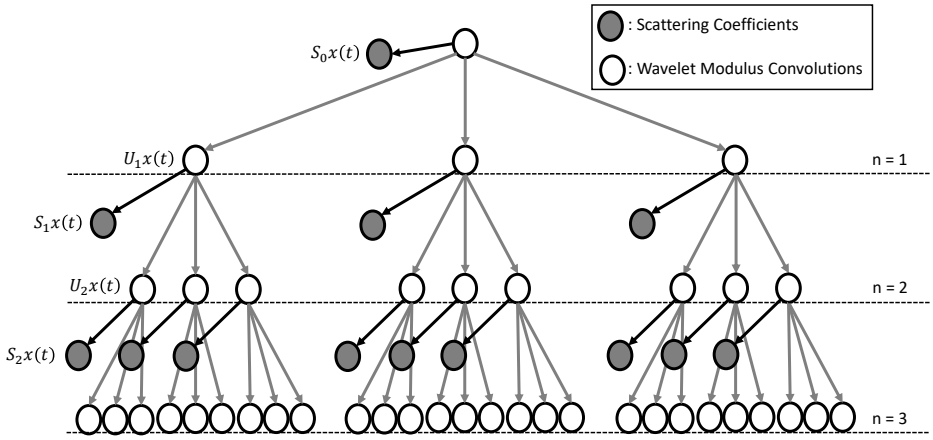


Figure 2.7: Wavelet scattering network structure displayed with zeroth, first and second order scattering coefficients S_0x, S_1x, S_2x and first and second order modulus coefficients U_1x, U_2x. [44]

Time-shift Invariance and Time-warp Stability

The goal of transforming the input data is to obtain features with valuable analysis information. We want features that balance inter-class variability against intra-class variability. That is, features that differ for different classes, but not for the same class. We aim for sparse, stable, coherent, and robust representations with properties that reduce variability to find inter-class similarities but preserve intra-class variability to separate classes. Scattering networks provide these kinds of representations with time-shift invariance and stability to time warping-deformations imposed by averaging at scale T.

For signal variations at scale $c \ll T$, the scattering coefficient output is similar. This time-shift invariance can be expressed as

$$S(x(t-c)) \approx S(x(t)), \quad c \ll T. \tag{2.16}$$

Variations on time scales lower than T are lost for each order of the transform but fully recovered in higher orders. This is a variability-reducing property of the scattering transform. The value of T is important since it decides what analysis information is wanted. With a large invariance scale, we get a good frequency representation of what happens on that time interval, but specific events are blended together over the larger scale, making

them less distinguishable. Setting a smaller time scale will better preserve the order of events but make the frequency detail of each event more prone to noise and reduce the invariance of the scattering coefficients. Without averaging, there is too much variability, and a classifier won't effectively be able to separate classes. In a machine learning environment, setting the scale larger will reduce the number of training instances for the classifier, making it more prone to overfitting. On the other hand, choosing a small T may yield many instances that lack stimuli relevant to fitting it to a class, hence increasing the bias and chance of underfitting.

Stability to small-time warps is another important property of the scattering transform. Suppose that x is just not translated in time, but time-warped to give $x_\tau(t) = x(t - \tau(t))$ with $|\tau'(t)| < 1$. A representation $\Gamma(x)$ is said to be stable to deformations if the Euclidean norm $\|\Gamma(x) - \Gamma(x_\tau)\|$ is small when the deformation is small. The deformation is measured by the supremum norm $\|\tau'(t)\|_\infty = sup_t|\tau'(t)|$. When the representation is the scattering transform, stability is formally defined as a Lipschitz continuity condition. There exists $C > 0$ such that for $x(t)$ and all τ satisfying $\|\tau'(t)\|_\infty < 1$ and $\|\tau(t)\|_\infty \ll T$

$$\|Sx - Sx_\tau\| \leq C\|\tau'(t)\|_\infty \|x\|. \tag{2.17}$$

Where the constant C is proportional to the octave frequency resolution Q. The choice of Q can thus be considered a tradeoff between time-warp stability and frequency resolution (higher Q gives less time-warp stability). (2.17) implies that time-warping deformations are locally linearized by the scattering transform. The proof for the stability of the scattering transform to small diffeomorphisms is published in [29]. Stability to time-warping deformations makes a classifier less likely to isolate training instances from similar processes based on rate changes. For example, if we train a classifier to recognize words from audio data, a word spoken at different speeds could be more likely to yield two different class labels if the features we classify lack time-warp stability.

Renormalization

As proposed in [4], a way to further increase the invariance of the scattering coefficients is to perform renormalization. First-order coefficients are renormalized to retain local amplitude information below a certain time scale:

$$\widetilde{S}_1 x\left(t, \xi_{\lambda_1}\right) = \frac{S_1 x\left(t, \xi_{\lambda_1}\right)}{|x| * \phi(t)} \tag{2.18}$$

The lowpass filter $\phi(t)$ can have a larger scale than the one chosen for the scattering transform. Amplitude information of $S_1(x)$ contained under this scale will be retained in $\widetilde{S}_1$, creating invariance only to amplitude changes over larger intervals.

The second-order coefficients can be renormalized by "parent" coefficients of the previous order:

$$\widetilde{S}_2 x\left(t, \xi_{\lambda_1}, \xi_{\lambda_2}\right) = \frac{S_2 x\left(t, \xi_{\lambda_1}, \xi_{\lambda_2}\right)}{S_1 x\left(t, \xi_{\lambda_1}\right)}. \tag{2.19}$$

This operation decorrelates the different order coefficients making the normalized coefficients less redundant. $\widetilde{S}_2 x(t, \xi_{\lambda_1}, \xi_{\lambda_2})$ also provide high-resolution spectral information through interferences as discussed in [28].

The normalized scattering transform of the second order is thus formulated as:

$$\widetilde{S}x(t, \xi_{\lambda_1}, \xi_{\lambda_2}) = \begin{pmatrix} \widetilde{S}_1 x(t, \xi_{\lambda_1}) \\ \widetilde{S}_2 x(t, \xi_{\lambda_1}, \xi_{\lambda_2}) \end{pmatrix}. \tag{2.20}$$

In the context of classification, normalizing the scattering coefficients makes a considerable difference in the size relationships between the first and second-order coefficients. Figure 2.8 shows how the renormalization affects the coefficient values. The second-order coefficients are scaled in proportion to the first-order coefficients making them easier to discriminate with for classifiers.

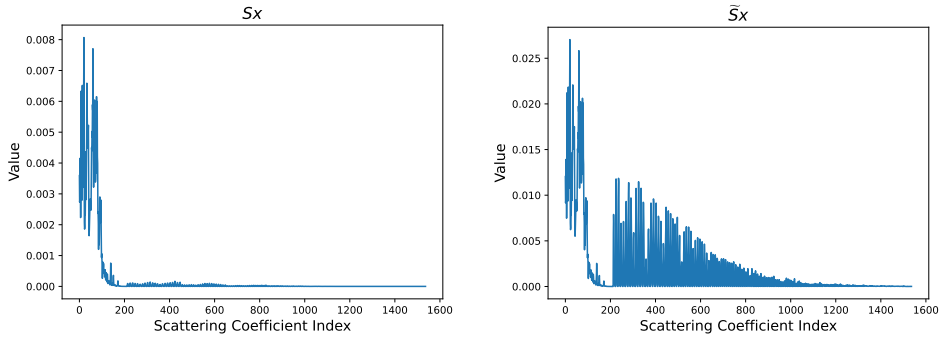


Figure 2.8: Scattering Coefficient values before (left) and after (right) normalization for an averaging window of size $T = 2^J$, 1000Hz PCG data. Scattering network parameters are $J = 13$, $Q = (20, 1)$.

2.4 Classification

2.4.1 K Nearest Neighbor

The k-nearest neighbor classifier is one of the most straightforward classifiers and has the attribute of being sensitive to the local structure in the data. The input is the feature vector that needs to be classified, and the number of neighbors, k, to include in a majority vote is specified [41].

To fit the classifier, the feature vectors and labels of the training set are stored. When predicting a sample from the test set, the labels of the k nearest samples in the training set corresponding to the input sample are included in a majority vote. The feature vector is then classified as the most common label among the k neighbors.

To compute the distance between two points, various measures are used. Several commonly used distance measures are versions of the Minkowski distance. Consider n-dimensional

points $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$. The Minkowski distance is defined as:

$$D_M(X, Y, p) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}. \tag{2.21}$$

Choosing $p = 1$ gives the Manhattan/Cityblock distance function, and $p = 2$ gives the Euclidean distance.

2.4.2 XGBoost

XGBoost (Extreme Gradient Boosting) is a powerful and versatile machine learning algorithm used widely by data scientists, achieving state-of-the-art performance in many machine learning challenges, including regression, classification, and ranking problems.[13]. It belongs to the family of gradient-boosting algorithms, which are ensemble methods that combine the predictions of multiple weak learners to create a robust and accurate classifier.

At its core, XGBoost is based on gradient boosting, which involves iteratively adding weak learners, called classification and regression trees (CART), to the ensemble. These trees differ from decision trees in that the tree's score is associated with each leaf, which gives richer interpretations than if each leaf only had a class value (making CART trees applicable to regression tasks). The CARTs are added one at a time, with each added learner attempting to correct the mistakes made by the previous ones and further reducing the objective function. The objective function is the sum of a loss term and a regularization term. The regularization term is introduced as part of the objective function to prevent overfitting and control the model complexity. This will discourage the model from fitting noise in the training data, optimizing the bias-variance tradeoff. For a mathematical introduction to the objective function used, please refer to [13]. This sequential nature of boosting allows XGBoost to learn complex patterns and relationships in the data by focusing on the instances that were previously misclassified.

XGBoost has attained a great reputation for its focus on efficiency and scalability. XGBoost incorporates parallelization techniques and takes advantage of multi-core processors to accelerate training and prediction processes. This makes the algorithm computationally efficient and capable of handling large-scale datasets with high-dimensional feature vectors and many data points.

XGBoost offers a wide range of hyperparameters that can be tuned within a standard cross-validation procedure to leverage the maximum power of XGBoost both in terms of computation times and classification results. Tree booster parameters such as the number of CARTs ($n\_estimators$), learning rate (eta), minimum loss function reduction to make a node split ($gamma$), the maximum depth of the CARTs (max_depth), the minimum sum of weights for observations required in a child (min_child_weight), regularization parameters (reg_lambda and reg_alpha), and many more are important for CART functionality. Parameters relevant for defining the optimization objective include the choice for loss function ($objective$), validation data metric ($eval\_metric$), and the random number seed ($seed$). Careful tuning of booster and objective hyperparameters can significantly impact the performance and generalization ability of the XGBoost classifier. Setting the

n_thread parameter allows for parallel processing over multiple threads, making computations faster.

2.4.3 Cross-validation

Cross-validation is a technique used to evaluate a classifier's performance and estimate the model's ability to generalize on an independent dataset. When designing a classifier for a dataset, the standard procedure is to split the data into training and testing sets. The training set is used for cross-validation, while the testing set is held out and used for the final model evaluation. This testing set is often referred to as a holdout set [24].

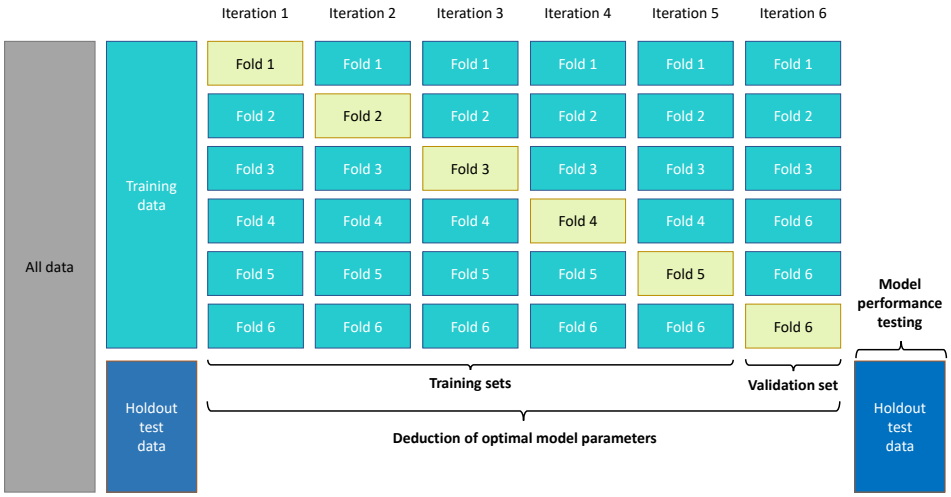


Figure 2.9: K-fold cross-validation procedure with the holdout test set. [44]

One of the most commonly used techniques for cross-validation is k-fold cross-validation. The procedure is illustrated in figure 2.9. The input training dataset is divided into k folds for each algorithm iteration. One of the folds is selected as the validation set, and the remaining k-1 folds are defined as the training set for that iteration. A classifier is trained on the training set and evaluated on the validation set. The training and validation sets are switched for each iteration. When the classifier has been validated on all k folds, the performance measures for each iteration are averaged to evaluate the model's overall performance on the dataset. This way, by testing different model hyperparameters, optimal model hyperparameters can be found using cross-validation and a grid search of varying hyperparameter values. Many open-source Python libraries, such as SciKit Learn, can be used for this purpose.

Compared to the traditional approach of splitting data into training and validation sets, this method of testing classifiers is superior as it allows for utilizing all data points not included in the holdout set for both training and validation.

2.4.4 Performance Measures

When making predictions with a binary classifier, it is common to represent the results using a confusion matrix, as illustrated in Table 2.1. This matrix is a tool for visualizing the classifier's predictions on the test set compared to the actual values. The predictions are classified as either true positive (TP), true negative (TN), false positive (FP), or false negative (FN).

Table 2.1: Confusion matrix.

		Ground Truth	
		Abnormal	Normal
Predicted		$(+)$	$(-)$
Abnormal	$(+)$	TP	FP
Normal	$(-)$	FN	TN

To evaluate the performance of binary classifiers, it is necessary to utilize measures that reveal the relationships between TPs, TNs, FPs, and FNs. The specific metrics required may vary depending on the context of use. Here are some important metrics that can be used for this purpose.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.22}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{2.23}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.24}$$

In medical research, sensitivity and specificity are essential metrics that assess the accuracy of a test in predicting the presence or absence of a particular condition. Sensitivity, also known as the true positive rate (TPR), is the ratio of individuals with the condition correctly identified by the test. On the other hand, specificity, also known as the true negative rate (TNR), is the proportion of individuals without the condition correctly identified by the test [2]. Accuracy is a measure of the overall performance of the test, determined by the proportion of correct predictions made by the test. These metrics are critical for evaluating the reliability and effectiveness of medical tests.

Receiver Operating Characteristic Analysis

Binary classifiers can be evaluated through a receiver operating characteristic (ROC) analysis, which visualizes the tradeoff between sensitivity and specificity [50].

The false positive rate (FPR), which is the proportion of individuals without the condition that the classifier incorrectly identifies, can be defined as

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{TNR} = 1 - \mathrm{Specificity}. \tag{2.25}$$

The ROC space is defined as all possible combinations of points $(\mathrm{FPR}, \mathrm{TPR}) \in [0, 1] \times [0, 1]$. A $(\mathrm{FPR}, \mathrm{TPR})$ pair gives a point on the ROC curve, and the curve is drawn by sliding the threshold/cut-off value to determine whether the probabilistic output of the classifier is considered positive or negative. This results in a ROC curve with as many points as the number of chosen threshold values. The ideal point on the ROC curve is (0,1), indicating 100% sensitivity and 100% specificity. The classifier's performance improves as the curve approaches the upper left corner. The area under the ROC curve (AUC) is a commonly used performance measure. An AUC of 0.5 indicates random performance, while an AUC above 0.9 is considered excellent. Figure 2.10 illustrates an example ROC curve with the AUC shaded in gray.

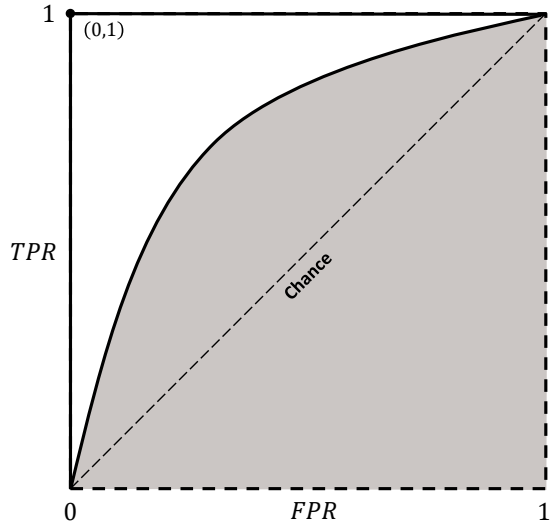


Figure 2.10: ROC curve with AUC shaded in gray. The dotted 'chance' line indicates the performance of a random guess. [44]

3

Multimodal Mental Stress Dataset — Data Acquisition and Analysis

This chapter covers information on a mental stress dataset collected in collaboration with four other master students at NTNU as part of the specialization project and master thesis work. The dataset comprises 83 PCG recordings and 108 EEG recordings from 28 students attending NTNU, 16 male and 12 female, in the age group of 20–28 years old. Recordings were done in two sessions, S001 (December 2022) before students' institute examinations and project deliveries and S002 (January 2023) after Christmas vacation. For each recording session, subjects did two five-minute recordings: Run 001 and Run 002. Run 001 recordings were done with no external stimuli, while run 002 recordings were done with an arithmetic test to induce stress. Complementary to the data, mental stress scores and participant info are provided as labels, including SSA and STAI (S-Anxiety) scores, age, and gender. In addition, all Run 002 recordings have complementary question-and-answer data with exact timestamps.

The data collection for the Mental Stress dataset is anonymized and approved by Norsk Senter for Forskningsdata (NSD).

3.1 Delimitations

Due to the technical difficulties of recording two modalities simultaneously, some PCG recordings are missing. Thus, depending on the desired use case (PCG only, EEG only, multimodal data), the dataset has an unequal amount of valid recordings. In this thesis, only PCG data is used, but as the end goal is to utilize both modalities in a classifier, only recordings with both valid PCG and EEG data are used. The part of the data usable in a multimodal system (time-synchronized EEG and PCG) consists of 83 5-minute recordings from 26 participants (16 male, and 10 female). Protocol for the EEG measurement setup is

outside the scope of this thesis. For information on this, please refer to Christian Sletten's thesis [45].

3.2 Inclusion Criteria and Data Collection Protocol

To obtain appropriate separation in the response variable (mental stress), students attending exams in high-workload studies were chosen as a target group. Recruitment for the recordings was done orally during university lecture breaks and through friends. To be part of the dataset, the students had to confirm the following inclusion criteria:

1. Participant is a student at NTNU
2. Participant has an exam or project delivery scheduled after S001
3. No diagnosed heart or neurological disease
4. Not using any medication affecting heart rate or brain wave function
5. Not diagnosed with any mental illness

Participants signed a consent form confirming they suited the inclusion criteria and volunteered to share their data. The consent form is attached in appendix A.

The dataset collection process involved two recording sessions. The first session (S001, December 2022) was conducted prior to the exam(s) or project deliveries, with the aim of obtaining high-stress state recordings. The second session (S002, January 2023) was conducted after the Christmas vacation, with the objective of capturing a low-stress state. This recording timeline was chosen to provide maximal differences in the response variable, mental stress. Two 5-minute runs were recorded per session: Run 001 and Run 002. During Run 001, PCG and EEG recordings were conducted in the absence of external stimuli, with the subject seated quietly in a chair. During Run 002, an arithmetic test was administered to the subject while recording. The arithmetic questions and answers from the subject are also provided for all Run 002 recordings. More details on the design of the arithmetic test are detailed below.

For both recording sessions, test subjects registered through an online form and selected a suitable time for the recording. On the day of the recording, the subjects provided their consent by signing a consent form (see Appendix A). Subsequently, the participants were required to complete State-Trait Anxiety Inventory (S-Anxiety) forms before each recording (one form before each run), assessing their perceived mental stress and anxiety levels. The participants were also asked to rate their perceived stress state on a scale of 1-10 before each recording. This stress assessment will be referred to as Subjective Stress Assessment (SSA). The information collected for each participant, including STAI-S, SSA, age, and gender, was logged.

3.2.1 Heart Sound Auscultation

For capturing heart sounds, the device of choice was the Eko Duo Digital Stethoscope from Eko. The device has an aux output fed directly into a computer as a microphone

input (figure 3.2). This enables real-time PCG signal acquisition (and exceeds the recording time limit in the Eko app). To be able to time-sync PCG, EEG, and arithmetic test marker recordings with the multimodal setup, the different data sources were streamed to Lab Streaming Layer (LSL). LSL syncs the streamed inputs in time and concatenates all inputs to one .xdf formatted file per recording. PCG input was recorded with the open-source LSL-compatible software Audio Capture (AC). All PCG recordings were done at a sampling frequency of 22050 Hz with a bit depth of 16 bits per sample. The files were named with respect to participant number (%p), session number (%s), task/block number (%b), run number (%r) and modality (%m) as

$$sub\text{-}\{\%p\}\_ses\text{-}\{\%s\}\_task\text{-}\{\%b\}\_run\text{-}\{\%r\}\_\{\%m\}.xdf \tag{3.1}$$

Since LSL has no native support for PCG as a modality, the EEG modality was used. Figure 3.1 shows AC and LSL running with tunable parameters. Both AC and LSL are publicly available software. Stable downloads can be found at GitHub repositories [9], [10].

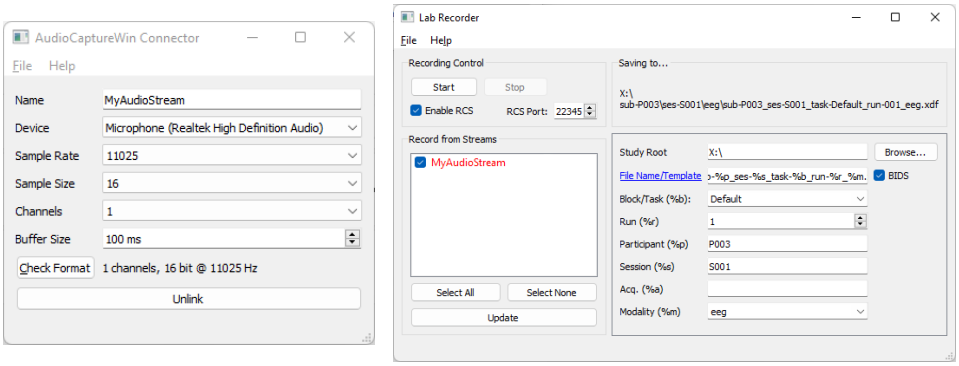


Figure 3.1: Lab Streaming Layer receiving stream "MyAudioStream" from AudioCapture. Mic directly connected to the computer's sound card [44].

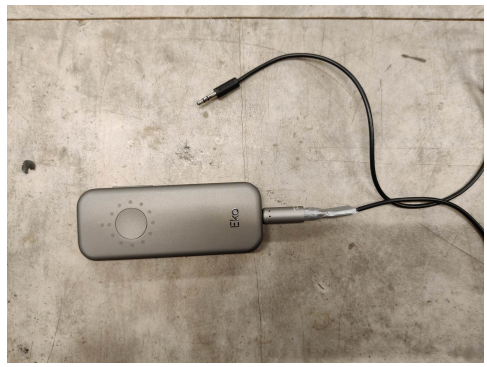


Figure 3.2: Eko Duo with connected aux cable [44].

An arithmetic test for inducing stress was created with Psychopy [37], a Python library that provides a graphical interface for research with markers that can be streamed to LSL. The code for the arithmetic test can be found in $arithmetic.py$ with related problems and answers in $arithmetic.xlsx$. The arithmetic equations and answers are also listed in appendix B. All problems were made to be answered as TRUE or FALSE with a single keyboard press to ensure minimal noise pollution for the PCG recording. The test displays the questions in random order and sends a marker to LSL each time a new problem is shown with the problem number and correct answer and each time a keypress is received with the value of the received keypress. Arithmetic test markers are synced with LSL and stored in the same .xdf file as the PCG and EEG recordings.

All recordings were conducted in the office of Professor Marta Molinas at the Norwegian University of Science and Technology (NTNU). Participants were instructed to take a seat by a computer monitor. As the Eko Duo did not possess any inherent mounting capabilities, an elastic strap secured the PCG stethoscope around the participant's torso during recording. This served to stabilize the device and apply pressure on the microphone, thereby enhancing signal quality. Potential auscultation points for PCG recording are depicted in Figure 2.2, with Erb's auscultation point being the location of choice for this study. Each participant was recorded twice, with each recording session consisting of two 5-minute intervals. The first recording was performed without the arithmetic test, thereby exclusively eliciting internal mental stress as a stressor. For the second recording, participants were positioned in close proximity to a computer and keyboard to perform the arithmetic Psychopy test. The arithmetic test was used to obtain data that accurately reflects both internal and external stress responses.

3.3 Dataset Reformatting

The .xdf file format outputted from LSL is suboptimal for load times, especially when not all data needs to be loaded at once. For this reason, a choice was made to restructure the dataset. In the downloadable version of the dataset [43], the data comes in binary files depending on the modality. PCG, EEG, and arithmetic test markers come in separate folders. The data structure is summarized below for each modality.

PCG: 83 full-quality 22050Hz recordings stored as (.npy) files with file names identifying participant number (p), session number (s), and run number (r) as

$$\text{key format} = P\{p\}\_S\{s\}\_\{r\} \tag{3.2}$$

EEG: 83 full-quality 250Hz recordings of 8-channel data (.xdf files). For more details, refer to [45].

Markers: One Python dictionary with keys in the same format as (3.2), stored as a .mat binary file. Each value entry of the dictionary is formatted as a list of tuples containing the question number (QNR), the time at which the question is displayed (TD), the time at which the respondent answered (TA), and if the answer is TRUE or FALSE (ANS).

$$\text{arithmetic tuple} = (\text{QNR, TD, TA, ANS}) \tag{3.3}$$

3.4 Dataset Analysis

This section aims to give an understanding of the collected mental health dataset. Only the multimodal data is covered. The analysis explores STAI stress score distributions over session number, run number, age, and gender. Specifically, the analysis investigates whether there are any trends or patterns in how stress scores vary across these factors. All STAI scores used are S-Anxiety scores, as these are most relevant when detecting stress from physiological measures. The analysis also covers a frequency analysis comparison between example low and high-rated stress recordings.

3.4.1 Labels

Figure 3.3 is a boxplot showing the distribution of all STAI stress scores over the session and run numbers.

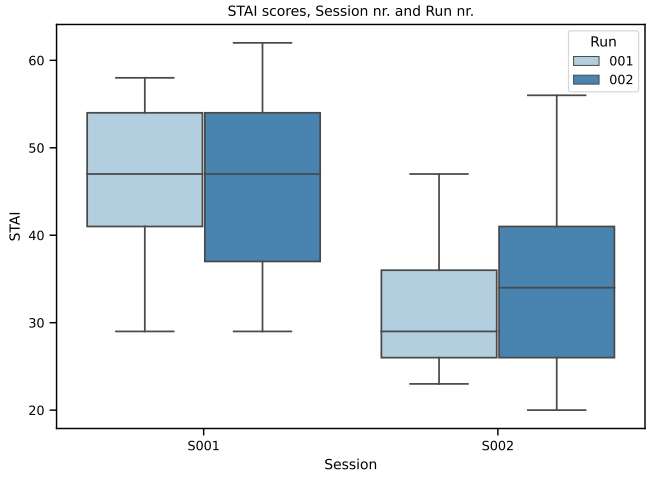


Figure 3.3: STAI score box plot displaying stress score distribution over the session and run numbers. S001 represents all recordings done in December 2022 (intended stressed state), and S002 represents all recordings done in January 2023 (intended low-stress/baseline state).

Stress scores range from 20 to 62, with an average of 38.05 and a median of 36, indicating good support on the low-stress part of the STAI scale and worse support for the high-stress part of the scale (with the maximal score attainable being 80). As one would expect, stress scores reported by students for S001 in the pre-examination- and project delivery phase are generally higher than those reported after Christmas vacation in S002. Furthermore, Run 001 recordings without external stimuli appears to better separate high- and low-stress states than Run 002 recordings done with the arithmetic test as external stimuli. The arithmetic test shows its intended purpose in the S002 recordings, where the median stress score of the Run 002 recordings is significantly higher than the median of the Run 001 recordings. For S001, the arithmetic test did not seem to induce any significant difference

in stress scores.

Figure 3.4 displays a boxplot comparing the stress scores of male and female participants over the two sessions. There are 45 recordings from males and 38 recordings from females in the dataset.

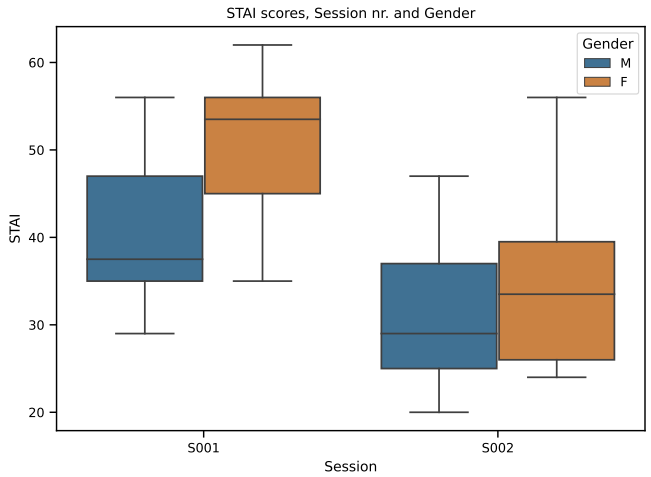


Figure 3.4: STAI score box plot displaying stress score distribution over session number and gender.

The plot shows a tendency for higher stress scores from females. This distinction is especially noticeable for S001 recordings. Stress scores were generally higher for both genders during S001 than for S002.

Next, we look at the age distribution of the dataset. Figure 3.5 displays the number of recordings per age.

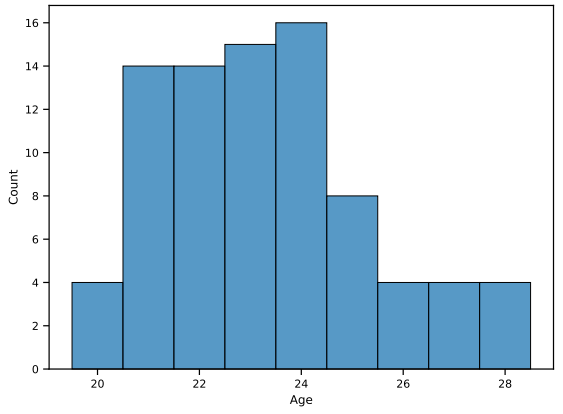


Figure 3.5: Age distribution over all recordings.

The mean age over the recordings is 23.31, and the median is 23. The plot shows that most recordings are from individuals between 21 and 25 years of age. Figure 3.6 shows a joint plot comparing age and STAI scores.

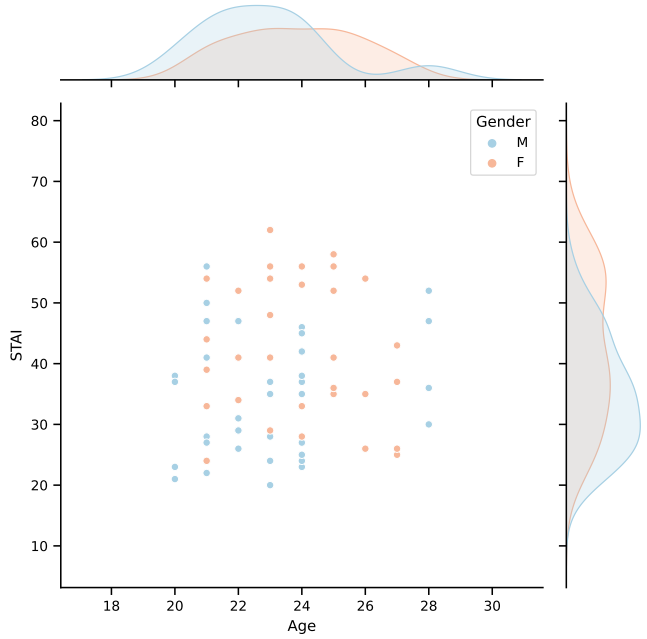


Figure 3.6: Joint plot comparing age to STAI score distribution for all recordings. Male recordings are marked in blue, and female recordings are marked in orange.

For lower ages up to 24, we see a good distribution of STAI scores for both male and female recordings. In the higher age range, there are fewer participants and a smaller spread in genders over age.

3.4.2 Frequency Analysis

Due to the nature of the multimodal measurement setup, the PCG recordings are polluted with noise at 50 Hz and 60 Hz and the harmonics of these noises (100 Hz, 120 Hz, 150 Hz, 180 Hz, ...). This noise pollution becomes apparent in the power spectrum of the data. Figure 3.7 shows the power spectrum of all data in the dataset concatenated. Since all recordings are concatenated, any environmental noise for each recording is minimized, and the overall systematic noise trend is more clearly indicated.

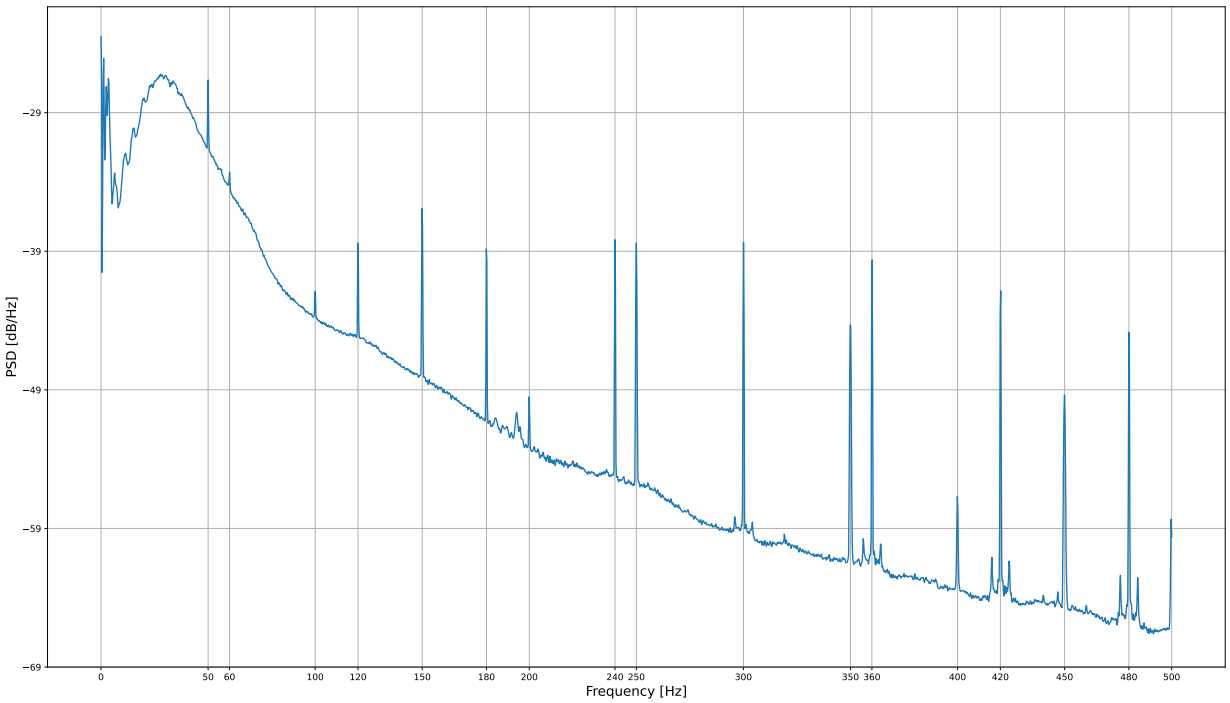


Figure 3.7: Power Spectral Density of all data in the dataset concatenated. Data is downsampled from 22050Hz to 1000Hz and normalized between [-1,1] before computing the PSD.

The noise spikes at 50Hz and harmonics are most likely due to electromagnetic interference from the power line, which is prevalent since the PCG sensor was connected with an aux cable to the computer. 60Hz peaks and harmonics come from an unknown origin. A plausible explanation is that it results from amplifiers in different computer components and the computer sound card. A more plausible explanation is that since the Eko Duo is made to capture both ECG and PCG signals, electrode lift from the ECG sensor, which often causes 50/60Hz spikes [46], is polluting the recording. That is, since the Eko Duo is connected to the computer through the bottom aux output, which is only intentionally meant to be used with the Eko earpiece, interference from the ECG-related circuitry may have affected the PCG output. The Eko Duo is a US-manufactured device where a power-line frequency of 60Hz is used. Therefore, an electrode lift from the ECG device might cause 60Hz and harmonics spikes in the aux output. Most of these noise peaks could have been avoided by not wiring the digital stethoscope to the computer and recording wirelessly through Eko's app interface. Still, since the dataset requirements were time-synchronized EEG, PCG, and arithmetic test markers, this was not possible without Lab Streaming Layer and a wired connection. Figure 3.8 shows a wireless recording through Eko's app interface.

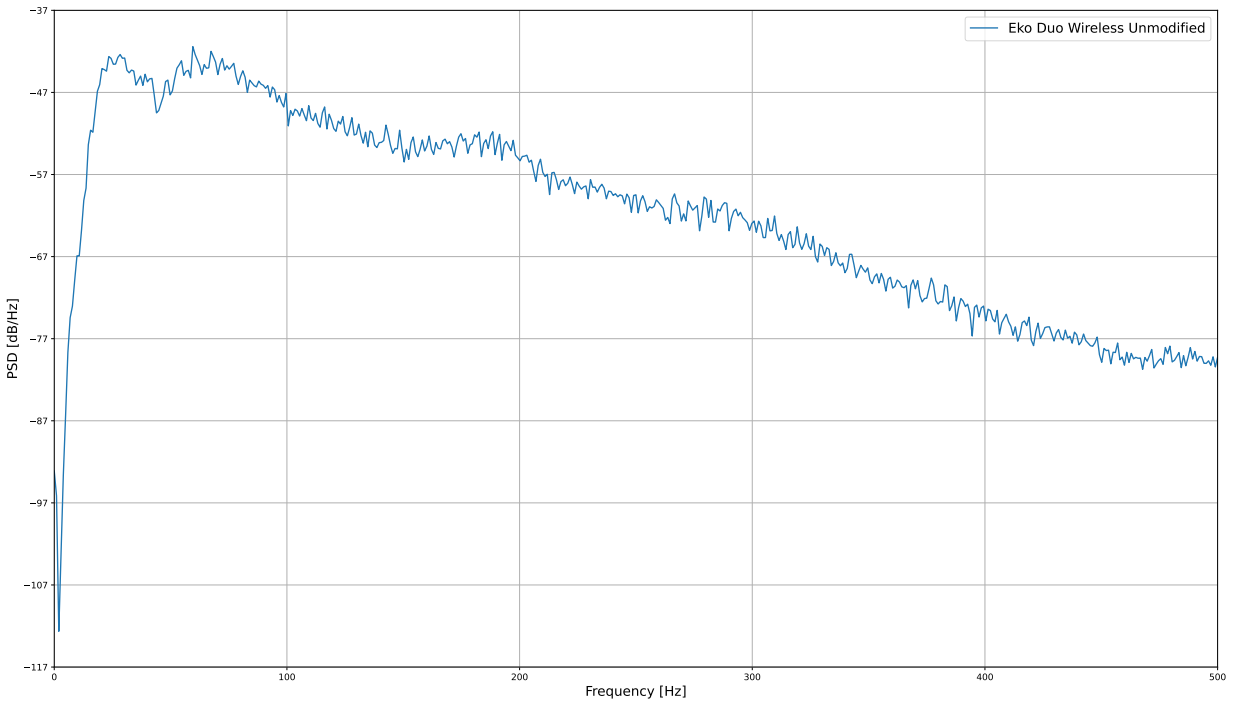


Figure 3.8: Wireless unmodified example recording through Eko's app interface. 120 seconds of 1000 Hz PCG data. The 50/60 Hz and harmonic peaks are non-existent.

As seen from the figure, the 50/60 Hz spikes are unnoticeable, confirming that the wired aux connection is the source of the systematic noise.

Comparing the mental health data to normal heart function recordings from the publicly available Physionet/CinC 2016 challenge dataset [33], previously mentioned assumptions are strengthened. Figure 3.9 compares two recordings from the Physionet dataset with two recordings from the collected mental stress dataset. The PSD is computed from an 8-second interval for all recordings displayed.

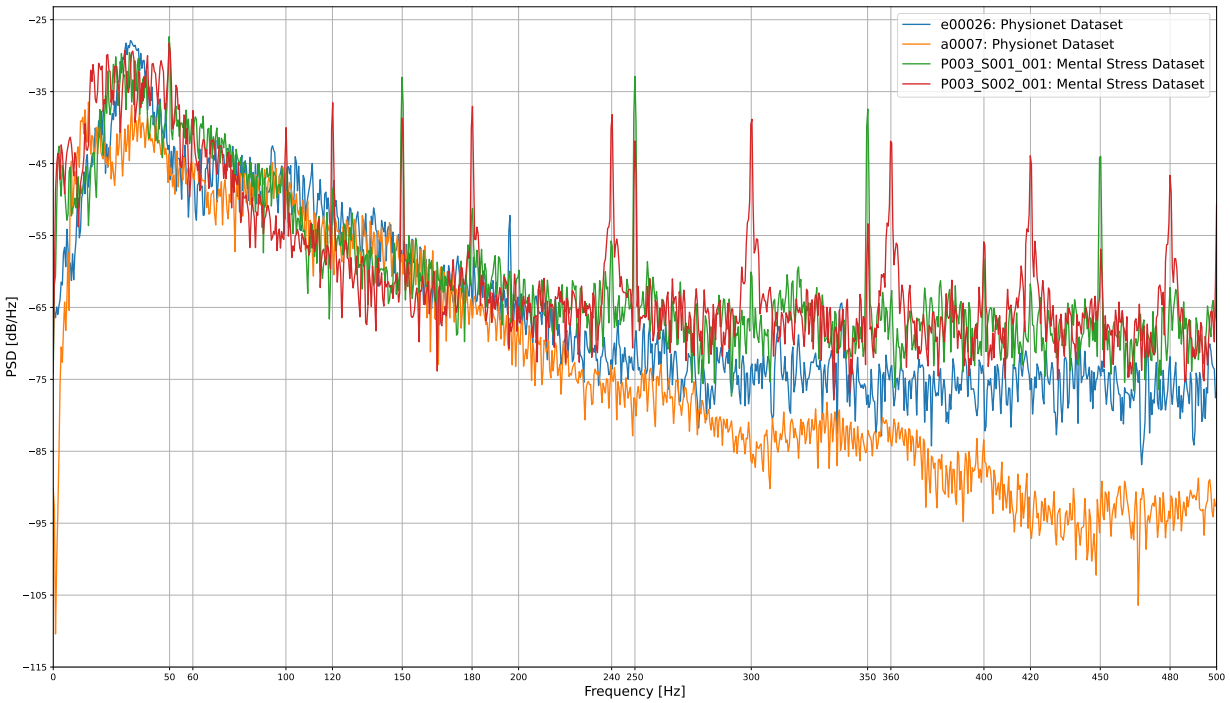


Figure 3.9: Power Spectral Density comparing normal heart function Physionet dataset recordings to Mental Stress dataset recordings. All recordings are downsampled to 1000Hz before computing the PSD. An 8-second window of each signal is used.

The 50/60Hz and harmonic peaks are not present in the Physionet recordings but are still present in the mental health recordings. The 'P003_S001_001' recording has spikes clear spikes at 50Hz and most harmonics and smaller spikes at 60 Hz and harmonics, while the 'P003_S002_001' recording spikes at 50/60Hz and harmonics. This strengthens the assumption that the ECG circuitry of the EKO duo is leaking noise into the PCG output at 50Hz and harmonics for some recordings or parts of recordings. Since the PSD is now computed over a shorter interval of 8 seconds per recording, we see the impacts of environmental noise as rapid fluctuations in the PSD curve. The environmental noise is dependent on a multitude of factors such as background noise during recording, how the PCG sensor is mounted, and other mechanical vibrations that enter the patient through epidermal tissue. Without environmental noise and power-line noise, we would have smoother PCG waveforms, thus, a smoother PSD curve.

To limit the impact the 50/60Hz frequencies and their harmonics induce on machine learning results, features created from frequency bands around these frequencies will be excluded. This method is explained in chapter 4.

By employing a binary stress label split distributing recordings based on STAI score, distinct differences between low- and high-stress classes in the power spectrum can be seen.

Figure 3.10 is a PSD plot comparing the high- and low-stress class differences over different frequencies using the STAI-35-20 split (this binary dataset split is detailed in chapter 4).

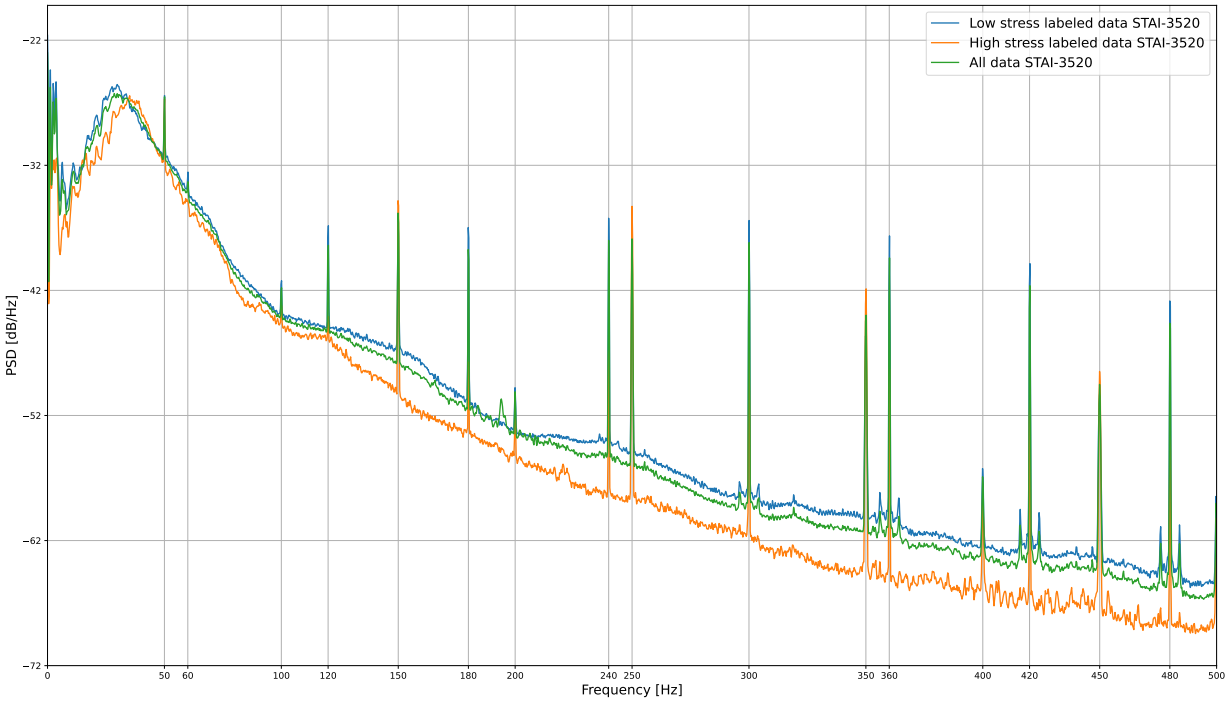


Figure 3.10: Power Spectral Density comparing concatenated data from low (blue) and high (orange) stress-labeled classes with the STAI-35-20 split. All data concatenated is also included in the plot (green).

As figure 3.10 shows, there are apparent differences in frequencies between the low- and high-stress classes when inspecting frequency content over large time intervals. From figure 3.9, the opposite is true; when averaging over shorter time intervals (8 seconds here), the frequency differences for different recordings are less visible, and a classifier might instead benefit from a high temporal resolution rather than high frequency-resolution.

4

PCG Machine Learning Methods and Classification Pipeline

This chapter aims to explain all methods, established and developed, used on the multimodal mental stress dataset. All augmentation and testing steps are accurately described to secure a reproducible method. All code used for implementation is referenced in the text and can be accessed through GitHub [43]. A conveniently formatted version of the PCG data ready for ML use can be found in the same repository.

4.1 Label Augmentation

To obtain a dataset that better discriminates between high and low-stress recordings, and because the STAI scores are susceptible to errors due to individuality biases as discussed in chapter 2, two dataset splits with binary stress labels are proposed:

1. **STAI-30-30**: Only recordings with STAI scores in the upper and lower 30 percentile is used. This removes recordings with STAI scores in the range $[31, 43]$, effectively reducing the number of recordings from 83 to 51. Recordings with STAI scores ≥ 44 will be labeled 'high stress', and recordings with STAI scores ≤ 30 will be labeled 'low stress'.
2. **STAI-35-20**: Recorings in the lower 35 and upper 20 percentile are used. This better discriminates the highest stress recordings and removes recordings with STAI scores in the range $[34, 47]$, which reduces the number of recordings from 83 to 50. Recordings with STAI scores ≥ 48 will be labeled 'high stress' and recordings with STAI scores ≤ 33 will be labeled 'low stress'.

Figure 4.1 displays the dataset before reduction, and figure 4.2 displays reduced STAI-30-30 and STAI-35-20 binary labels.

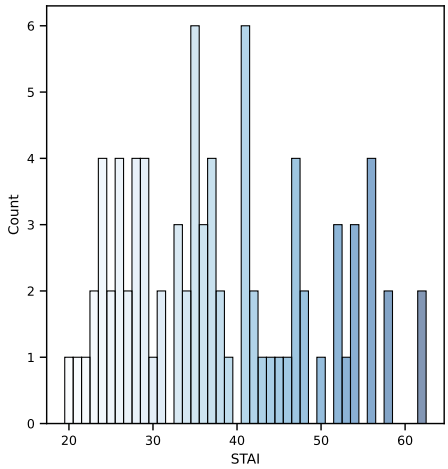


Figure 4.1: STAI score distribution pre-augmentation.

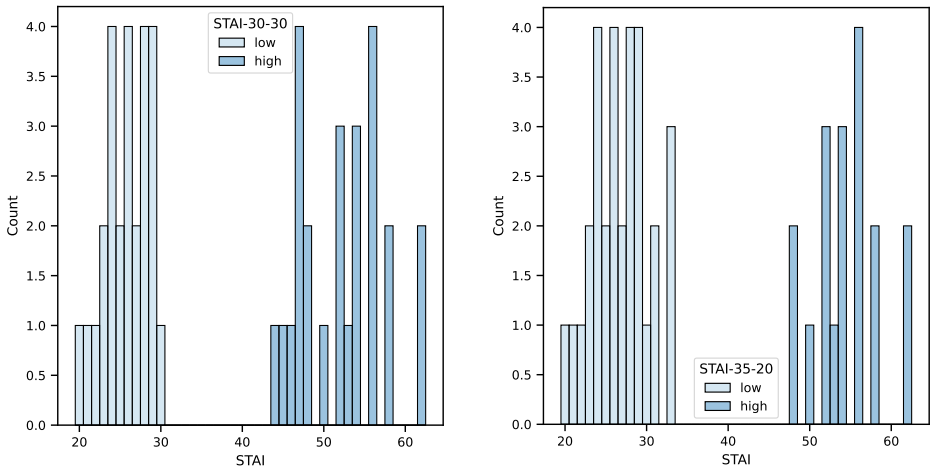


Figure 4.2: Dataset reduction from STAI labels to binary STAI labels. STAI-30-30 binary labels (left) and STAI-35-20 binary labels (right).

A clear distinction is that STAI-35-20 has a more uniform distribution on the STAI scale than STAI-30-30. Figure 4.3 shows the distribution of 'low' and 'high' binary labels for the two dataset splits. STAI-30-30 gives a more even support of the two classes, while STAI-35-20 has a lower support for 'high' stress levels.

Each dataset split is processed identically for the rest of the processing steps.

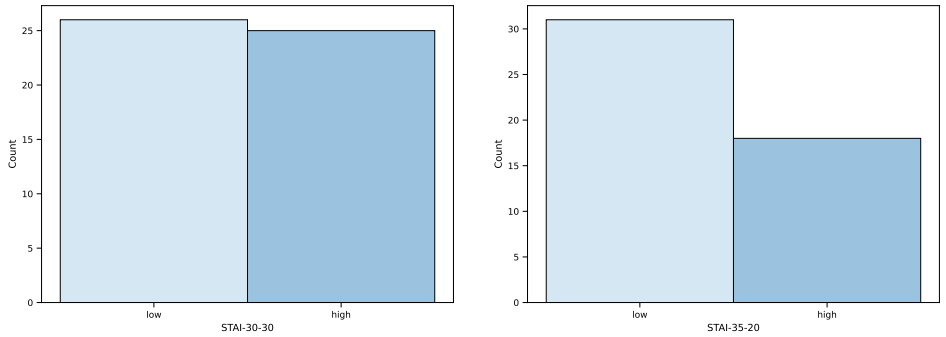


Figure 4.3: Binary label distribution of reduced datasets. STAI scores pre-augmentation (left), STAI-30-30 binary labels (left), and STAI-35-20 binary labels (right).

4.2 Data Augmentation

The next augmentation step involves processing the raw PCG data to prepare it for the scattering transform.

4.2.1 Resampling and Normalization

Each recording is resampled from 11025Hz to 1000Hz, to make computations later down in the pipeline less expensive while keeping analysis information of important spectral properties of the PCG signal (Nyquist frequency is now 500Hz). Subsequently, each signal $x[i]$ for $i \in [0, N]$, where N is the signal sample length, is normalized in amplitude to the range $[-1, 1]$ as:

$$\widetilde{x}[i] = \frac{x[i]}{\max_{i \in [0,N]}(|x[i]|)}. \tag{4.1}$$

The next augmentation step involves preparing the signals for scattering by performing asynchronous segmentation with segmentation length L. The segmentation length needs to be dividable by the invariance scale of the scattering network $T = 2^J$. It is therefore chosen as:

$$L = \lfloor \frac{sf_s}{T} \rfloor T, \tag{4.2}$$

where s denotes the approximate segmentation length in seconds, and f_s is the sampling frequency of the signal, i.e 1000Hz. Doing asynchronous segmentation with segmentation length L results in $\lfloor N/L \rfloor$ signal segments for a signal of length N. The end of the signal (not dividable to an integer by N) is discarded. The labels are formatted to fit the new shape of the data. For implementation details, refer to *load_and_preprocessing_functions.py*

4.2.2 Normalized Scattering Transform

The segmented data is a 2D array with shape (n_{seg}, L), where n_{seg} is dependent on the length of each of the individual signals before segmentation. The scattering network

performs the scattering transform on each of the signal segments of length L with non-overlapping time windows of size $T = 2^J$, resulting in a three-dimensional tensor output Sx of shape $(n_{seg}, d, L/T)$ containing the zeroth, first, and second order scattering coefficients for each signal segment. d is dependent on the octave resolution Q, and invariance scale T, and max order of the network. The Python library Kymatio [3] is used to compute the scattering coefficients. Following this, the scattering coefficients are renormalized with (2.20), scaling the first-order coefficients by the average absolute amplitude of each signal segment over L, and scaling the second-order coefficients by their respectable first-order coefficient. This ensures that local amplitude modulations are kept under the scale L. This results in the normalized scattering coefficients $\widetilde{S}x$. For implementation detail, refer to *cv_and_classification_functions*.

Before being sent to the classification pipeline, the normalized scattering coefficients are reshaped once more to shape $(n_{seg}L/T, d)$. This means that the classifier is trained on spectral information derived from the PCG signal over distances time averaged in time over the invariance scale of the network T, with local amplitude modulation context of temporal length L.

4.2.3 Noise Related Feature Exclusion

As discussed in chapter 3, there are electronic noise peaks consistent in most recordings at very specific frequencies, specifically the 50Hz and 60Hz frequencies and their harmonics. Since these peaks vary a lot in intensity from recording to recording (especially the 60Hz frequency and harmonics) and have little to do with the actual PCG data we want from the patient, a method is proposed to exclude any features constructed from or around these frequencies.

A scattering coefficient feature is constructed by one or two wavelets (first and second-order coefficients, respectively) with given center frequencies and bandwidths stored in the first and second-order filter bank (with center frequency sets $\widetilde{\xi}_{\lambda_1}, \lambda_1 \in \Lambda_1, \widetilde{\xi}_{\lambda_2}, \lambda_2 \in \Lambda_2$ respectively). The center frequencies and bandwidths used are dependent on the (J, Q) parameters of the scattering network. To transform the normalized center frequencies $(\tilde{\xi})$ and bandwidths $(\tilde{\sigma})$ of the wavelet to relate to the frequencies in the signal, they are multiplied with the signal sample frequency (f_s) as:

$$\xi = \tilde{\xi} f_s, \quad \sigma = \tilde{\sigma} f_s, \tag{4.3}$$

denoting the proportionally scaled wavelet center frequencies and bandwidths. From here, any scattering coefficient constructed from wavelets with bandwidth ranges $[\xi - \sigma/2, \xi + \sigma/2]$ in an area near any of the problematic noise frequencies (f_i) will be excluded as:

$$\text{Exclusion Criteria} = \xi - \frac{\sigma}{2} - f_\epsilon < f_i < \xi + \frac{\sigma}{2} + f_\epsilon, \tag{4.4}$$

where f_ϵ is a threshold around the issue frequency set to 7Hz from testing. This implies that scattering features constructed from wavelets covering bandwidths in the ranges $[f_i -$

7, $f_i + 7$]Hz will be excluded. The frequency bands where feature exclusion happens with $f_\epsilon = 7$Hz is depicted in figure 4.4.

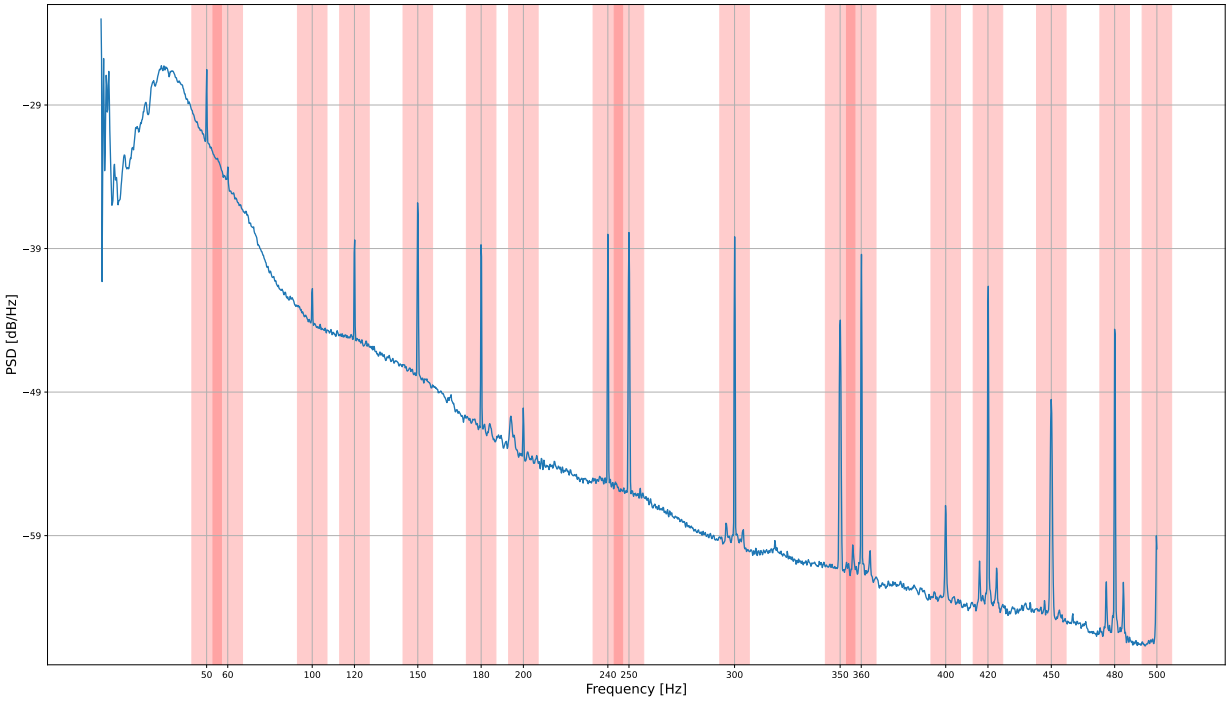


Figure 4.4: Frequency bands in the PCG data where scattering features are excluded for the machine learning pipeline ($f_\epsilon = 7$Hz). The graph (blue) shows the Power Spectral Density of all data in the Mental Stress Dataset concatenated.

This technique is applied instead of the traditional notch filtering because the noise peaks change in intensity for different recordings and at different times in the recordings. Thus a constant quality factor for a static filter that fits all recordings is not attainable. An adaptive filtering procedure could have been performed if we had the error signals or denoised versions of the signals. Still, since all data in the dataset is contaminated by systematic noise, this is not an option. PCG signals have important characteristics in overlapping frequency bands with some issue frequencies. Thus feature exclusion will likely reduce the model's performance since some of the features removed lie in important bands. However, this feature reduction step is necessary for the model to use the physiological response seen in the PCG signal of the patient as a stress detector instead of the difference in noise peak intensities between recordings. In a perfect scenario, with clean PCG data, all scattering features could be kept. The percentage of features removed with this method is a result of the wavelet filter distribution in the filterbanks, decided by J and Q. For implementation details, refer to *cv_and_classification_functions.py*.

4.3 Classification Pipeline

The classification pipeline entails dataset splitting, model parameter optimization by cross-validation, and final model evaluation with chosen scattering network parameters and hyperparameter combinations. The structure of the classification pipeline is summarized in figure 4.5.

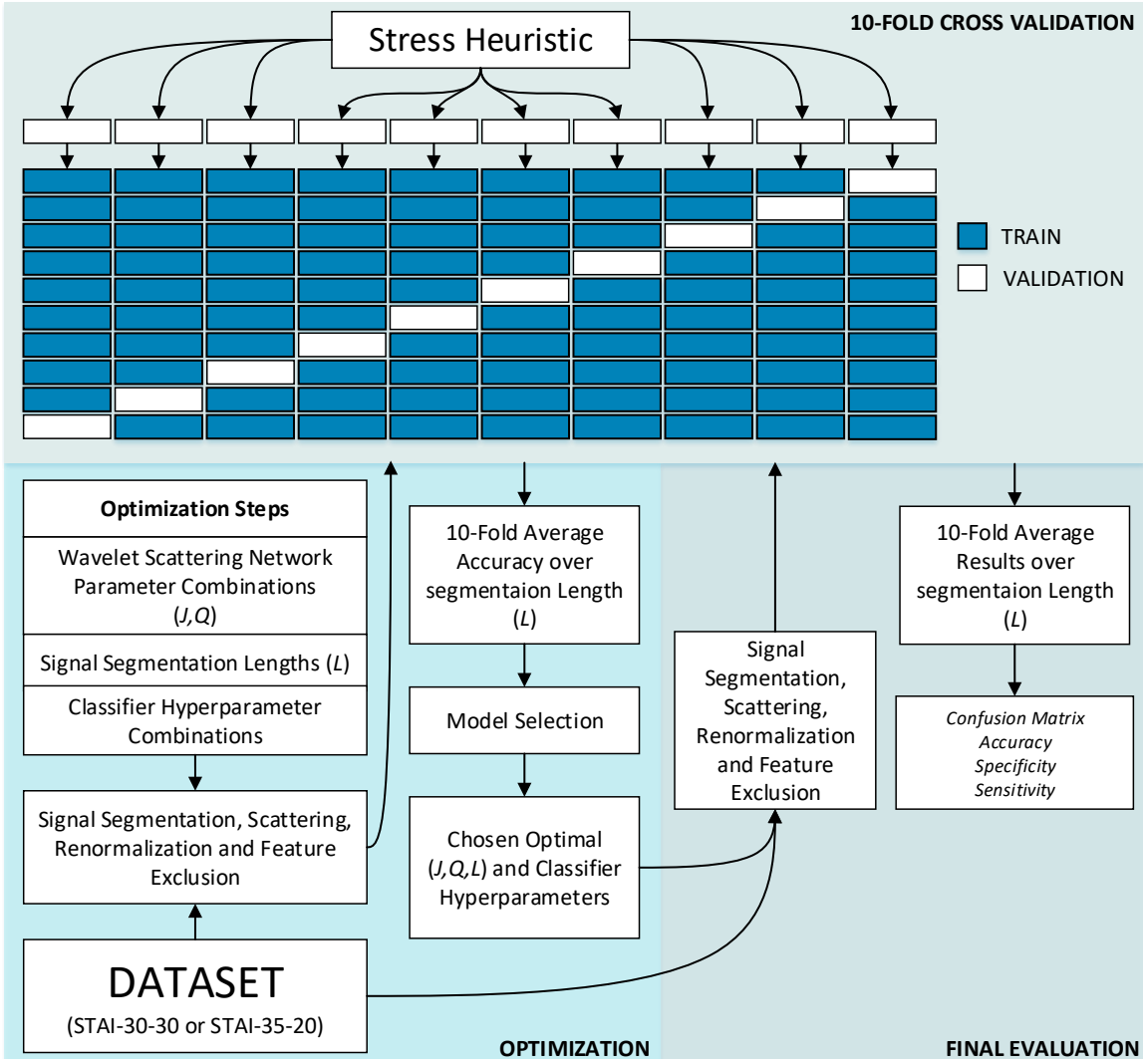


Figure 4.5: Classification pipeline used for STAI-30-30 and STAI-35-20. The figure shows the parameter optimization steps, the 10-fold cross-validation procedure, and the final model evaluation with results.

4.3.1 The Stress Heuristic

A stress heuristic function was implemented to ensure good splits in cross-validation. The stress heuristic aims to split data into similar subject sets stratified with respect to stress labels. The algorithm makes sure none of the sets have overlapping subjects. This is done for each subject, measuring if there is

1. An overweight of high stress recordings $\implies$ labeled 1
2. An overweight of low stress recordings $\implies$ labeled -1
3. A close to equal distribution $\implies$ labeled 0

and by using these labels, structuring sets of subjects with a close to equal proportion of high- and low-stress recordings to obtain a stress-stratified split with no overlapping subjects. It is very important that the splits have no overlapping subjects since if this was not the case, the classifier would train to recognize people, a much simpler task than training to recognize stress. Stratifying stress is important to ensure good support for both stress classes. The stress heuristic is used in the 10-fold cross-validation split.

4.3.2 Feature Standardization

To make the renormalized scattering features work optimally with classifiers, they are standardized with sklearn's StandardScaler [36]. For input feature train and validation matrices X_{tr} and X_{val} on the format number of samples and number of features, i.e $[n_{s,tr}, n_{f,tr}]$, $[n_{s,val}, n_{f,val}]$, each feature matrix for row i and column j is standardized as:

$$X_{tr,scaled} = \frac{X_{tr} - \bar{X}_{tr}}{\sigma(X_{tr})}, \tag{4.5}$$

where

$$\bar{X}_{tr} = \frac{1}{n_{f,tr}} \sum_{j=0}^{n_{f,tr}-1} X_{tr,j}, \quad \sigma(X_{tr}) = \sqrt{\frac{1}{n_{f,tr}} \sum_{j=0}^{n_{f.tr}-1} (X_{tr,j} - \bar{X}_{tr})^2}, \tag{4.6}$$

and

$$X_{val,scaled} = \frac{X_{val} - \bar{X}_{tr}}{\sigma(X_{tr})}. \tag{4.7}$$

This procedure gives the feature vectors zero mean and standard deviation of 1, similar to the standard distribution.

4.3.3 Cross-Validation

The cross-validation algorithm implemented aims to optimize the ML pipeline by finding the best model parameter combinations. This is done both by optimizing PCG segmentation lengths (L) and by doing a grid search over relevant scattering network parameter pairs (J, Q) for each of the segmentation lengths. For each (L, J, Q) tuple, a 10-fold cross-validation procedure is performed. The stress heuristic creates ten non-overlapping

subject sets where nine sets are used as train data and one as validation data. The train and validation sets are standardized according to equations (4.5, 4.6, 4.7). Training is done ten times so that each set is used once as validation data. The classifier is for each fold trained on standardized scattering features from signals of length T. It predicts the standardized validation data with this fitted classifier for each time segment T for all L/T parts of the original segmented signal of length L. Then the average class prediction of the L/T predictions is saved as the prediction for the input of length L. The average validation accuracy for all signal segments over the ten folds is saved as the (L, J, Q) tuples score. Using 10-fold cross-validation, we get data from 1–2 subjects per data-fold. This means the validation set has less room for stress level stratification. However, the train set will still be sufficiently stratified to prevent class imbalance issues since the classifier is trained nine folds at a time. The average of the ten evaluation results will thus indicate the model's generalizability on unseen data for the given (L, J, Q) tuple.

Two classifiers are tested in the cross-validation procedure. KNN [36], and XGBoost ensemble classifier [13]. During (L, J, Q) optimization, only the XGBoost classifier is used. Default XGBoost parameters are used except for $tree\_method$ which is set to 'hist' to speed up computations. For specifics on the default hyperparameter values, please refer to Appendix C. Normalized scattering coefficients, as described above, are used as feature vectors. Optimally, (L, J, Q) values should be tuned individually for each classifier since they can benefit from different feature characteristics. However, because testing a wide range of parameter permutations in a 10-fold procedure takes a lot of time, only the XGBoost classifier is used for this optimization step.

After (L, J, Q) is optimized, hyperparameter combinations for the two classifiers are optimized with normalized scattering features constructed with the obtained (L, J, Q) tuple values. This optimization step uses the Python package Hyperopt [8], which employs a meta-modeling approach for hyperparameter optimization over a set parameter space. Read more on specific hyperparameters tuned and chosen ranges below. The loss function of the Hyperopt package is set to judge performance as the negative accuracy over the ten cross-validation folds. Hyperopt finds the hyperparameter set within the set ranges that minimizes this loss function. Refer to *classifier_optimization.ipynb* for code implementation details.

Segmentation Length Optimization

A given segmentation length (L) imposes the temporal scale at which the amplitude modulation context of the normalized scattering coefficients is kept. Doing the scattering at scales larger than the network invariance scale T is also necessary to avoid border effects in the wavelet convolutions of the scattering transform. Tested segmentation lengths in seconds include $[10, 20, 30]$s which corresponds to $L \approx [10000, 20000, 30000]$ with the chosen $f_s = 1000Hz$. These values for L are only accurate if L is divisible by T, and are otherwise decided by (4.2).

Scattering Network Parameter Grid Search

The scattering network parameters (J, Q) are tested over a large range of parameter values with a grid search. J sets the maximum log scale of the scattering transform, i.e., the invariance scale of the network $T = 2^J$. The decision of J imposes the number of samples the classifier is trained with. A larger J implies a smaller set of training samples since the time averaged over to create one spectral scattering sample is longer. The Q parameter sets the number of wavelets per octave, i.e., the spectral resolution. Thus, a larger Q will result in a higher dimensional feature vector for each sample fed into the classifier. Values tested include $J \in [3, 13]$, $Q_1 = \{6, 9, 12, 15, 17, 20, 22, 26\}$, $Q_2 = 1$. These values for Q_1 were chosen in a way that provides the least amount of features lost from the feature exclusion procedure described above. Higher wavelet octave resolutions than $Q_1 = 26$ are not tested because computation times increase algorithmically for higher Q. Lower values for $J = 3$ are not included since this will result in the loss of second-order scattering coefficients for the higher Q values, thereby reducing performance.

KNN Hyperparameter Space

Hyperparameter space values include:

- Number of neigbors: $n\_neigbors \in [1\,..\,100]$
- Minkowski distance type: $p \in [1\,..\,2]$

The n_jobs hyperparameter of the KNN is set to -1 to utilize all CPU threads and thus speed up computations. All non above mentioned hyperparameters are set to default values. max_evals for Hyperopt is set to 50. Thus we are testing 50 hyperparameter permutations over the hyperparameter space.

XGBoost Hyperparameter Space

Hyperparameter space includes:

- Number of trees: $n\_estimators \in [20\,..\,300]$
- Learning rate: $eta \in [0, 1]$
- Max tree depth: $max\_depth \in [3\,..\,20]$
- Minimum loss reduction required to make a further partition on a leaf node of the tree: $gamma \in [0, 12]$
- L1 weight regularization term: $reg\_alpha \in [0\,..\,110]$
- L2 weight regularization term: $reg\_lambda \in [0, 2]$
- Subsample ratio of columns when constructing each tree: $colsample\_bytree \in [0.5\,..\,1]$
- Minimum sum of instance weight (hessian) needed in a child: $min\_child\_weight \in [0\,..\,6]$

The $tree\_method$ parameter is set to 'hist' during hyperparameter optimization for much faster computations. All XGBoost hyperparameters not mentioned above are set to default values. max_evals for Hyperopt is set to 100. Thus we are testing 100 hyperparameter permutations over the hyperparameter space.

4.3.4 Final Model Evaluation

For the final model evaluation, optimal scattering network parameters and classifier hyperparameters decided in cross-validation are evaluated. KNN, and XGBoost final performances are evaluated similarly to cross-validation with a 10-fold approach. A hold-out test set is not utilized mainly because of the uncertainty in the STAI labels and partly due to the small dataset size. The classifiers are trained on features of size d where the features are constructed from a time segment with the length of the invariance scale of the network T. The classification performance is rated over the original segmentation length L. Performance evaluations such as confusion matrix, sensitivity, and specificity are saved.

Since the XGBoost classifier is a gradient boosting algorithm, information on which specific features are most important for the classifier's mental stress predictions can be found from the trained boosted trees. Each feature value of the feature vector is related to the underlying wavelet filter and the center frequency used to extract it. Thus, relationships between important features and important frequencies for discerning mental stress states can be found this way. Importances of all features are summed for each training fold and divided by 10 (the number of folds) to retrieve the average feature importance for each binary dataset split. This feature-frequency importance relationship is saved during the final model evaluation.

5

Cross-validation Results and Final Evaluation

5.1 Segmentation Length and Scattering Network Optimization

XGBoost classifier 10-fold cross-validation grid search results for asynchronous segmentation lengths (L), scattering network log invariance scale (J), and octave frequency resolution (Q) are detailed in the subsections below.

5.1.1 STAI-30-30 (L, J, Q) Optimization

STAI-30-30 XGBoost classifier 10-fold cross validation (L, J, Q) grid search results are shown in figure 5.1.

A consistent pattern is seen for higher accuracies from lower network log invariance scales, J, which yields better time localization and more data samples ($L/2^J$ per segment of length L) for the classifier training. The accuracies for $J = 3, Q > 15$ are very bad because second-order scattering coefficients are not possible to calculate for such short sample size filters with this high frequency resolution, and thus only first order features are used. Mid-range scales J work the worst for all lengths L with accuracies averaging in the low 70s.

The (L, J, Q) parameter set chosen for further classifier hyperparameter optimization (marked in red in figure 5.1) is $(L, J, Q) = (30000, 4, 12)$. This parameter combination is chosen due to having the highest 10-fold average accuracy score over the parameters tested. A slight increase in accuracy is seen for high frequency resolutions Q and high network log invariance scales J. Marked in pink, $(L, J, Q) = (24576, 13, 20)$ performs best among the highest Js. This parameter combination might be more viable for a larger dataset.

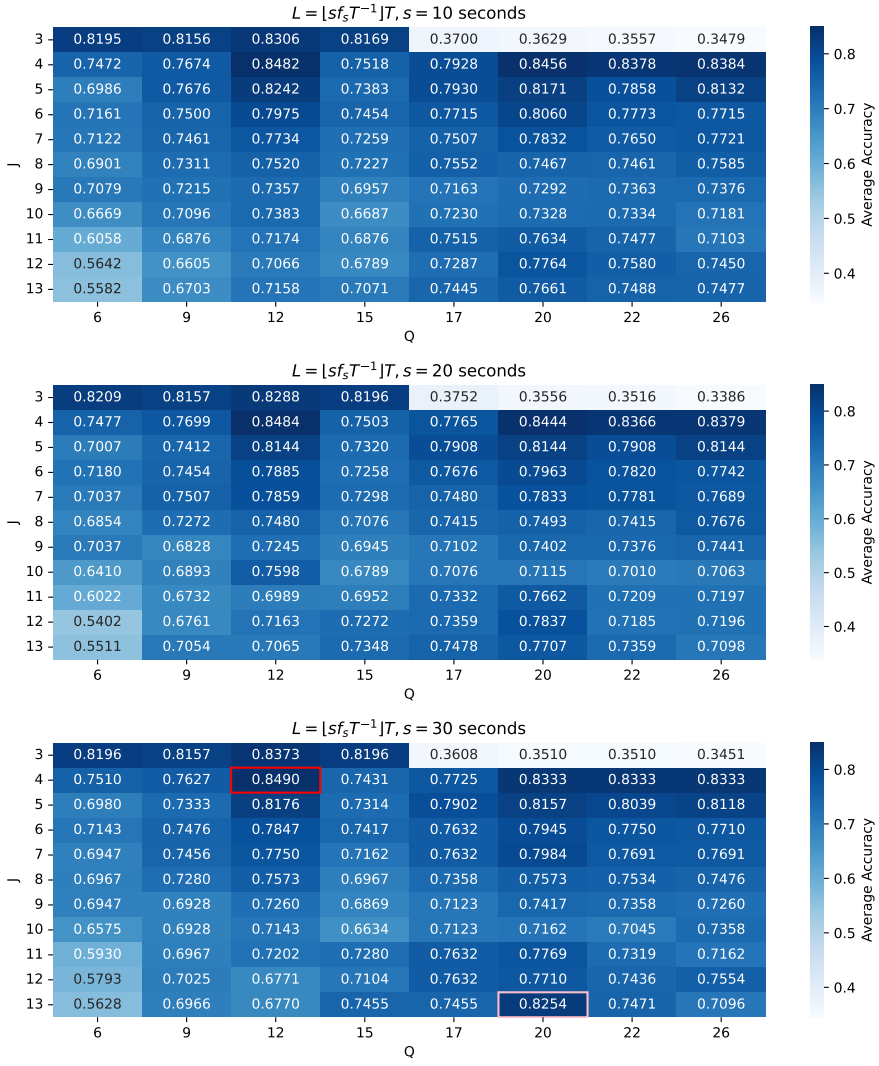


Figure 5.1: (L, J, Q) optimization STAI-30-30 for XGBoost Classifier with default hyperparameters except tree_method='hist'. Average accuracy over 10-fold cross-validation. Chosen parameter combination for further optimization is marked in red. A hopeful combination for larger datasets is marked in pink.

The chosen scattering parameters, $(L, J, Q) = (30000, 4, 12)$, yields the scattering respresentaion dispalyed in figure 5.2.

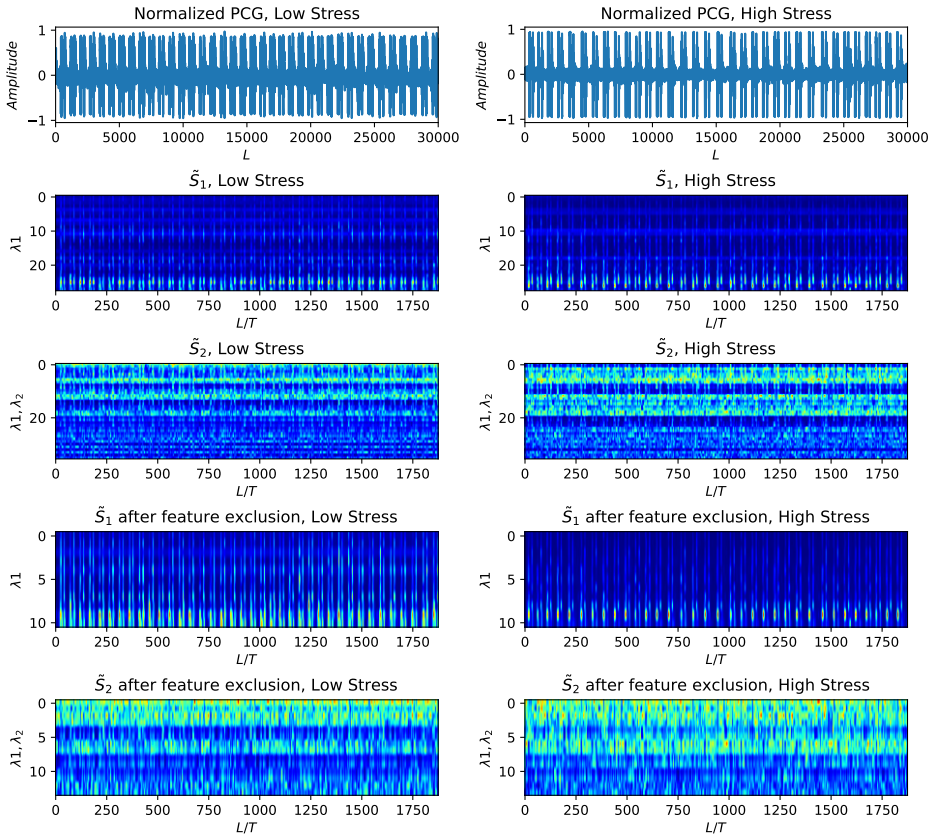


Figure 5.2: Scattering representation of $(L, J, Q) = (30000, 4, 12)$. The original signal and Normalized scattering coefficients before and after feature exclusion are shown for 10-second segments from a low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress recording in STAI-35-20. Feature exclusion effectively removes the noise bands of the powerline frequencies visible as light blue horizontal lines in $\widetilde{S}_1$ before exclusion. Every L/T column spectral vector in $\widetilde{S}_1$ and $\widetilde{S}_2$ after feature exclusion is used as one classification data sample.

5.1.2 STAI-35-20 (L,J,Q) Optimization

STAI-35-20 XGBoost classifier 10-fold cross validation (L, J, Q) grid search results are shown in figure 5.3.

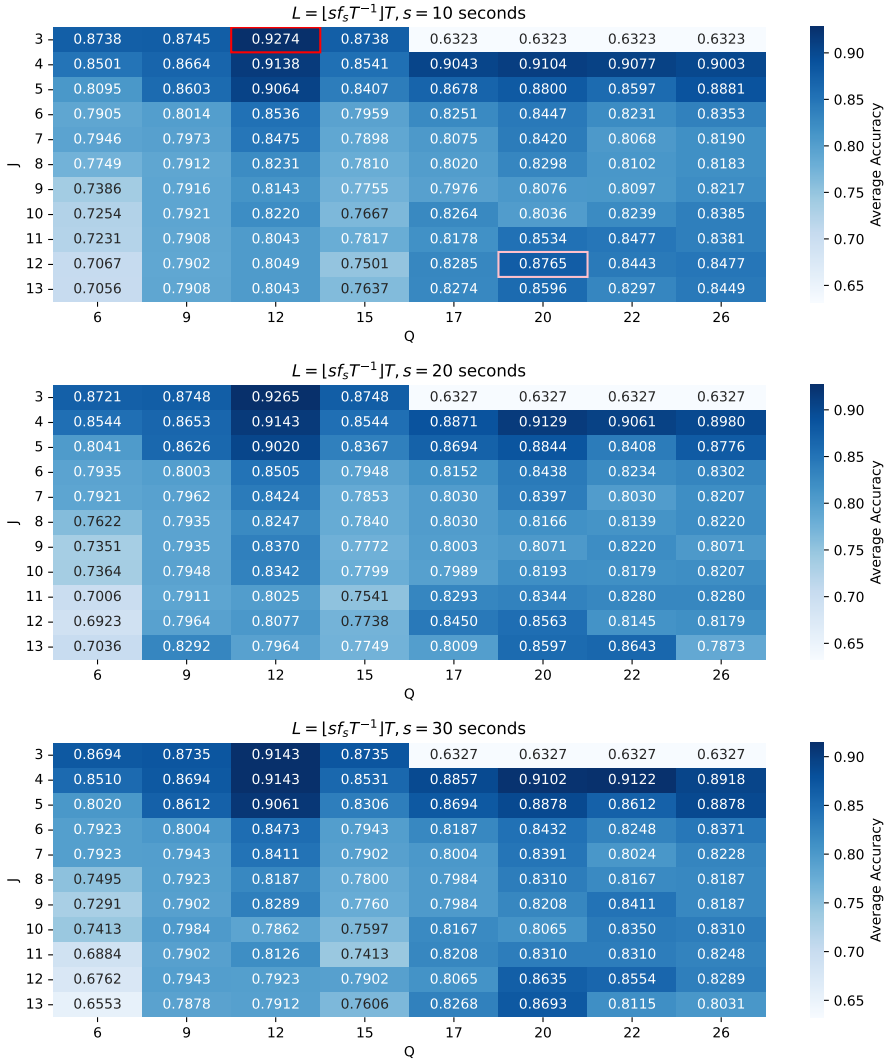


Figure 5.3: (L, J, Q) optimization STAI-35-20 for XGBoost Classifier with default hyperparameters except tree_method='hist'. Average accuracy over 10-fold cross-validation. Chosen parameter combination for further optimization is marked in red. A hopeful combination for larger datasets is marked in pink.

The same pattern is seen as for STAI-30-30 with higher accuracies from lower network log invariance scales, J. Mid-range scales J work the worst for all lengths L with accuracies averaging in the low 80s.

The (L, J, Q) parameter set chosen for further classifier hyperparameter optimization (marked in red in figure 5.3) is $(L, J, Q) = (10000, 3, 12)$. These parameters are chosen since

they yield the highest 10-fold average accuracy score over the parameters tested. A slight increase in accuracy is seen for high frequency resolutions Q and high network log invariance scales J. Marked in pink, $(L, J, Q) = (28672, 12, 20)$ performs best among the highest Js. This might be a more viable parameter combination for a larger dataset.

The scattering parameters chosen, $(L, J, Q) = (10000, 3, 12)$, yield the scattering representation displayed in figure 5.4.

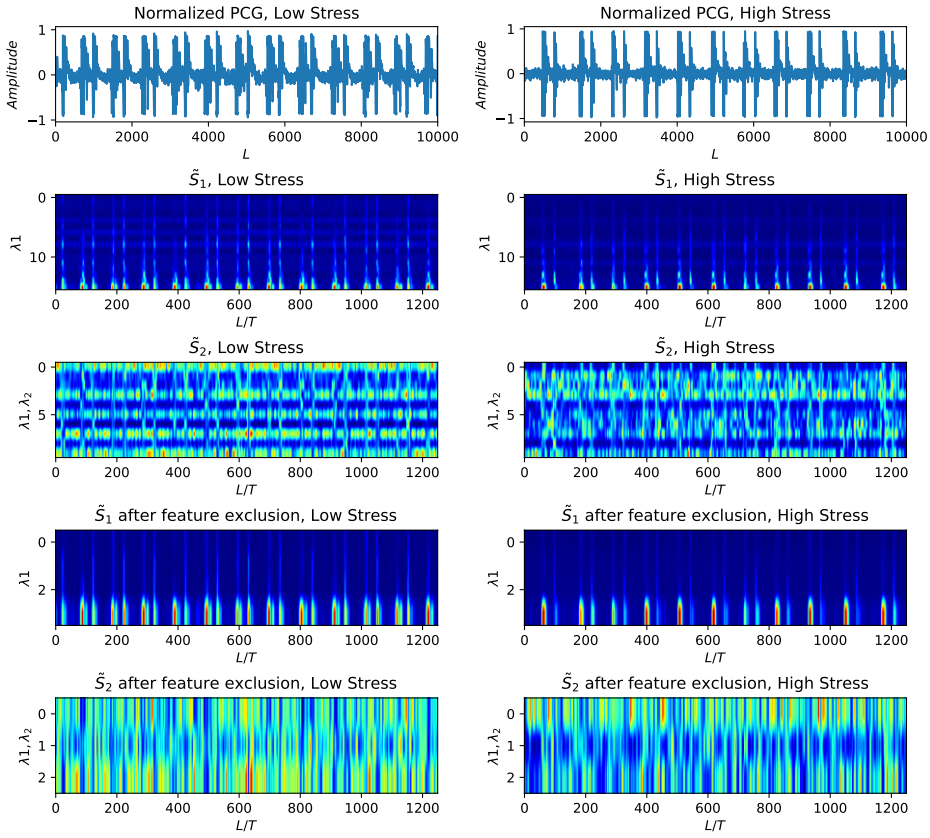


Figure 5.4: Scattering representation of $(L, J, Q) = (10000, 3, 12)$. The original signal and Normalized scattering coefficients before and after feature exclusion are shown for 10-second segments from a low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress recording in STAI-35-20. Feature exclusion effectively removes the noise bands of the powerline frequencies visible as light blue horizontal lines in $\widetilde{S}_1$ before exclusion. Every L/T column spectral vector in $\widetilde{S}_1$ and $\widetilde{S}_2$ after feature exclusion is used as one classification data sample.

5.1.3 (L, J, Q) Optimization Summary

The two dataset split, segmentation, and scattering network parameter combinations chosen for classifier hyperparameter optimization are summarized in table 5.1.

Table 5.1: L,J,Q optimization summary for STAI-30-30 and STAI-35-20.

Parameter	STAI-30-30	STAI-35-20
L	30000	10000
J	4	3
Q	12	12

5.2 Hyperparameter Optimization

The hyperparameter optimization is done using Hyperopt with both dataset binary splits for both the KNN- and the XGBoost classifier. The search for the best hyperparameters are presented as heat contour plots of two hyperparameter ranges at a time, where cold parts indicate high accuracy / low loss (loss = negative 10-fold average accuracy). For specifics on hyperparameters used and loss result for each evaluation, please refer to tables 7.2, 7.3, 7.4, and 7.5 in appendix D. The final hyperparameter values from this tuning procedure are summarized in tables 5.2 and 5.3.

5.2.1 STAI-30-30, KNN Classifier

Figure 5.5 shows a heat contour plot comparing parameter ranges of the minkowski distance type p and the number of neighbors $n\_neighbors$.

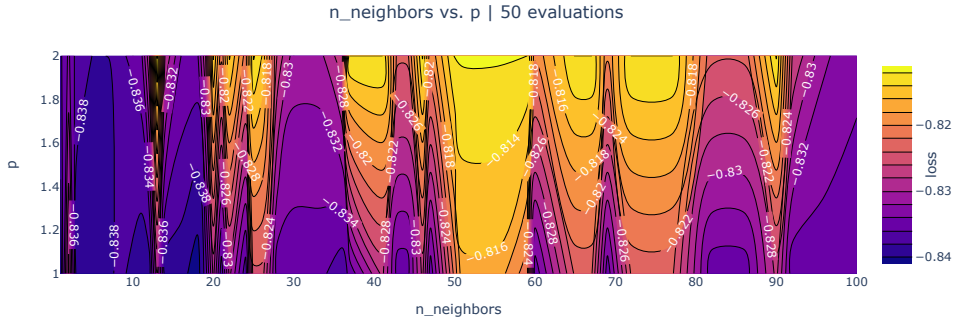


Figure 5.5: STAI-30-30 Hyperopt parameter search illustrated over 50 model evaluations (trials). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $p = 1$ and $n\_neigbors = 18$.

From the figure it is observed that $p = 1$ generally yield lower loss values and is thus preferred over $p = 2$ for STAI-30-30. The best hyperparameters over the 50 evaluations are found to be $p = 1$ and $n\_neighbors = 18$.

5.2.2 STAI-30-30, XGBoost Classifier

Figure 5.6 shows a heat contour plot comparing parameter ranges of the XGBoost learning rate (eta) and the number of CARTs $(n\_estimators)$ for STAI-30-30. In the same way figure 5.7 compares the regularization parameter $gamma$ and the maximal tree depth (max_depth), figure 5.8 compares the L1 regularization term reg_alpha and L2 regularization term reg_lambda, and figure 5.9 compares the subsample ratio of columns when constructing each tree $(colsample\_bytree)$ and the minimum sum of instance weight needed in a child (min_child_weight).

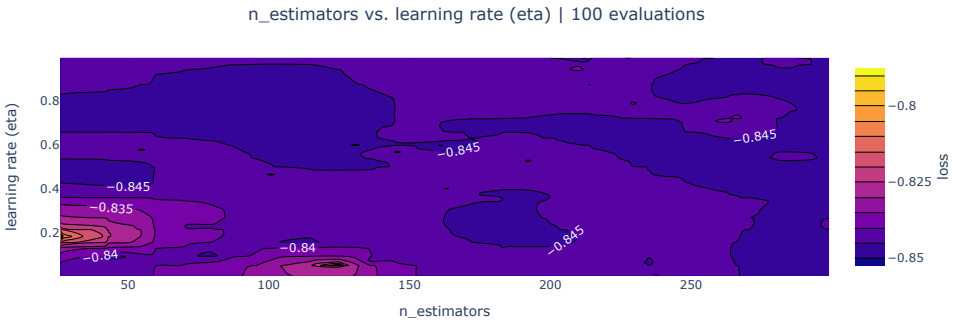


Figure 5.6: STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing number of trees $(n\_estimators)$ and learning rate (eta). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $n\_estimators = 131$ and $eta = 0.598764$.

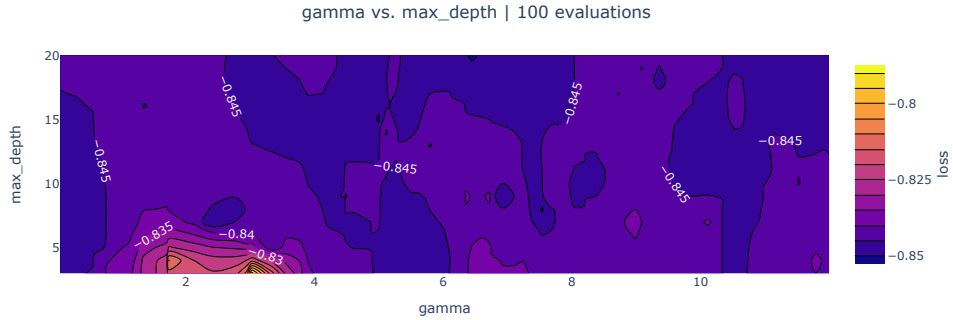


Figure 5.7: STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing regularization parameter $gamma$ and maximal tree depth (max_depth). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $gamma = 4.999442$ and $max\_depth = 15$.

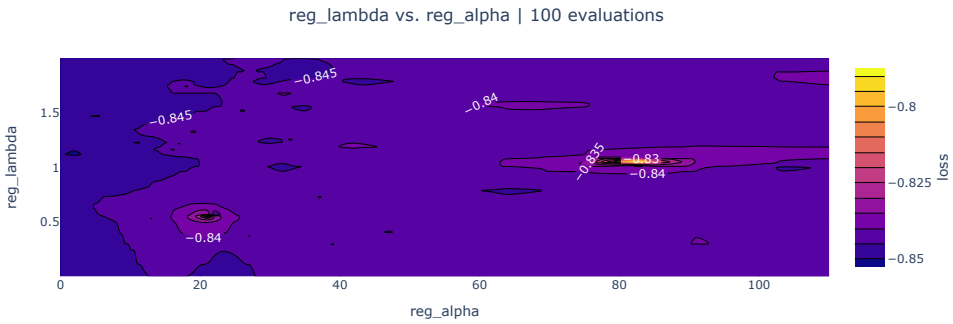


Figure 5.8: STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing L1 regularization term reg_alpha and L2 regularization term reg_lambda. The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $reg\_alpha = 5$ and $reg\_lambda = 1.467703$.

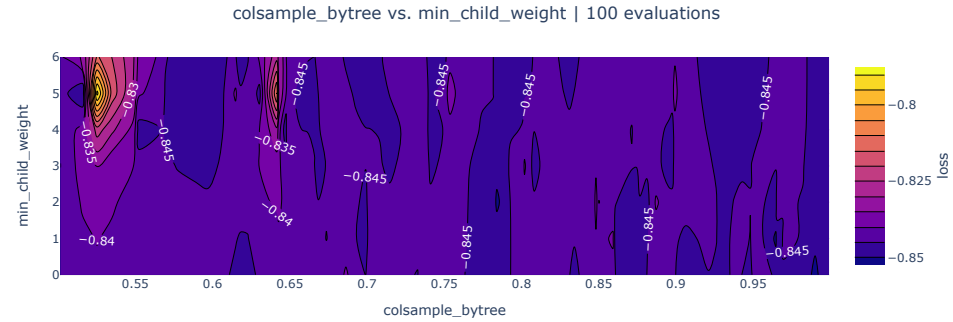


Figure 5.9: STAI-30-30 Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing the subsample ratio of columns when constructing each tree ($colsample\_bytree$) and the minimum sum of instance weight (hessian) needed in a child (min_child_weight). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $colsample\_bytree = 0.990046$ and $min\_child\_weight = 0$.

From the figures, the hyperparameter values found to be optimal for STAI-30-30 XGBoost over 100 evaluations are $eta = 0.598764$, $n\_estimators = 131$, $gamma = 4.999442$, $max\_depth = 15$, $reg\_alpha = 5$, $reg\_lambda = 1.467703$, $colsample\_bytree = 0.990046$, and $min\_child\_weight = 0$.

5.2.3 STAI-35-20, KNN Classifier

Figure 5.10 shows a heat contour plot comparing parameter ranges of the minkowski distance type p and the number of neighbors $n\_neighbors$ for STAI-35-20.

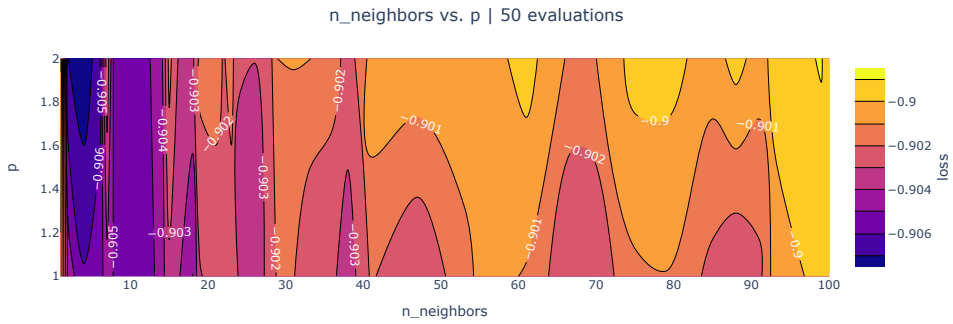


Figure 5.10: Hyperopt parameter search illustrated over 50 model evaluations (trials). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $p = 2$ and $n\_neigbors = 4$.

From the figure it is observed that a low number of neighbors yield the best KNN performance for STAI-35-20. The best hyperparameters over the 50 evaluations are found to be $p = 2$ and $n\_neighbors = 4$.

5.2.4 STAI-35-20, XGBoost Classifier

Figure 5.11 shows a heat contour plot comparing parameter ranges of the XGBoost learning rate (eta) and the number of CARTs $(n\_estimators)$ for STAI-35-20. In the same way figure 5.12 compares the regularization parameter $gamma$ and the maximal tree depth (max_depth), figure 5.13 compares the L1 regularization term reg_alpha and L2 regularization term reg_lambda, and figure 5.14 compares the subsample ratio of columns when constructing each tree $(colsample\_bytree)$ and the minimum sum of instance weight needed in a child (min_child_weight).

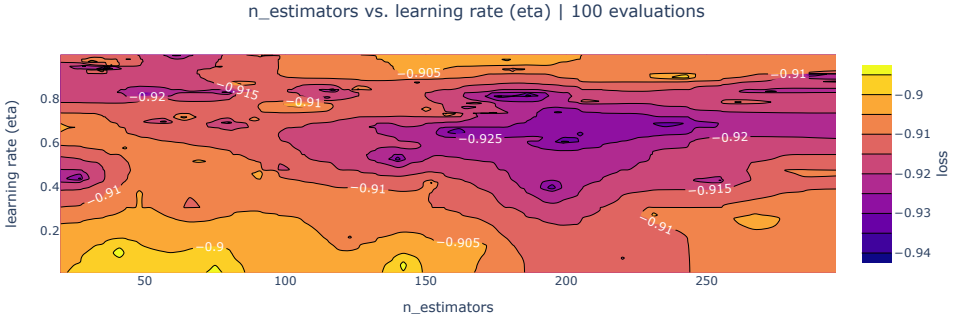


Figure 5.11: Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing number of trees $(n\_estimators)$ and learning rate (eta). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $n\_estimators = 81$ and $eta = 0.829672$.

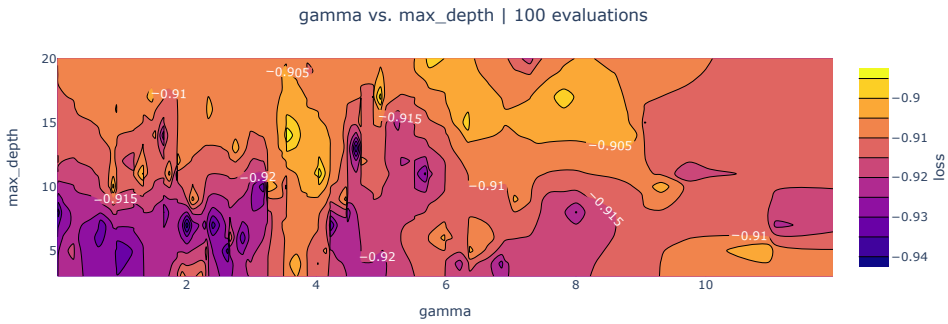


Figure 5.12: Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing regularization parameter $gamma$ and maximal tree depth (max_depth). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $gamma = 2.005672$ and $max\_depth = 7$.

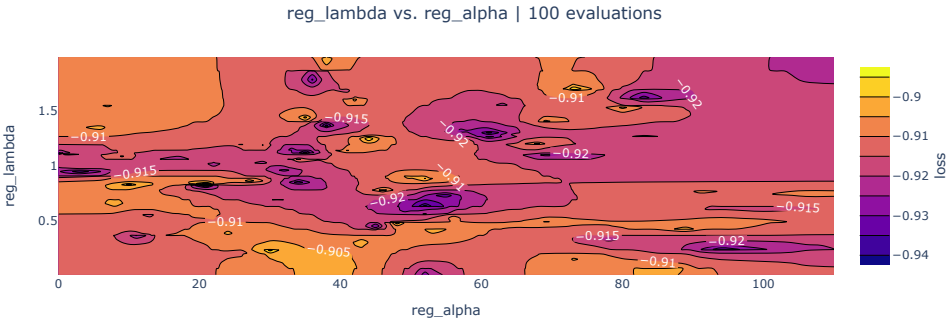


Figure 5.13: Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing L1 regularization term reg_alpha and L2 regularization term reg_lambda. The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $reg\_alpha = 52$ and $reg\_lambda = 0.631727$.

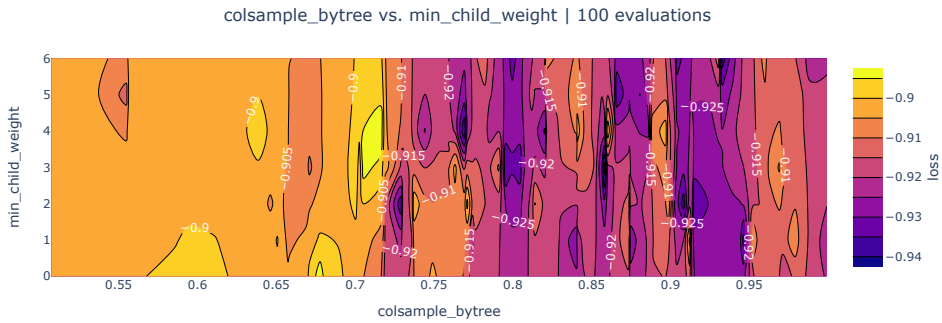


Figure 5.14: Hyperopt parameter search illustrated over 100 model evaluations (trials) comparing the subsample ratio of columns when constructing each tree ($colsample\_bytree$) and the minimum sum of instance weight (hessian) needed in a child (min_child_weight). The loss marks negative average 10-fold accuracy and is to be minimized. The best results are seen for $colsample\_bytree = 0.858025$ and $min\_child\_weight = 3$.

From the figures, the hyperparameter values that are found to be optimal for STAI-35-20 XGBoost over 100 evaluations are $eta = 0.829672$, $n\_estimators = 81$, $gamma = 2.005672$, $max\_depth = 7$, $reg\_alpha = 52$, $reg\_lambda = 0.631727$, $colsample\_bytree = 0.858025$, and $min\_child\_weight = 3$.

5.2.5 Hyperparameter Optimization Summary

The final hyperparameter values chosen for the KNN classifier and the XGBoost classifier are summarized in table 5.2 and 5.3 respectively.

Table 5.2: KNN classifier hyperparameter selections for each binary dataset split.

Hyperparameter	STAI-30-30	STAI-35-20
p	1	2
$n\_neigbors$	18	4

Table 5.3: XGBoost classifier hyperparameter selections for each binary dataset split.

Hyperparameter	STAI-30-30	STAI-35-20
eta	0.598764	0.829672
$n\_estimators$	131	81
$gamma$	4.999442	2.005672
max_depth	15	7
reg_alpha	5	52
reg_lambda	1.467703	0.631727
$colsample\_bytree$	0.990046	0.858025
min_child_weight	0	3

5.3 Final Model Evaluations

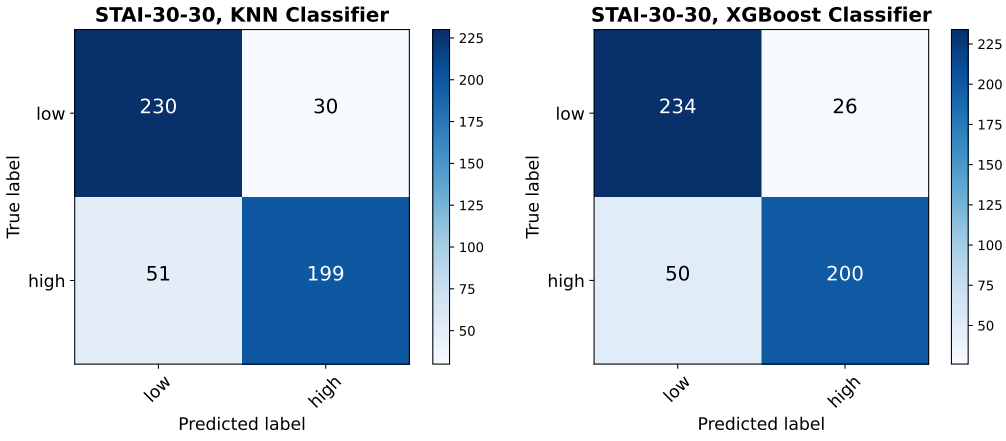
The segmentation lengths, scattering network parameters and classifier hyperparameter values found in previous steps are evaluated with performance metrics. Table 5.4 shows the sensitivity, specificity, AUC, and accuracy for the four models from predictions of all validation sets in the 10-fold cross validation procedure.

Table 5.4: Final 10-fold cross validation model performances. Performance measures obtained from concatenated predictions on all 10 fold's validation sets.

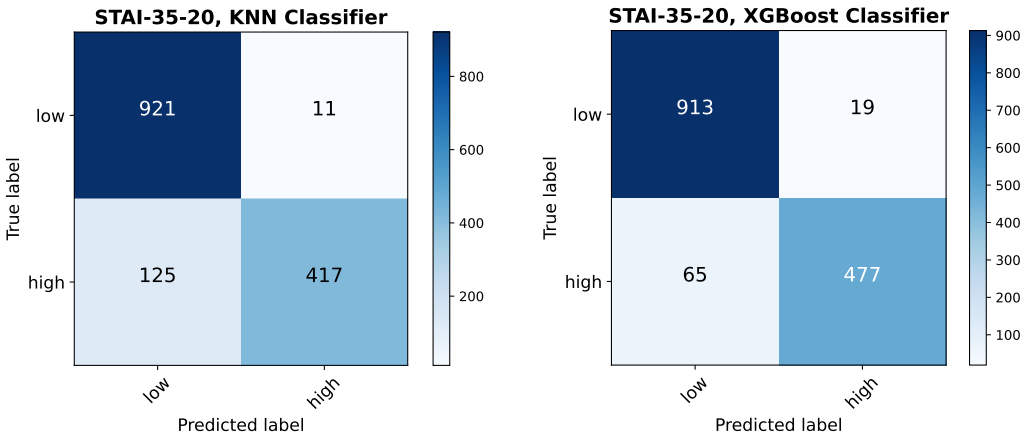
Classifier/Split	(L, J, Q)	Hyperparameters	Sensitivity	Specificity	AUC	Accuracy
KNN / STAI-30-30	(30000, 4, 12)	Table 5.2	0.8846	0.7960	0.8403	0.8412
XGB / STAI-30-30	(30000, 4, 12)	Table 5.3	0.9000	0.8000	0.8500	0.8510
KNN / STAI-35-20	(10000, 3, 12)	Table 5.2	0.9881	0.7694	0.8788	0.9078
XGB / STAI-35-20	(10000, 3, 12)	Table 5.3	0.9796	0.8801	0.9298	0.9430

There is a clear performance distinction between the predictions on STAI-30-30 and STAI-35-20. Both classifiers perform better on the STAI-35-20 binary split despite the uneven support of low-stress and high-stress recordings. This hints that a more even STAI-score distribution in the high- and low-stress sets used in STAI-35-20 is more critical for model performance than balanced class support, which is the case for STAI-30-30. The best Sensitivity is seen for KNN / STAI-35-20. However, XGB / STAI-35-20 shows the overall best performance with the best specificity, AUC, and accuracy among the final models. The XGB classifier also performs slightly superior to KNN on the STAI-30-30 split, scoring higher on all performance measures.

Confusion matrices for the four model predictions over the 10 folds are shown in figure 5.15.



(a) **(b)**

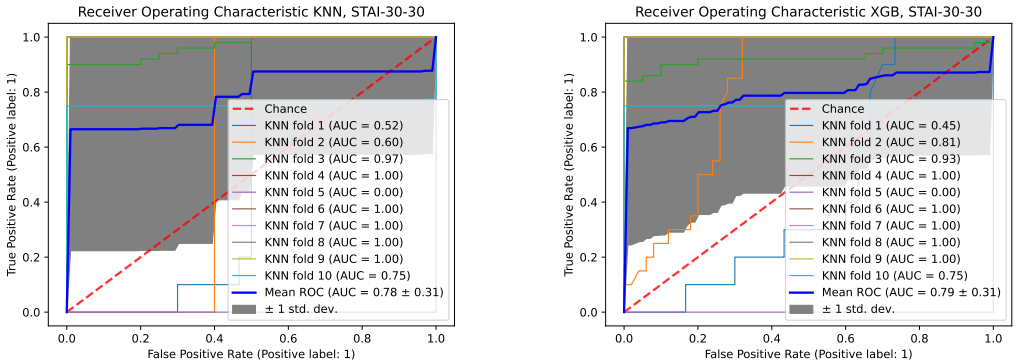


(c) **(d)**

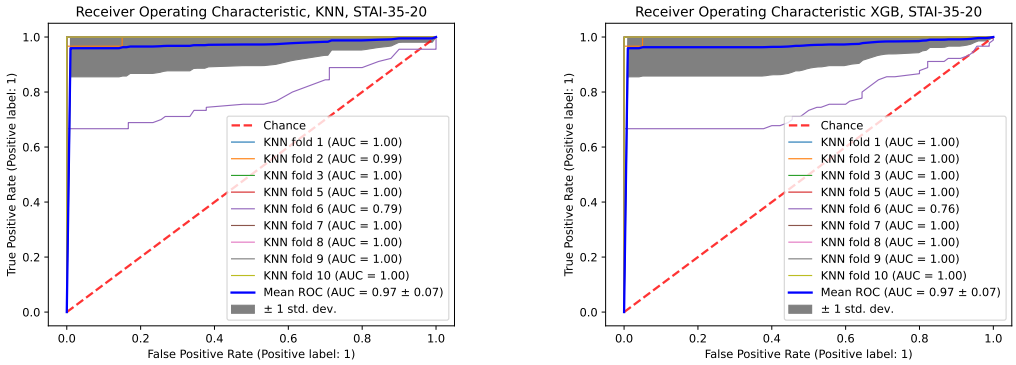
Figure 5.15: Final evaluation confusion matrices for the tested binary labels and classifiers. KNN, STAI-30-30 (a). XGBoost, STAI-30-30 (b). KNN, STAI-35-20 (c). XGBoost, STAI-35-20 (d).

For STAI-30-30, there are fewer segmented recordings to predict due to the higher chosen segmentation length L. The XGBoost classifier has more accurate predictions than the KNN classifier, with better sensitivity and specificity. For STAI-35-20, the KNN classifier has fewer FN predictions than the XGB classifier, yielding higher sensitivity. However, the XGB classifier outperforms KNN in every other way, better handling the class imbalance of STAI-35-20.

Providing insights into how some data folds drastically reduce performance, ROC AUC results per fold are displayed in figure 5.16. Fold 4 in STAI-35-20 contains no high-stress labeled samples and is thus excluded from the results since it would not make sense to refer to TPR or FPR in this case.



(a) **(b)**



(c) **(d)**

Figure 5.16: ROC AUC results for the tested binary labels and classifiers. KNN, STAI-30-30 (a). XGBoost, STAI-30-30 (b). KNN, STAI-35-20 (c). XGBoost, STAI-35-20 (d). For STAI-35-20, fold 4 is removed since the validation fold contains no high-stress samples and would therefore be irrelevant for ROC AUC analysis.

From figure 5.16, the mean AUC score over the folds does not precisely match the AUC scores listed in table 5.4. This is because each fold has some deviation with regards to the number of segmented recording samples it contains and thus the mean AUC score slightly differs from the AUC score obtained from the one calculated from all concatenated predictions over the ten folds in table 5.4.

For STAI-30-30 in figure 5.16a and 5.16b, we see folds 1, 2 and 5 obtain bad AUC scores, meaning that the train data of these folds generalize poorly to the validation data. Fold 5 gets an AUC score of 0, indicating that 0% of the high-stress recordings were classified correctly for this fold. The other eight folds generalize well. The XGBoost classifier performs slightly better on fold 2 than the KNN classifier and thus achieves a better performance in total on STAI-30-30.

For STAI-35-20 in figure 5.16c and 5.16d, most folds generalize well. Fold 6 has subpar performance indicating that this validation set is harder to generalize the rest of the data to. KNN and XGBoost achieve a similar mean AUC score. However, the XGBoost performs better in concatenated recordings' AUC score (see table 5.4).

5.3.1 Feature Importance

The scattering coefficient features have one or two underlying wavelets that extract the coefficient from the original PCG signal. For first-order coefficients, only one wavelet is used with center frequency $\xi_{\lambda_{1,j}}$, and for second-order coefficients, two wavelets are used with center frequencies $(\xi_{\lambda_{1,j}}, \xi_{\lambda_{2,j}})$. Figures 5.17 and 5.18 shows the feature/frequency importance relationship derived from the XGBoost classifier for STAI-30-30 and STAI-35-20, respectively.

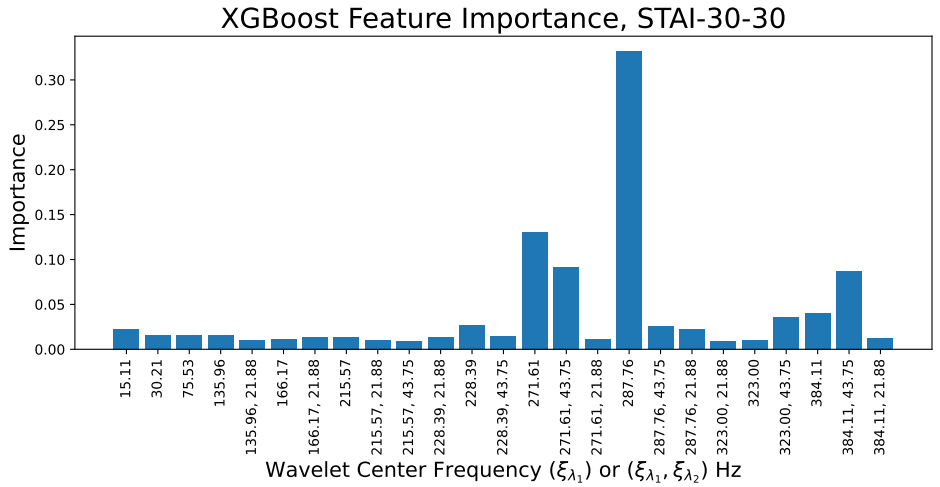


Figure 5.17: Barplot showing feature/frequency importance relationship for STAI-30-30. The scattering coefficients' importance is shown with underlying first and second-order wavelet center frequencies are shown in Hz as (ξ_{λ_1}) for first-order coefficient features and $(\xi_{\lambda_1}, \xi_{\lambda_2})$ for second-order coefficient features. The most important frequencies for discerning mental stress states in STAI-30-30 are 287.76 Hz, 271.61 Hz, (271.61,43.75) Hz, and (384.11,43.75) Hz.

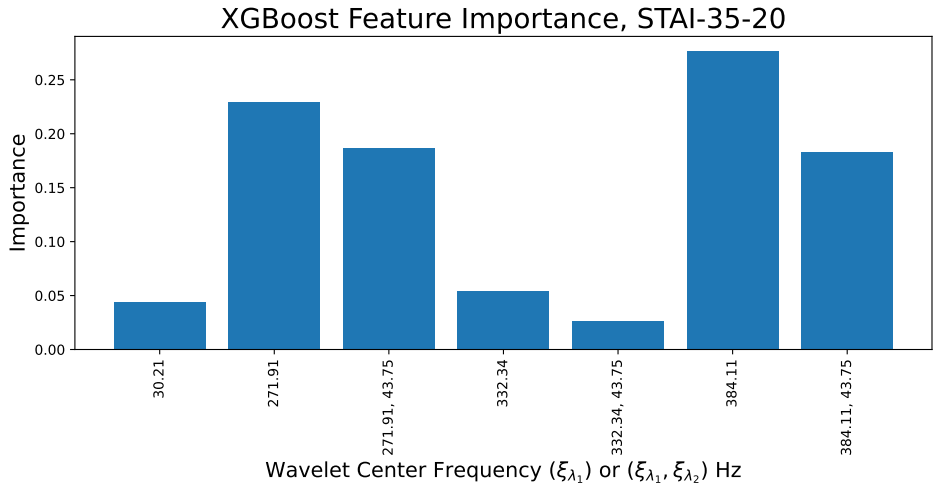


Figure 5.18: Barplot showing feature/frequency importance relationship for STAI-35-20. The scattering coefficients' importance is shown with underlying first and second-order wavelet center frequencies are shown in Hz as (ξ_{λ_1}) for first-order coefficient features and $(\xi_{\lambda_1}, \xi_{\lambda_2})$ for second-order coefficient features. The most important frequencies for discerning mental stress states in STAI-35-20 are 384.11Hz, 271.91Hz, (271.91,43.75)Hz, and (384.11,43.75)Hz.

For both binary dataset-splits, higher wavelet center frequency features are most important for discerning the mental stress state of the subject. The results indicate that features constructed from frequency bands 270-290Hz and 380-390Hz in the PCG data are essential for the classifier to make good predictions. Lower wavelet center frequency features also play some importance in the predictions. However, their contribution are more negligible.

The four most important features from STAI-30-30 and STAI-35-20 are displayed as horizontal dashed lines in figures 5.19 and 5.20, respectively. The figures compare the same low-stress and high-stress labeled recordings present in both STAI-30-30 and STAI-35-20.

Discerning from the plots, $\widetilde{S}_1$ coefficients appear most significant for both dataset splits in lower energy/amplitude frequency ranges. In contrast, the $\widetilde{S}_2$ coefficients appear most significant for both dataset splits in higher energy/amplitude frequency ranges.

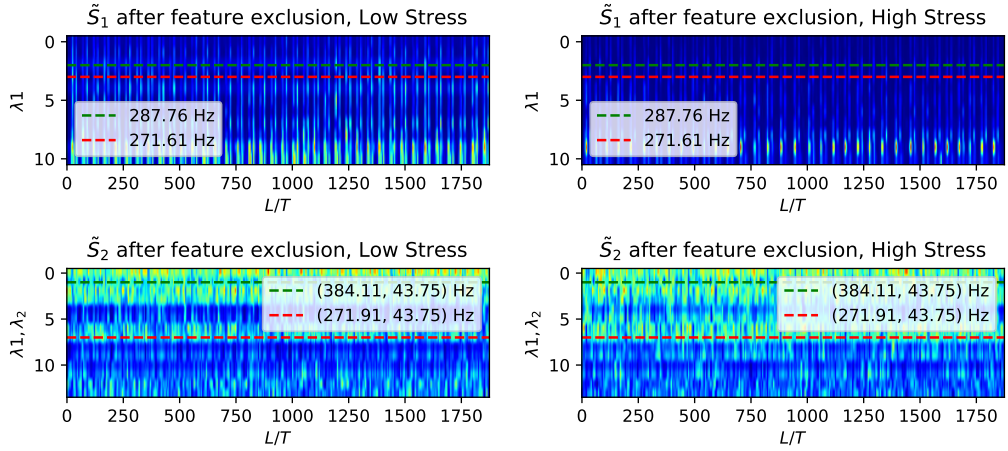


Figure 5.19: Normalized scattering after feature exclusion $(L, J, Q) = (30000, 4, 12)$. The data displayed is: low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress from STAI-30-30. The four most important frequencies for STAI-30-30 are marked as dashed horizontal lines.

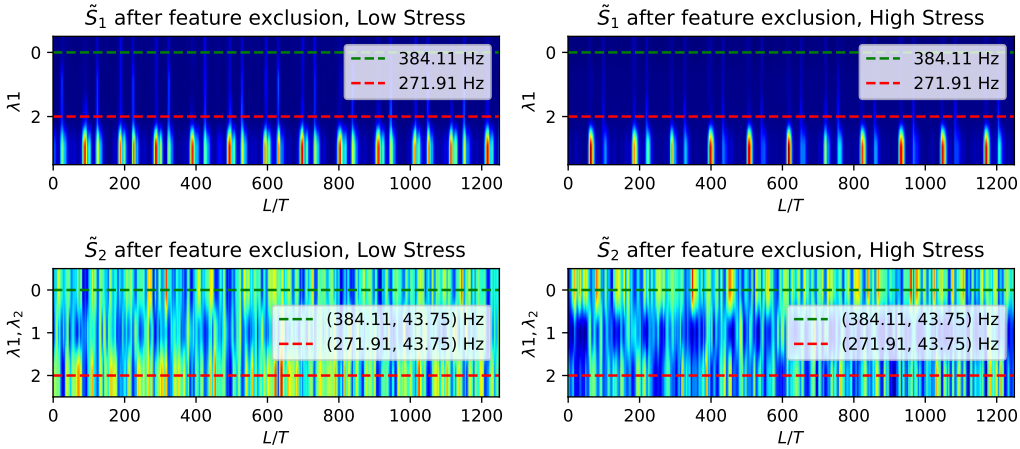


Figure 5.20: Normalized scattering after feature exclusion $(L, J, Q) = (10000, 3, 12)$. The data displayed is: low-rated ('P017_S002_001') stress and a high-rated ('P009_S001_002') stress from STAI-35-20. The four most important frequencies for STAI-35-20 are marked as dashed horizontal lines.

6

Discussion

This chapter provides in-depth reasoning and analysis of various aspects related to the collection, preprocessing, and classification of the Multimodal Mental Stress dataset. The protocol design, PCG auscultation setup, label selection, noise mitigation by feature exclusion, machine learning techniques, evaluation metrics, and potential future work are thoroughly examined, shedding light on the challenges and opportunities in the field of mental stress classification using physiological signals.

6.1 Mental Stress Dataset

A big part of the thesis work was collecting the final part of the Multimodal Mental Stress Dataset in January 2023, continuing the collection started as part of the specialization project in autumn 2022. Because the dataset is quite complex, its use cases go beyond PCG and STAI labels. The research done from the Mental Health dataset in this thesis, analyzing the subject demographic and labels, and doing STAI stress label classification from the PCG data with machine learning techniques, is just the tip of the iceberg. Ideas for further work with the dataset will be discussed in the final section of the discussion (section 6.3). This section will cover the design choices made for the dataset collection and discuss the collected dataset in detail from the analysis done in chapter 3.

6.1.1 Inclusion Criteria

To select suitable subjects for the dataset, we constructed a list of inclusion criteria that the subjects needed to fulfill to be part of the study. These criteria are covered in section 3.2 and were made to ensure participants were healthy students with no current or previous diseases affecting the physiological measurements. Potential subjects with mental health conditions were also excluded as the aim was to make ML models for early detection of mental stress, which can lead to more serious mental conditions if sustained over longer

periods of time.

To ensure the dataset would have a good distribution of stress states from study participants, another requirement for inclusion was that the student had a scheduled exam or project delivery after the recording session performed in December 2022 (S001). Previous studies show that this is an effective way to obtain high-stress-state recordings since students undergo a high degree of mental pressure before exams and deadlines [11]. The criterion was not a requirement for the baseline recordings done in January 2023 (S002), enabling new participants to opt-in. From the dataset analysis done in chapter 3, we see from figure 3.3 that this inclusion criterion worked as intended with higher averaging STAI scores attained from the recordings done in S001 than in S002. However, a lot of the subjects obtained STAI scores in the middle of the scale, which in term will deem the related PCG recording close to useless for training a binary classifier later on since the recording represents a moderate stress state (not low stress, nor high stress).

In addition to the inclusion criteria, we paid close attention to gender and age when selecting participants to obtain an even age and gender distribution in the dataset. Training the model on both male and female data from different age groups will make it broader and more relevant for clinical use. As the analysis from figures 3.5 and 3.6 shows, both age and gender distributions had a decent spread. This distribution could be more even and cover larger age ranges if the dataset is expanded at a later time.

6.1.2 Two Runs per Session

As part of the recording protocol, we recorded two 5-minute runs per subject per session (run 001 and run 002). Run 001 was recorded without external stimuli, while run 002 was recorded while the subject did an arithmetic test. The arithmetic test was introduced to elevate the subject's state stress after run 001. Inducing extra state stress from a mental task was intended to further increase discriminability over stress states, as seen from the biosignal. In this thesis, data from both runs are processed the same since the dataset size is too small to make performance a model on only run 001 or run 002 recordings.

Looking at the STAI label results in figure 3.3, the arithmetic test did lead to a slightly higher STAI median score for run S002, where the subjects were more relaxed after the holidays. Counteractively, this makes the run 002 recordings of S002 less usable for training a binary classifier, as the arithmetic test made the originally low-stress students moderately stressed. Thus this data is edge case data and unusable for binary classifier training since it does not accurately describe a class. It makes introducing the arithmetic test work against its purpose, reducing the potential low-stress support. Because of this, the final model performance could be somewhat affected, especially in terms of sensitivity score which benefits from a larger low-stress support. However, further research with the Mental Stress dataset could still use the arithmetic test markers for another purpose, and therefore its introduction is not entirely worthless. For the S001 recordings, the arithmetic test did not significantly impact the STAI scores with the same median as run 001.

6.1.3 PCG Auscultation setup and Systematic Noise

Subjects of the dataset were recorded in a sitting position with the digital stethoscope placed at Erb's auscultation point. The sitting position was chosen since half the recordings were to be done with the arithmetic test (sitting upright next to a computer), and since EEG was to be measured simultaneously, making other positions problematic. Since little is known about how specific components of the heart cycle relate to stress response, Erb's point was chosen as the auscultation point as it gives a good compromise in the loudness of S1 and S2 heart sounds marking important components of the heart cycle.

To capture PCG in synchronization with EEG and arithmetic test markers, LSL and AC were used to record the PCG audio output of the Eko Duo by connecting the digital stethoscope to the sound card of the computer with an aux cable. As advised by Doctor Andres Soler, this allowed for recording time-synchronized data over the wireless recording limit of the Eko Duo (120 seconds) with a wired connection. This seemed to be a good compromise for the Eko Duo's limitations at the time. However, using this setup would lead to electronic component- and power line noise in the final PCG data.

Explored in section 3.4.2, the PCG Mental Stress dataset, unfortunately, has systematic noise at 50 Hz and 60 Hz and at harmonics of these frequencies (see figure 3.7). The spikes in the power spectrum at these frequencies most likely appear at 60 Hz and harmonics due to the modified wired aux connection to the computer sound card where the Norwegian power line frequency pollutes the data. The 50 Hz and harmonic peaks most likely appear due to ECG electrode lift artifacts since the US-designed circuitry runs on a 50 Hz battery supply in the Eko Duo digital stethoscope.

If a new dataset is to be recorded, it is advised to record data with another measurement device better suited for wired recording without modifications or use the Eko App interface with wireless recording which eliminates the systematic noise peaks as shown in figure 3.8.

6.1.4 Dataset Size

A recording time of five minutes per recording per run was selected to obtain a sizable dataset from the number of participants we had time to record in the selected time frame of two months. For PCG data, a time frame of 5 minutes also technically allows for heart rate variability calculations if a synchronous segmentation procedure is performed. From the 83 PCG recordings conducted, the dataset contains 415 minutes of PCG data at 22050 Hz.

6.1.5 STAI Binary Label Split

Individual bias affects the STAI scores in ways that are hard to control. A person might underestimate or overestimate his/her stress state leading to uncertainty in whether the obtained STAI score is representative of the subject's actual stress state. For this reason, the STAI labels should not be treated as an exact estimate of the stress-state, and we should rather assume that most labels are rough estimates of the stress state. Treating stress level prediction as a regression problem is therefore a bad idea since there is not necessarily a

linear relationship between STAI score and actual stress state, especially not on an inter-individual level. Because of the uncertainties of the STAI-labels, they were simplified to cover specific ranges of STAI scores to minimize uncertainty between label classes.

Initially, a three-label split of low-, moderate-, and high-stress classes was tested to use all PCG recordings in the dataset. This split gave models with bad performance, with too many moderate-stress samples being classified as low-stress or high-stress, or vice versa. The reason for this most likely comes from the uncertainty in the STAI score, and thus we need larger STAI score class ranges to discriminate between classes accurately. Therefore the choice was made to use binary stress labels where only the recordings with STAI scores in the upper and lower percentage of the STAI score-interval were included. This would guarantee that we got larger STAI-score class ranges and leave out edge case recordings in the middle of the STAI-score interval.

Two binary STAI labels were used, STAI-30-30 and STAI-35-20. A sorted list of the recording's STAI scores was used to construct these splits. STAI-30-30 uses the first 30% of recordings in the list (inferred as low-stress) and the last 30% of recordings in the list (inferred as high-stress). This split was selected as it gave similar supports for the two classes with an equal number of low- and high-stress recordings while leaving out moderate-stress edge case recordings. STAI-35-20 was made to give a more uniform STAI-scale distribution between the two classes. Using the 35% first recordings in the sorted list (inferred as low-stress) and the last 20% of recordings in the sorted list (inferred as high-stress), the STAI scale ranges for high and low stress are roughly the same size.

Because of the uncertainty in the STAI-labels, it is a fair assumption to think that many recordings are mislabeled somehow, even with STAI scores reduced to binary labels. An ML model trained on features from the data and binary STAI labels will reach good accuracy since most labels are representative of either a low- or a high-stress state. Regardless, the model is going to predict the wrong label for recordings with misleading labels. This means that reaching perfect accuracy on predictions is unobtainable without some sort of data relabeling procedure. See the future work section, which discusses such a relabeling procedure.

6.2 Machine Learning

6.2.1 Segmentation Choices

All previously written research covering the topic of this thesis uses features from synchronously segmented PCG data to classify mental stress states [11], [12]. Using asynchronously segmented data introduces some notable benefits and difficulties. By asynchronous segmentation, we either remove the dependence on having a parallel ECG signal to find S1-S1 peaks (thus segmenting each heart cycle) or remove the reliance on using advanced PCG synchronous segmentation algorithms (see section 2.1.3). Instead, by segmenting with a fixed temporal length, the start and end of each segment can be at any position in the heart cycle. A result of this is that features that retain temporal information created from asynchronous segments will have high variance, especially for small and medium segmentation lengths under the duration of the heart cycle. In comparison,

features that retain temporal information from synchronously segmented recordings will have less variance since the PCG waveform segment, each corresponding to a heart cycle, will have a similar shape with slight differences in duration. A solution for this is either using fully frequency-based features on the asynchronous segments with long segmentation lengths or using time-frequency features with good temporal invariance to reduce the variance between segments, thus making an ML model able to find similar structures in features. Wavelet scattering is used in this thesis and yields time-frequency representations with tunable characteristics in terms of time and frequency resolution, which is optimal for this problem.

In PCG signal analysis, high frequency resolution is beneficiary for discerning the whats of the heart cycle. For small time segments, a fine frequency resolution will, for example, be better able to discern which heart sound or which part of a heart sound the segments covers. For time segments lasting approximately one heart cycle (800 ms), a fine frequency resolution can better pinpoint exactly which heart sounds are present and their intensity, in addition to giving information on which murmurs and bowel sounds are present. For longer segments lasting multiple heart cycles, we can attain a more averaged spectral content over the segment in fine detail and gain a better perspective on which sound frequencies carry the highest intensities over many heart cycles. For wavelet scattering features, the frequency resolution is set by the wavelets per octave parameter (Q). The wavelets are distributed over the frequency axis as explained in the filterbank section of chapter 2 (see section 2.3.3).

Temporal resolution should not be neglected, as heart sound timing and heart rate are affected by emotional and physiological stress [18]. We need a sufficient temporal resolution to represent the order of events in the heart cycle in our segment features. For wavelet scattering features, the temporal resolution is set by the invariance scale of the network ($T = 2^J$) and is equal to T/f_s seconds, where f_s is the PCG signal's sampling frequency.

See below (section 6.2.3) for an overview covering the scattering network optimization and which time- and frequency resolutions were found to be optimal for the Mental Stress Dataset.

6.2.2 Inter-subject Classifier

A critical notion when splitting the ML dataset is to keep each subject's data separated into different sets. We do not want the model to be trained on the same subject data as it predicts since we want a usable model on unseen data. Therefore this is a requirement to obtain a good inter-subject classifier. Training the data on the same subjects we predict will result in close to perfect accuracies since the classifier will find intra-subject similarities and recognize the subject's recording instead of finding inter-subject similarities and differences in the stress state.

The Mental Stress dataset with STAI binary labels was divided with the stress heuristic introduced in section 4.3.1 into 10 sets with no overlapping recordings and no overlapping subjects. These ten recording sets were used for all subsequent 10-fold cross-validation procedures in model creation and evaluation.

6.2.3 Normalized Scattering Features

Scattering features were implemented with Kymatio, and the normalized scattering transform was computed with self-implemented code. As hinted above, the parameters J and Q set the time-frequency resolution and tiling of the wavelets in the filter bank constructed by Kymatio. The signal segments sent to scattering have length L. The scattering output is L/T feature frequency vectors, where each feature vector represents spectral PCG content over temporal length T, and the temporal modulation context of the original segmentation length L is kept with the normalized scattering transform. In a 10-fold cross-validation procedure, the classifier is trained on feature frequency vectors from all segments in the training set and then predicts on L/T similar future vectors for each segment of length L in the validation set. The average prediction of the L/T feature vectors becomes the final prediction for the validation segment.

As explained in section 4.3.3, segmentation length, network invariance scale, and wavelet octave resolution were optimized in a grid search for both STAI-30-30 and STAI-35-20. This optimization was referred to as (L, J, Q) optimization in chapter 5.1.

Looking at the (L, J, Q) optimization 10-fold cross-validation results for STAI-30-30 (figure 5.1), low invariance scales (J), and moderate to high frequency resolutions, provides the best average 10-fold accuracies. The best results are seen for $L = 30000$, $J = 4$, and $Q = 12$. A network invariance scale of $T = 2^3 = 8$ samples indicates that each of the L/T frequency vectors of a segment has a temporal context of 16 ms, with a temporal modulation context of 30000 samples or 30 seconds. Using such a small invariance scale increases the variance over features, but the amount of training samples fitting the classifier (L/T per segment) is large and makes up for this. Every time J is decremented by 1, the number of training samples doubles. From the segmentation lengths tested, changing L between 10-30 seconds did not drastically change the average accuracy over the (J, Q) grid. However, a distinction is seen for the highest Js (11,12,13), in that longer segmentation lengths generally increased average accuracy for $15 \leq Q \leq 20$. With $J = 13$, the spectral information of one feature vector is averaged over 8.2 seconds, meaning that temporal context over this interval is lost at the cost of higher frequency resolution. As expected, because of the time-frequency tradeoff, we can afford to use higher quality factors (Q) for the larger invariance scales, strictly prioritizing frequency resolution. We do not achieve as good performance by using high Js as compared to low Js. Choosing $J = 4$ yields 512 times more training samples than choosing $J = 13$. If the performance for larger Js with a larger dataset would catch up or overtake the performance of lower Js is hard to predict but can be tested if the dataset is expanded at a later date.

From (L, J, Q) optimization for STAI-35-20 (figure 5.3) we see similar results to STAI-30-30. Here, $L = 10000$, $J = 3$, and $Q = 12$ give the best average 10-fold accuracies. The same quality factor still yields the best results, however, for this binary label split we see slightly increased performance for lower L and lower J. The performance differences between different segmentation lengths are marginal enough to propose that tuning the hyperparameters of XGBoost for the same set of (J, Q) values but different L in the chosen range would yield very similar final results.

An interesting result can be drawn from the fact that $Q = 12$ is found to be optimal for both STAI-30-30 and STAI-35-20. Recall from background material that the wavelet center frequencies are defined as $\xi_{\lambda_{i,j}} = a^{-j}, j \in \mathbb{Z}, a = 2^{1/Q}$. On a log-frequency axis we then get the following relationship between center frequencies:

$$\log_2 \xi_{\lambda_{i,j}} = \log_2 2^{-j/Q} = -j/Q, \quad j \in \mathbb{Z}, \tag{6.1}$$

i.e., a uniform quantization of the log-frequency axis with a step $1/Q$. For $Q = 12$, choosing $j = -19$ and $j = -28$, the scaling factor of the wavelets approximately equals the integers 3 and 5 ($2^{19/12} \approx 3$ and $2^{28/12} \approx 5$), which corresponds to Pythagorean intervals of perfect fifth and major third [27]. Therefore the quality factor in wavelet scattering is often set to 12 (or multiples of 12) in musical instrument classification applications, as the wavelet center frequencies match the 12-tone equal temperament (12-TET) scale often used in Western music. The fact that this wavelet octave resolution is found to be optimal for mental stress classification from PCG data could be a coincidence. However, the Pythagorean series tends to appear in different natural phenomena, and heart sound vibration frequencies may be another example.

6.2.4 Feature Exclusion

As explained in section 4.2.3, a method for removing the impact the PSD spikes of the data had on the machine learning results, a feature exclusion technique was implemented. This technique proved to be a good compromise to aid the systematic noise in the dataset. Combining peculiar theoretical knowledge from wavelet scattering theory, coefficient features attained from wavelets with frequential bandwidths in the issue frequency bands shown in figure 4.4 were removed from the final feature vectors. From figure 5.4 and figure 5.2, we see that the coefficient features related to noise bands present in the scattering features were successfully removed (light blue horizontal lines disappear). Compared to techniques such as notch filtering and adaptive filtering, this technique was preferred since the original normalized and resampled PCG data did not have to undergo filtering, which would change the underlying PCG waveform and potentially be more harmful to classification performance. With the used method, the normalized scattering transform was computed straight from the normalized and resampled PCG data, and the feature exclusion was done subsequently.

Removing features from issue frequencies might have affected the (L, J, Q) optimization in that the center frequencies of the wavelets in the filter bank have to line up with important frequency ranges for mental stress detection, not covered by issue frequency intervals. For clean PCG data with no systematic noise spikes in the power spectrum, feature exclusion would not be necessary, and other (L, J, Q) parameters could have been found to be more optimal. For reference, when testing (L, J, Q) optimization with feature vectors untouched by the feature exclusion, higher wavelet octave frequency resolutions (higher Q) gave superior performance. This was because the classifier could rely on coefficient features from wavelets with bandwidths specifically covering the issue frequency spikes. Thereby just making predictions based on spike intensities and recognizing inter-recording differences in spike intensities connecting recordings to a particular stress label rather than

making predictions on the actual PCG signal properties relevant to identifying the stress state of the subject. Therefore, feature exclusion was implemented to prevent the final model from being useless when predicting recordings without noise spikes and potentially make the model viable for recordings done with other digital stethoscopes.

The feature exclusion most likely reduced the potential performance of the model compared to if we made a model from data free from the systematic noise without feature exclusion. Using a new noise-free dataset with STAI labels, the method presented in this thesis could be retested without feature exclusion to see if this hypothesis is correct.

6.2.5 Hyperparameter Optimization

After (L, J, Q) optimization, the hyperparameters of KNN and XGBoost classifiers were optimized with a meta-modeling procedure. Hyperopt found the optimal parameters over 100 10-fold cross-validation evaluations for XGBoost and over 50 evaluations for the KNN classifier. The hyperparameter spaces were defined as summarized in section 4.3.3, decided by typical hyperparameter ranges found in the respective classifiers' documentation (see appendix C).

For XGBoost, 8 hyperparameters were tuned over 100 10-fold cross-validation procedures per binary label split. As summarized in table 5.3, the optimal hyperparameters for STAI-30-30 and STAI-35-20 are quite different. The STAI-30-30 hyperparameters yield a classifier with more and deeper CARTs than STAI-35-20. The learning rate of the classifier is also set to a lower number for STAI-30-30 than STAI-35-20. This indicates that a more advanced classifier is needed to achieve accurate class discrimination for STAI-30-30. This makes sense since STAI-30-30 defines a larger percentage of the upper STAI-scores as high-stress than STAI-35-20, and some of these recordings probably are bad examples of true high-stress recordings. This all comes down to the uncertainty of the STAI labels. For the low-stress defined recordings in STAI-35-20, the extra 5 percent of recordings in the low range of the STAI scale not included in STAI-30-30 seems to generalize better as low-stress data since the model can easily find class discrimination patterns with a less advanced classifier. This indicates that we could have included a new dataset split STAI-35-30, yielding the same performance or better than STAI-30-30 providing more training data for the model.

Looking at the summarized KNN optimal hyperparameters in table 5.2, a lower number of neighbors is needed for STAI-35-20 to reach optimal performance than for STAI-30-30. This repeats the same message as for the XGBoost classifier in that more recordings feature distances need to be compared for STAI-30-30 to achieve good results than for STAI-35-20, indicating a more advanced model. For STAI-30-30 the Manhattan Distance measure $(p = 1)$ works best. This is probably because we have higher dimensional feature vectors for STAI-30-30 than for STAI-35-20 since $J = 4$ yields higher dimensional features than $J = 4$ with $Q = 12$. As explored in [1]: "for a given problem with a fixed (high) value of the dimensionality d, it may be preferable to use lower values of k. This means that the L1 distance metric (Manhattan Distance metric) is the most preferable for high dimensional applications, followed by the Euclidean Metric (L2), then the L3 metric, and so on", where the mentioned k is the Minkowski distance parameter referred to as p in this thesis. This

coincides with the optimal hyperparameter p obtained for STAI-35-20 and STAI-30-30.

6.2.6 Final Model

Judging from the final evaluation of the classifiers in table 5.4, the XGBoost classifier generally performs superior to the KNN classifier and creates better models with higher AUC scores. The difference in performance is most notable for STAI-35-20 where XGBoost achieves 3.52% better average 10-fold accuracy than KNN. However, KNN still serves as a good baseline in performance and holds up quite well when considering algorithm complexity. Comparing the final performance results of XGBoost to the results before hyperparameter optimization, we see that the optimization increased the average 10-fold accuracy on STAI-30-30 by 0.2% (from 0.8490 to 0.8510) and increased the average 10-fold accuracy on STAI-35-20 by 1.56% (from 0.9274 to 0.9430). This performance difference is notable since the accuracies already were quite high.

We can draw notable differences between the classifiers from the confusion matrices in figure 5.15 and sensitivity and specificity scores in table 5.4. All binary label split and classifier combinations achieve better sensitivity scores than specificity scores (i.e., more FNs than FPs). This is expected for STAI-35-20 since we have less high-stress support. For STAI-30-30, the reason might be that we most likely have some high-stress labeled recordings that do not accurately represent the high-stress class as explored in the previous subsection. Looking at the confusion matrices for STAI-35-20 in figure 5.15c and 5.15d, the XGBoost classifier better adapts to the low high-stress support by sacrificing some sensitivity for a lot better specificity than the KNN classifier, which in term bumps the overall accuracy. Sensitivity and Specificity for STAI-30-30 between the two classifiers are close to similar.

6.2.7 ROC AUC

The ROC AUC results in section 5.3 showcase how each fold in 10-fold cross-validation affects the overall model performance and how it generalizes to the different validation sets. The results give further indications that we are dealing with a great portion of mislabeled data. From figure 5.16a and 5.16b displaying STAI-30-30 ROC-AUC, 7 of the folds' validation sets generalize perfectly or close to perfectly to the training data (AUC-scores close to 1), whilst fold 1,2 and 5 lack in performance.

For STAI-35-20 in figure 5.16c and 5.16d, all folds except fold 6 generalize perfectly, indicating that one of the recordings in the 6th fold's validation set has a misleading label.

Using a leave one subject out (LOSO) or a leave one recording out (LORO) cross-validation procedure it would be possible to pinpoint exactly which subjects or recordings have misleading labels and possibly relabel them to obtain a dataset that could yield a more stable model with better AUC and accuracy. A more precise relabeling procedure is proposed below (see section 6.3.2).

6.2.8 Important Frequencies for Mental Stress Detection from PCG

An interesting feature of XGBoost combined with wavelet scattering theory lets us explore what frequencies in the PCG data are important for mental stress detection. This relationship can be drawn from the frequency/scale used to dilate the wavelet for each scattering feature in the frequency feature vector. Since XGBoost is a gradient-boosting algorithm it can give the relative importance each feature coefficient has for the classifier's predictions.

Figure 5.17 and figure 5.18 show the importance of each feature coefficient for STAI-30-30 and STAI-35-20, respectively. Looking at the bar plots, it is apparent that the same frequency ranges are essential for both binary dataset label splits. Specifically, scattering features from wavelets with center frequencies in the ranges 270-290 Hz and 380-390 Hz appear to be most important for the XGBoost prediction. This is of great interest for further research on stress detection from PCG signals since features can be specialized to pinpoint these exact frequency intervals. Furthermore, finding that these heart sound frequencies relate to mental stress is crucial for understanding the underlying heart mechanisms elevated by the stress response. To my knowledge, these connections have not been drawn in any other research.

6.3 Future Work

6.3.1 Potential Pipeline Improvements

Because of time limitations, (L, J, Q) optimization was not performed for the KNN classifier. Running this could yield slightly better final performance for the KNN classifier if other (L, J, Q) parameters are found to be more optimal.

By downsampling to a higher sampling frequency than 1000 Hz, performance could possibly be improved since this will allow for scattering on higher frequency harmonics that could be useful for stress detection. This would however come at the cost of longer computation times unless more powerful hardware is used.

Increasing the dataset size would allow us to have a separate hold-out test set for the final evaluations. This would give a better indication of how the fine-tuned model after optimization works on unseen data. Reusing the 10-fold cross-validation for the final model evaluation was a compromise that needed to be taken because of the small dataset size (low subject number) and because of the uncertainty in the STAI labels. Using one or two subject recordings with potentially misleading labels as the test set would with low probability give an accurate performance evaluation. A relabeling procedure would eliminate most issues related to this.

6.3.2 Automatic Relabeling

As a much-needed step to improve the uncertainty seen in STAI labels, an automatic relabeling procedure is proposed. Using similar principles to what is done by Hao et al. in [19], mislabeled data can be identified through either a cross-entropy loss or an influence function. Thereafter, the labels of expected mislabeled data are flipped, and the model is

trained with the new labels before being retested in performance. Using this method on mammography data with a varying percentage of intentionally flipped labels Hao et al. achieved a close to perfect detection rate of the flipped labels, drastically increasing the model AUC when retrained on the relabeled data.

This method can be reformatted to suit the STAI labeling uncertainty issue of the Mental Stress Dataset by using wavelet scattering features from the data similar to the method in this thesis and a logistic classifier with a cross-entropy loss function. From there, identify the most likely mislabeled data, flip the labels, and retrain the model.

6.3.3 Joint Time-Frequency Scattering

The scattering method used in this thesis is the simplest form of wavelet scattering, namely temporal scattering. Temporal scattering has many important properties explained in this thesis background material (see section 2.3.3), such as time-shift invariance and stability to time-warping deformations. However, a more intricate scattering method, joint time-frequency scattering (JTFS), can also be tested in further work on the Mental Stress Dataset. JTFS's primary additions to temporal scattering are sensitivity to frequency-dependent time shifts and frequency transposition invariance (invariance to shifts in log frequency). In JTFS, the scattering procedure is similar to temporal scattering up to the second order, but for the second order, scattering is done both over frequency and time with a joint filter bank from the scalogram of the input signal. This creates the extra invariance in the features. For a mathematical explanation, please refer to [29].

Further increasing invariance in the PCG scattering features could increase the classifier's high- and low-stress class discrimination, thus improving the final model's performance.

6.3.4 Heart-Brain Coherence

The Mental Stress Dataset has use cases beyond what is explored in this thesis. Since the dataset contains time-synchronized EEG and PCG data, a heart-brain coherence analysis could be performed. Coherence conceptualizes how information flow within and between physiological systems in the body plays an important role in determining experienced feelings and emotions [31]. Studies have shown that the alpha rhythm activity of the brain is synchronized with the cardiac cycle. Moreover, the synchronization increases during psychophysiological coherence [30]. By utilizing the STAI stress labels, PCG, and EEG data, differences in heart-brain interactions over different stress states can be examined. Furthermore, using the time-synchronized arithmetic test markers, one can examine how heart-brain interactions change when a subject is given an arithmetic question and how the subsequent heart-brain interaction leads to a false or correct answer.

7

Conclusion

Finding inter-subject PCG signal differences and similarities with changing stress states is a complex task well suited for ML with a sufficiently large dataset. Summarized in three steps, the thesis aimed to finish the collection of a Multimodal Mental Stress dataset, conduct an in-depth dataset analysis, and infer machine learning techniques on the PCG and STAI label part of the dataset.

The remainder of the Mental Stress dataset was collected in January 2023. Comprising 83 multimodal recordings from 10 female and 16 male students attending NTNU under varying states of stress coupled with STAI and SSA stress labels, the dataset lays a foundation for multiple areas of study.

Four models using wavelet scattering features and KNN or XGBoost classifiers were made (two for each binary label STAI split). The models were optimized concerning asynchronous segmentation lengths (L), wavelet scattering network parameters (J, Q), and classifier hyperparameters. The KNN / STAI-30-30 model achieved a sensitivity of 0.8846, specificity of 0.7960, AUC of 0.8403, and accuracy of 0.8412. The XGB / STAI-30-30 model achieved a sensitivity of 0.9000, specificity of 0.8000, AUC of 0.8503, and accuracy of 0.8510. The KNN / STAI-35-20 model achieved a sensitivity of 0.9881, specificity of 0.7694, AUC of 0.8788, and accuracy of 0.9078. Lastly, the XGB / STAI-35-20 model achieved a sensitivity of 0.9796, specificity of 0.8801, AUC of 0.9298, and accuracy of 0.9430. The models created from STAI-35-20 perform better and are significantly more stable than models created from STAI-30-30. Uncertain and misleading STAI labels from a higher amount of STAI-30-30 recordings are most likely the reason for the performance discrepancies. Out of the models, the XGB / STAI-35-20 model is recommended for stress diagnosis as it builds on the most stable label split and a better classifier based on the performance results.

The thesis has introduced a framework for wavelet-scattering-based noise-band removal by feature exclusion as part of the model pipeline. From the results in the thesis, this method

showed to be highly efficient and is recommended for similar noise-band-related issues in similar problems. PCG frequencies associated with mental stress states were found by looking at feature importance from XGBoost and relating the importance to underlying wavelet center frequencies of the frequency feature vectors. The frequency ranges 270-290 Hz and 380-390 Hz were found to be of the highest importance for discerning stress states from the PCG data. This finding should be of great interest for future research covering mental stress detection from PCG data, as features can be pinpointed at these frequency intervals, potentially simplifying the features and achieving better performance.

Bibliography

[1] Aggarwal, C.C., Hinneburg, A., Keim, D.A., 2001. On the surprising behavior of distance metrics in high dimensional space, in: Van den Bussche, J., Vianu, V. (Eds.), Database Theory — ICDT 2001, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 420–434.

[2] Altman, D.G., Bland, J.M., 1994. Statistics notes: Diagnostic tests 1: sensitivity and specificity. BMJ 308, 1552. URL: *https://www.bmj.com/content/308/6943/1552*, doi:*10.1136/bmj.308.6943.1552*.

[3] Andreux, M., Angles, T., Exarchakis, G., Leonarduzzi, R., Rochette, G., Thiry, L., Zarka, J., Mallat, S., andén, J., Belilovsky, E., Bruna, J., Lostanlen, V., Chaudhary, M., Hirn, M.J., Oyallon, E., Zhang, S., Cella, C., Eickenberg, M., 2018. Kymatio: Scattering transforms in python. URL: *https://arxiv.org/abs/1812.11214*, doi:*10.48550/ARXIV.1812.11214*.

[4] Andén, J., Mallat, S., 2013. Deep scattering spectrum URL: *http://arxiv.org/abs/1304.6763http://dx.doi.org/10.1109/TSP.2014.2326991*, doi:*10.1109/TSP.2014.2326991*.

[5] Arnott, P., Pfeiffer, G., Tavel, M., 1984. Spectral analysis of heart sounds: Relationships between some physical characteristics and frequency spectra of first and second heart sounds in normals and hypertensives. Journal of Biomedical Engineering 6, 121–128. URL: *https://www.sciencedirect.com/science/article/pii/0141542584900542*, doi:*https://doi.org/10.1016/0141-5425(84)90054-2*.

[6] Arts, L.P.A., van den Broek, E.L., 2022. The fast continuous wavelet transformation (fCWT) for real-time, high-quality, noise-resistant time–frequency analysis. Nature Computational Science 2, 47–58. URL: *https://doi.org/10.1038/s43588-021-00183-z*, doi:*10.1038/s43588-021-00183-z*.

[7] Barnes, L.L., Harp, D., Jung, W.S., 2002. Reliability generalization of scores on the spielberger state-trait anxiety inventory. Educational and psychological measurement 62, 603–618.

[8] Bergstra, J., Yamins, D., Cox, D., 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures, in: Dasgupta, S., McAllester, D. (Eds.), Proceedings of the 30th International Conference on Machine Learning, PMLR, Atlanta, Georgia, USA. pp. 115–123. URL: *https://proceedings.mlr.press/v28/bergstra13.html*.

[9] Boulay, C., Stenner, T., Medine, D., Bellotti, A., 2020a. App-audiocapture. *https://github.com/labstreaminglayer/App-AudioCapture*.

[10] Boulay, C., Stenner, T., Medine, D., Bellotti, A., 2020b. Labstreaminglayer. *https://github.com/sccn/labstreaminglayer*.

[11] Cheema, A., Singh, M., 2019. Psychological stress detection using phonocardiography signal: An empirical mode decomposition approach. Biomedical Signal Processing and Control 49, 493–505. URL: *https://www.sciencedirect.com/science/article/pii/S1746809418303276*, doi:*https://doi.org/10.1016/j.bspc.2018.12.028*.

[12] Cheema, A., Singh, M., Kumar, M., Setia, G., 2023. Combined empirical mode decomposition and phase space reconstruction based psychologically stressed and non-stressed state classification from cardiac sound signals. Biomedical Signal Processing and Control 82, 104585. URL: *https://www.sciencedirect.com/science/article/pii/S1746809423000186*, doi:*https://doi.org/10.1016/j.bspc.2023.104585*.

[13] Chen, T., Guestrin, C., 2016. XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA. pp. 785–794. URL: *http://doi.acm.org/10.1145/2939672.2939785*, doi:*10.1145/2939672.2939785*.

[14] Chu, B., Marwaha, K., Sanvictores, T., Ayers, D., 2023. Physiology, stress reaction, in: StatPearls. StatPearls Publishing, Treasure Island (FL).

[15] Eickenberg, M., Exarchakis, G., Hirn, M., Mallat, S., Thiry, L., 2018. Solid harmonic wavelet scattering for predictions of molecule properties URL: *https://arxiv.org/abs/1805.00571*, doi:*10.48550/ARXIV.1805.00571*.

[16] Everly, G.S., Lating, J.M., 2013. The Anatomy and Physiology of the Human Stress Response. Springer New York, New York, NY. pp. 17–51. URL: *https://doi.org/10.1007/978-1-4614-5538-7_2*, doi:*10.1007/978-1-4614-5538-7˙2*.

[17] Fernando, T., Ghaemmaghami, H., Denman, S., Sridharan, S., Hussain, N., Fookes, C., 2020. Heart sound segmentation using bidirectional lstms with attention. IEEE Journal of Biomedical and Health Informatics 24, 1601–1609. doi:*10.1109/JBHI.2019.2949516*.

[18] Giannakakis, G., Grigoriadis, D., Giannakaki, K., Simantiraki, O., Roniotis, A., Tsiknakis, M., 2019. Review on psychological stress detection using biosignals. IEEE Transactions on Affective Computing 13, 440–460.

[19] Hao, D., Zhang, L., Sumkin, J., Mohamed, A., Wu, S., 2020. Inaccurate labels in weakly-supervised deep learning: Automatic identification and correction and their

impact on classification performance. IEEE Journal of Biomedical and Health Informatics 24, 2701–2710. doi:*10.1109/JBHI.2020.2974425*.

[20] Herley, C., Kovacevic, J., 1993. Tilings of the time-frequency plane: Construction of arbitrary orthogonal bases and fast tiling algorithms. IRE Transactions on Audio 41, 3341–3359. doi:*10.1109/78.258078*.

[21] Jiang, Y., Chen, W., You, Y., 2020. Scattering transform-based features for the automatic seizure detection. Biocybernetics and Biomedical Engineering 40, 77–89. URL: *https://www.sciencedirect.com/science/article/pii/S020852161930470X*, doi:*https://doi.org/10.1016/j.bbe.2019.11.002*.

[22] Julian, L.J., 2011. Measures of anxiety: State-trait anxiety inventory (stai), beck anxiety inventory (bai), and hospital anxiety and depression scale-anxiety (hads-a). Arthritis Care & Research 63, S467–S472. URL: *https://onlinelibrary.wiley.com/doi/abs/10.1002/acr.20561*, doi:*https://doi.org/10.1002/acr.20561*.

[23] Khan, K.N., Khan, F.A., Abid, A., Olmez, T., Dokur, Z., Khandakar, A., Chowdhury, M.E.H., Khan, M.S., 2021. Deep learning based classification of unsegmented phonocardiogram spectrograms leveraging transfer learning. Physiological Measurement 42, 095003. URL: *https://dx.doi.org/10.1088/1361-6579/ac1d59*, doi:*10.1088/1361-6579/ac1d59*.

[24] Learn, S., 2007 - 2022. Cross-validation: evaluating estimator performance. *https://scikit-learn.org/stable/modules/cross_validation.html*. Accessed: 2023.14.01.

[25] Li, J., Ke, L., Du, Q., 2019. Classification of heart sounds based on the wavelet fractal and twin support vector machine. Entropy 21, 472. URL: *https://doi.org/10.3390/e21050472*, doi:*10.3390/e21050472*.

[26] Lostanlen, V., 2017a. Convolutional operators in the time-frequency domain. Theses. Université Paris sciences et lettres. URL: *https://theses.hal.science/tel-01559667*.

[27] Lostanlen, V., 2017b. Opérateurs convolutionnels dans le plan temps-fréquence. Ph.D. thesis. URL: *http://www.theses.fr/2017PSLEE012*.

[28] Lostanlen, V., Cohen-Hadria, A., Pablo Bello, J., 2021. One or two frequencies? the scattering transform answers, in: 2020 28th European Signal Processing Conference (EUSIPCO), pp. 2205–2209. doi:*10.23919/Eusipco47968.2020.9287216*.

[29] Mallat, S., 2012. Group invariant scattering. Communications on Pure and Applied Mathematics 65, 1331–1398.

[30] McCraty, R., Atkinson, M., Tomasino, D., Bradley, R.T., 2009. The coherent heart heart-brain interactions, psychophysiological coherence, and the emergence of system-wide order. Integral Review: A Transdisciplinary & Transcultural Journal for New Thought, Research, & Praxis 5.

[31] McCraty, R., Zayas, M.A., 2014. Cardiac coherence, self-regulation, autonomic stability, and psychosocial well-being. Front. Psychol. 5, 1090.

[32] McGee, S., 2018. Chapter 39 - auscultation of the heart: General principles, in: McGee, S. (Ed.), Evidence-Based Physical Diagnosis (Fourth Edition). fourth edition ed.. Elsevier, Philadelphia, pp. 327–332.e1. URL: *https://www.sciencedirect.com/science/article/pii/B9780323392761000391*, doi:*https://doi.org/10.1016/B978-0-323-39276-1.00039-1*.

[33] Moukadem, A., Dieterlen, A., Clifford, G.D., 2016. An open access database for the evaluation of heart sound algorithms. Physiological Measurement 37, 2181–2213. URL: *https://hal.science/hal-01511279*, doi:*10.1088/0967-3334/37/12/2181*.

[34] Navarro, R., Tabernero, A., Cristóbal, G., 1996. Image representation with gabor wavelets and its applications, Elsevier. volume 97 of *Advances in Imaging and Electron Physics*, pp. 1–84. URL: *https://www.sciencedirect.com/science/article/pii/S1076567008700934*, doi:*https://doi.org/10.1016/S1076-5670(08)70093-4*.

[35] Oliveira, J., Castro, A., Coimbra, M., 2014. Exploring embedding matrices and the entropy gradient for the segmentation of heart sounds in real noisy environments , 3244–3247doi:*10.1109/EMBC.2014.6944314*.

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

[37] Peirce, J., Gray, J.R., Simpson, S., MacAskill, M., Höchenberger, R., Sogo, H., Kastman, E., Lindeløv, J.K., 2019. PsychoPy2: Experiments in behavior made easy. Behavior Research Methods 51, 195–203. URL: *https://doi.org/10.3758/s13428-018-01193-y*, doi:*10.3758/s13428-018-01193-y*.

[38] Prasad, R., Yilmaz, G., Chetelat, O., Magimai.-Doss, M., 2020. Detection of s1 and s2 locations in phonocardiogram signals using zero frequency filter , 1254–1258doi:*10.1109/ICASSP40776.2020.9053155*.

[39] Rezaee, K., Khosravi, M.R., Jabari, M., Hesari, S., Anari, M.S., Aghaei, F., 2022. Graph convolutional network-based deep feature learning for cardiovascular disease recognition from heart sound signals. International Journal of Intelligent Systems 37, 11250–11274. doi:*https://doi.org/10.1002/int.23041*.

[40] Ricke, A., Povinelli, R., Johnson, M., 2005. Automatic segmentation of heart sound signals using hidden markov models , 953–956doi:*10.1109/CIC.2005.1588266*.

[41] Singh, S.A., Devi, N.D., Majumder, S., 2022. An improved unsegmented phonocardiogram classification using nonlinear time scattering features. The Computer Journal doi:*10.1093/comjnl/bxac025*.

[42] Singh, S.A., Majumder, S., 2020. Short unsegmented pcg classification based on ensemble classifier. Turkish Journal of Electrical Engineering and Computer Sciences 28, 875–889. doi:*10.3906/elk-1905-165*.

[43] Øystein Sletta, 2023. Pcg-mental-stress-master-thesis. *https://github.com/oyste/PCG-Mental-Stress-Master-Thesis*.

[44] Sletta, Ø.S., 2023. Classifying unsegmented phonocardiogram signals using cepstral, temporal, and wavelet scattering features URL: *https://rgdoi.net/10.13140/RG.2.2.11435.11046*, doi:*10.13140/RG.2.2.11435.11046*.

[45] Sletten, C., 2023. Automatic Detection of Mental Stress Responses from Electroencephalogram Signals. Ph.D. thesis.

[46] Verma, A., Singh, Y., 2015. Adaptive tunable notch filter for ecg signal enhancement. Procedia Computer Science 57, 332–337. URL: *https://www.sciencedirect.com/science/article/pii/S1877050915018761*, doi:*https://doi.org/10.1016/j.procs.2015.07.347*. 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015).

[47] Vitasari, P., Wahab, M.N.A., Herawan, T., Othman, A., Sinnadurai, S.K., 2011. Re-test of state trait anxiety inventory (stai) among engineering students in malaysia: Reliability and validity tests. Procedia - Social and Behavioral Sciences 15, 3843–3848. URL: *https://www.sciencedirect.com/science/article/pii/S1877042811009293*, doi:*https://doi.org/10.1016/j.sbspro.2011.04.383*.

[48] Wu, J., Jiang, L., Han, X., Senhadji, L., Shu, H., 2014. Performance evaluation of wavelet scattering network in image texture classification in various color spaces. URL: *https://arxiv.org/abs/1407.6423*, doi:*10.48550/ARXIV.1407.6423*.

[49] Yao, L., Bendich, P., 2020. Graph spectral embedding for parsimonious transmission of multivariate time series, in: 2020 IEEE Aerospace Conference, pp. 1–12. doi:*10.1109/AERO47225.2020.9172767*.

[50] Zhu, W., Zeng, N., Wang, N., 2010. Sensitivity, specificity, accuracy, associated confidence interval and roc analysis with practical sas ® implementations. NorthEast SAS users group, health care and life sciences URL: *https://lexjansen.com/nesug/nesug10/hl/hl07.pdf*.

[51] Zsido, A.N., Teleki, S.A., Csokasi, K., Rozsa, S., Bandi, S.A., 2020. Development of the short version of the spielberger state—trait anxiety inventory. Psychiatry Research 291, 113223. URL: *https://www.sciencedirect.com/science/article/pii/S0165178120300834*, doi:*https://doi.org/10.1016/j.psychres.2020.113223*.

Appendix

A Mental Stress Dataset Consent Form

All subjects included in the Mental Stress Dataset signed the following consent form, confirming their eligibility to be included in the study.

Department of Engineering Cybernetics

DATA ACQUISITION CONSENT FORM

You are being invited to participate in a research study, which the Norwegian Center for Research Data (NSD) has reviewed and approved for conduction by the investigators named here. This form is designed to provide you - as a human subject - with information about this study. The investigator or his/her representative will describe this study to you and answer any of your questions. You are entitled to a copy of this form. If you have any questions or complaints about the informed consent process of this research study or your rights as a subject, please contact the PI or Co-PI (marta.molinas@ntnu.no, +47 94287670, andres.f.soler.guevara@ntnu.no).

Project Title: FlexEEG in Mental Health

Principal Investigators: Marta Molinas

Co-investigator: Andres Soler & Mohit Kumar

Thank you for agreeing to participate in this research project. This study involves research aimed at detecting the presence of psychological stress in the human body based on the analysis of EEG and PCG signals. You will participate in two separate data collection sessions. The first session will take place in the exam period of nov-dec 2022, and the second will take place after the holidays, early 2023. Before each session we will ask you to answer a self-evaluation questionnaire called 'State-Trait Anxiety Inventory'. This questionnaire will be used to determine whether you are stressed or not. During both sessions, you will be recorded twice: one five-minute period with no stressor, and one five-minute period with an Arithmetic stressor. You will be asked to rate your stress level on a scale from 1-10 after each recording. The Arithmetic stressor consists of different arithmetic statements presented on a screen. Your task will be to calculate each task in your head and click "T" on the keyboard if the statement is True, and "F" if it is False. This task is supposed to induce stress so please keep this in mind. Each session will last about 30 minutes. 10 of these minutes are for recording of EEG and PCG signals using Mentalab EEG and EkoDuo stethoscope. We will clean the areas of the scalp where the electrodes are placed with isopropyl alcohol. Electrode cap gel will be applied to the areas, but it is easily washed out with water and shampoo.

Participation in this study will take approximately 60 minutes of your time. We warn that the set-up of the EEG cap can lead to some discomfort, and the tasks you are given will (hopefully) induce some stress response. Your participation in this study is completely voluntary. Should you decide to discontinue participation or decline to answer any specific part of the study, you may do so without penalty.

Your participation in this study may help you understand the manifestations of stress on EEG signals. We are not asking you to place your name anywhere on the experimental booklet, so your participation is anonymous. None of your answers can be directly traced back to you. Should you have any further questions, please feel free to contact the study's principal investigator or co-PI, Marta Molinas and Andres Soler at the Department of Engineering Cybernetics. Her office is at Elektro D+B2 room D244, her phone number is +47 94287670, and her e-mail address is marta.molinas@ntnu.no.

By signing below, I confirm that:

- I give my consent to participate in the research study entitled “FlexEEG in Mental Health”.
- I hereby confirm that I have read the above information and have been informed about the content and purpose of the research.
- I fully understand that I may withdraw from this research project at any time without prejudice or effect on my standing with NTNU.
- I also understand that I am free to ask questions about techniques or procedures that will be undertaken.
- I give my consent for the collection and use of all data of the research “EEG and PCG in mental health” for use in research and teaching purposes.
- I give my consent to use my data for scientific purposes, its documentation and publications (including any exhibitions and further publications)
- I hereby declare that I am currently not diagnosed by a with any heart disease, or neurological disease
- I am also not on any medications affecting heart rate and/or brain wave function
- I hereby declare that I am not officially diagnosed with any mental illness

Date and place: _____ and _____

Participant’s signature: _____

First and last name: _____

Date of Birth and current Age: _____and _____

I hereby certify that I have given an explanation to the above individual of the contemplated study and its risks and potential complications.

29/11/2022

_____ _____

Principal Investigator’s signature Date

B Arithmetic Test Equations with Answers

The following 29 arithmetic tasks shown in table 7.1 were included as questions of the arithmetic test given to subjects during all run 002 recordings. The questions appeared on the screen in a random order. FALSE and TRUE correct answers are denoted as 't' or 'f' in the table. The same questions are included in the *arithmetic.xlsx* file, which is accessed by the *arithmetic.py* program to display the questions on the screen. Both files are available in my GitHub repository for the thesis [43].

Table 7.1: Arithmetic test equation numbers, problems, and answers.

Equation Number	Equation	Correct Answer
1	5 + 1 = 4	f
2	54 / 9 = 6	t
3	625 / 25 = 25	t
4	96 - 27 = 59	f
5	5x5 - 4x6 = -1	f
6	2 + (2/2) + (2x2x2)/2 = 8	f
7	52 - 38 + 21 = 35	t
8	5x8 – 9x6 = -15	f
9	2x4x6 = 48	t
10	32/8 + 60/5 = 16	t
11	15 + 38 – 27 = 25	f
12	168 – 35 + 101 = 234	t
13	(64/8) x (49/7) = 56	t
14	7x7 - 7 + 7/7 = 44	f
15	2x2 – 5x5 + 9x9 =60	t
16	59 + 32 – 65 = 26	t
17	32/4 + 82/2 = 50	f
18	523 – 348 = 175	t
19	296/2 = 148	t
20	(100 + 50 + 25 + 15)/2 = 95	t
21	8x5 + 2x3 + 4x6 = 72	f
22	90 + 80 + 70 + 60 = 300	t
23	8x8 – 9x9 = -17	t
24	2x3x4x5 = 120	t
25	6x7-8x9 = -32	f
26	198 – 56 + 232 = 374	t
27	3x4 + 2x12 + 24/2 = 48	t
28	6x6 + 6/6 + 6 = 43	t
29	923 – 734 + 128 = 327	f

C Default Hyperparameters for XGBoost, KNN and SVM Classifiers.

XGBoost:

This is an excerpt of the full list of hyperparameters available at source: *https://xgboost.readthedocs.io/en/stable/parameter.html*.

- booster [default= gbtree]
- verbosity [default=1]
- eta [default=0.3, alias:learning_rate]
- gamma [default=0, alias:min_split_loss]
- max_depth [default=6]
- min_child_weight [default=1]
- max_delta_step [default=0]
- subsample [default=1]
- sampling_method [default= uniform]
- colsample_bytree,colsample_bylevel, colsample_bynode[default=1]
- lambda [default=1, alias: reg_lambda]
- alpha [default=0, alias: reg_alpha]
- tree_method string [default= auto]
- scale_pos_weight [default=1]
- refresh_leaf [default=1]
- process_type [default= default]
- grow_policy [default= depthwise]
- predictor [default= auto]

KNN:

source: *https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html*.

- n_neighbors [default=5]
- weights [default='uniform']
- algorithm [default='auto']
- leaf_size [default='30']
- p [default=2]

- metric [default='minkowski']
- metric_params [default=None]
- n_jobs [default=None]

SVM:

source: *https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html*.

- C [default=1.0]
- kernel [default='rbf']
- degree [default='3']
- gamma [default='scale']
- coef0 [default=0.0]
- shrinking [default=True]
- probability [default=False]
- tol [default=1e-3]
- cache_size [default=200]
- class_weight [default=None]
- verbose [default=False]
- max_iter [default=-1]
- decision_function_shape[default='ovr']
- break_ties [default=False]
- random_state [default=None]

D Hyperparameter Optimization Trials

The hyperparameter optimizations for KNN and XGB classifiers were done with the Hyperopt [8] Python package. Tables 7.2, 7.3, 7.4, and 7.5, show the tested parameters over the trials and mark the chosen hyperparameter combinations, respectively.

STAI-30-30

Table 7.2: KNN Classifier — 50 hyperparameter combinations with Hyperopt for the STAI-30-30 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue.

d	n_neighbors	loss	trial_number
2.0	90.0	-0.811765	0
2.0	67.0	-0.811765	1
1.0	92.0	-0.835294	2
2.0	77.0	-0.811765	3
2.0	46.0	-0.811765	4
2.0	65.0	-0.811765	5
1.0	96.0	-0.835294	6
2.0	20.0	-0.813725	7
2.0	25.0	-0.811765	8
2.0	26.0	-0.813725	9
1.0	60.0	-0.835294	10
2.0	50.0	-0.811765	11
2.0	51.0	-0.811765	12
1.0	21.0	-0.837255	13
2.0	59.0	-0.811765	14
1.0	86.0	-0.835294	15
2.0	37.0	-0.811765	16
2.0	72.0	-0.811765	17
1.0	28.0	-0.835294	18
2.0	13.0	-0.811765	19
1.0	2.0	-0.827451	20
1.0	7.0	-0.839216	21
1.0	2.0	-0.827451	22
1.0	12.0	-0.839216	23
1.0	8.0	-0.837255	24
1.0	36.0	-0.837255	25
1.0	12.0	-0.839216	26
1.0	16.0	-0.839216	27
1.0	37.0	-0.835294	28
1.0	11.0	-0.839216	29
1.0	19.0	-0.837255	30
1.0	33.0	-0.835294	31
1.0	44.0	-0.835294	32

1.0	5.0	-0.839216	33
1.0	5.0	-0.839216	34
1.0	82.0	-0.835294	35
1.0	100.0	-0.835294	36
1.0	3.0	-0.839216	37
1.0	1.0	-0.839216	38
1.0	30.0	-0.835294	39
1.0	43.0	-0.835294	40
1.0	24.0	-0.837255	41
1.0	14.0	-0.839216	42
2.0	55.0	-0.811765	43
1.0	18.0	-0.841176	44
1.0	69.0	-0.835294	45
2.0	22.0	-0.813725	46
2.0	41.0	-0.811765	47
1.0	18.0	-0.841176	48
1.0	47.0	-0.835294	49

Note: Hyperopt aims to find the lowest loss score. Here loss is computed as the negative average accuracy over 10-fold cross-validation.

Table 7.3: XGBoost Classifier — 100 hyperparameter combinations with Hyperopt for the STAI-30-30 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue.

colsample_bytree	eta	gamma	max_depth	min_child_weight	n_estimators	reg_alpha	reg_lambda	loss	trial_number
0.781638	0.617239	6.135062	3.0	6.0	59.0	22.0	0.569657	-0.843137	0
0.630199	0.062868	2.774001	8.0	5.0	235.0	13.0	1.094483	-0.847059	1
0.837926	0.049587	7.765092	12.0	3.0	222.0	76.0	0.242874	-0.841176	2
0.993143	0.456610	8.378135	11.0	5.0	296.0	2.0	1.426361	-0.849020	3
0.982229	0.341833	10.645124	16.0	5.0	268.0	85.0	1.081904	-0.841176	4
0.812367	0.194582	3.897675	19.0	0.0	97.0	83.0	1.010505	-0.841176	5
0.732696	0.469792	8.724274	17.0	0.0	236.0	13.0	0.533594	-0.845098	6
0.607073	0.133103	1.334795	16.0	5.0	183.0	39.0	0.300414	-0.845098	7
0.515743	0.718494	5.304107	16.0	5.0	292.0	2.0	0.245851	-0.849020	8
0.897411	0.172514	8.037672	10.0	4.0	201.0	32.0	1.009201	-0.847059	9
0.753725	0.708789	8.984697	7.0	5.0	263.0	64.0	1.579158	-0.837255	10
0.859137	0.789775	5.113515	14.0	1.0	229.0	108.0	1.817246	-0.839216	11
0.627672	0.973272	9.221524	6.0	2.0	60.0	93.0	0.021800	-0.841176	12
0.534709	0.528137	11.783395	4.0	3.0	192.0	91.0	0.304976	-0.839216	13
0.588930	0.126929	10.531475	15.0	0.0	225.0	52.0	0.964945	-0.843137	14
0.955710	0.346351	8.815700	20.0	0.0	250.0	95.0	0.806074	-0.841176	15
0.647725	0.448689	5.163587	18.0	3.0	114.0	90.0	0.514662	-0.841176	16
0.577071	0.153556	0.574121	8.0	5.0	299.0	5.0	1.348724	-0.849020	17
0.899274	0.722499	6.608939	4.0	5.0	267.0	32.0	1.678089	-0.837255	18
0.569328	0.876897	1.697650	9.0	0.0	27.0	88.0	0.056724	-0.841176	19
0.693083	0.285683	0.050748	12.0	6.0	299.0	2.0	1.275994	-0.847059	20
0.681527	0.559756	7.213953	10.0	4.0	139.0	13.0	1.345411	-0.841176	21
0.503238	0.239724	3.836966	6.0	4.0	299.0	52.0	1.995250	-0.843137	22
0.933681	0.391569	10.681179	13.0	6.0	279.0	1.0	1.408024	-0.849020	23
0.998797	0.401225	11.917540	13.0	6.0	163.0	23.0	1.564986	-0.845098	24

0.552249	0.587487	0.295925	10.0	4.0	297.0	43.0	1.785662	-0.847059	25
0.925969	0.408689	10.329937	14.0	6.0	272.0	22.0	1.990257	-0.849020	26
0.999072	0.659130	9.743258	11.0	2.0	203.0	22.0	1.871795	-0.847059	27
0.948977	0.313750	10.978535	13.0	6.0	281.0	64.0	0.789458	-0.847059	28
0.901607	0.501472	6.334740	14.0	6.0	163.0	30.0	1.964850	-0.843137	29
0.800630	0.602026	9.891603	11.0	6.0	246.0	9.0	1.510331	-0.849020	30
0.790359	0.858612	7.366212	11.0	4.0	248.0	7.0	1.179554	-0.849020	31
0.783977	0.994343	7.527154	8.0	2.0	212.0	11.0	1.227222	-0.850980	32
0.848138	0.945004	7.226395	3.0	2.0	210.0	42.0	1.187899	-0.837255	33
0.758523	0.042428	11.265224	8.0	1.0	175.0	17.0	0.822378	-0.843137	34
0.707505	0.076851	2.943843	6.0	1.0	137.0	7.0	0.664383	-0.843137	35
0.826467	0.230873	3.763729	8.0	2.0	71.0	70.0	1.679179	-0.841176	36
0.501542	0.875217	5.382963	20.0	3.0	214.0	27.0	0.375203	-0.845098	37
0.793035	0.892469	7.469298	5.0	3.0	144.0	39.0	1.097287	-0.843137	38
0.871330	0.806585	8.204910	9.0	4.0	254.0	17.0	1.217730	-0.845098	39
0.734029	0.138698	1.107753	7.0	2.0	93.0	102.0	0.907698	-0.843137	40
0.969451	0.415696	9.872418	18.0	1.0	184.0	47.0	0.125197	-0.843137	41
0.765016	0.661195	5.788269	13.0	1.0	281.0	1.0	1.458005	-0.845098	42
0.642674	0.004330	6.717841	9.0	2.0	237.0	75.0	1.077085	-0.839216	43
0.668467	0.758648	4.580828	7.0	3.0	127.0	13.0	1.627497	-0.847059	44
0.921105	0.992910	9.230914	17.0	1.0	219.0	62.0	0.634891	-0.841176	45
0.819307	0.620875	7.962491	12.0	2.0	195.0	36.0	1.471553	-0.843137	46
0.876645	0.352180	11.445516	5.0	3.0	234.0	57.0	1.403330	-0.841176	47
0.722573	0.540535	8.549781	15.0	5.0	286.0	25.0	0.926760	-0.843137	48
0.614800	0.466402	4.477875	9.0	5.0	101.0	18.0	1.740566	-0.839216	49
0.984981	0.937002	6.017381	4.0	4.0	264.0	9.0	1.871160	-0.847059	50
0.665872	0.656332	2.090161	10.0	0.0	153.0	110.0	1.304981	-0.841176	51
0.772051	0.213546	10.329891	15.0	1.0	174.0	35.0	1.902110	-0.849020	52
0.850037	0.268702	9.073577	19.0	2.0	272.0	47.0	0.408792	-0.845098	53
0.525187	0.183116	3.024545	3.0	5.0	26.0	83.0	1.051429	-0.788235	54

0.808209	0.818804	9.584508	11.0	3.0	250.0	12.0	1.512565	-0.847059	55
0.781691	0.214949	6.758090	15.0	1.0	170.0	37.0	0.730557	-0.845098	56
0.927446	0.500529	10.909660	16.0	0.0	42.0	4.0	1.382742	-0.849020	57
0.834159	0.755576	7.779670	16.0	0.0	34.0	30.0	1.235534	-0.847059	58
0.738503	0.922369	8.509885	12.0	4.0	206.0	57.0	1.154547	-0.843137	59
0.874280	0.360051	11.993075	17.0	2.0	190.0	20.0	1.594439	-0.849020	60
0.801888	0.996500	10.113065	7.0	3.0	228.0	26.0	1.515682	-0.839216	61
0.710089	0.838949	11.519184	10.0	5.0	258.0	11.0	1.287434	-0.845098	62
0.596637	0.847602	5.709869	5.0	4.0	241.0	103.0	0.996264	-0.847059	63
0.741066	0.684872	4.455735	8.0	4.0	222.0	0.0	0.874325	-0.849020	64
0.913956	0.091011	10.967084	17.0	0.0	43.0	4.0	1.602491	-0.849020	65
0.880995	0.498639	11.682341	19.0	2.0	117.0	20.0	1.756712	-0.847059	66
0.766732	0.094936	9.355429	18.0	1.0	79.0	31.0	1.883975	-0.847059	67
0.548576	0.012895	1.242443	14.0	1.0	153.0	47.0	1.938222	-0.841176	68
0.948079	0.720825	3.550534	16.0	0.0	104.0	4.0	1.356390	-0.849020	69
0.622591	0.753700	0.729519	6.0	0.0	87.0	5.0	1.342775	-0.849020	70
0.618202	0.775235	0.556069	8.0	6.0	58.0	8.0	1.151537	-0.849020	71
0.686238	0.577661	6.347405	9.0	0.0	55.0	16.0	1.161071	-0.839216	72
0.861847	0.429522	7.094319	11.0	3.0	190.0	42.0	1.679585	-0.843137	73
0.567891	0.967953	4.862269	8.0	5.0	286.0	14.0	0.738339	-0.843137	74
0.774522	0.262985	11.983634	17.0	2.0	172.0	35.0	1.549581	-0.847059	75
0.647590	0.156515	2.415439	7.0	4.0	114.0	0.0	1.420052	-0.849020	76
0.641501	0.050641	1.731191	4.0	5.0	125.0	21.0	0.554049	-0.809804	77
0.699050	0.910591	10.411831	6.0	1.0	84.0	29.0	1.801985	-0.849020	78
0.701590	0.326713	8.763612	14.0	1.0	215.0	28.0	1.816794	-0.841176	79
0.895791	0.372849	7.323160	12.0	3.0	183.0	15.0	1.035325	-0.847059	80
0.965586	0.306464	8.201865	14.0	2.0	258.0	24.0	1.986469	-0.843137	81
0.749356	0.631494	4.115727	13.0	6.0	224.0	1.0	0.884866	-0.847059	82
0.906401	0.536245	3.337468	5.0	4.0	198.0	78.0	0.493280	-0.843137	83
0.714441	0.700988	4.196673	10.0	4.0	244.0	10.0	0.866226	-0.847059	84

0.661341	0.962715	5.603801	7.0	5.0	295.0	8.0	0.960934	-0.847059	85
0.887075	0.567929	0.760895	3.0	2.0	145.0	19.0	1.327279	-0.839216	86
0.937557	0.379757	11.144597	13.0	0.0	69.0	69.0	1.109506	-0.843137	87
0.673634	0.455299	10.742234	6.0	1.0	275.0	33.0	1.255252	-0.845098	88
0.830856	0.487262	6.924050	9.0	6.0	236.0	23.0	0.106354	-0.849020	89
0.817788	0.482007	7.662345	9.0	6.0	234.0	40.0	0.005889	-0.841176	90
0.846710	0.358946	9.547585	18.0	2.0	182.0	52.0	1.658685	-0.843137	91
0.990046	0.598764	4.999442	15.0	0.0	131.0	5.0	1.467703	-0.850980	92
0.984369	0.425942	4.896045	20.0	0.0	51.0	11.0	1.722621	-0.849020	93
0.985162	0.517927	5.015442	18.0	0.0	134.0	6.0	1.476018	-0.849020	94
0.968714	0.608117	6.321306	19.0	0.0	134.0	6.0	1.466882	-0.849020	95
0.964625	0.598209	6.442557	20.0	1.0	161.0	2.0	1.120088	-0.850980	96
0.952571	0.630295	6.106830	15.0	1.0	152.0	95.0	0.611714	-0.841176	97
0.860712	0.679724	5.256641	10.0	1.0	166.0	3.0	0.755146	-0.847059	98
0.942821	0.736877	5.505025	11.0	1.0	159.0	17.0	0.674754	-0.843137	99

Note: Hyperopt aims to find the lowest loss score. Here loss is computed as the negative average accuracy over 10-fold cross-validation.

STAI-35-20

Table 7.4: KNN Classifier — 50 hyperparameter combinations with Hyperopt for the STAI-35-20 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue.

d	n_neighbors	loss	trial_number
1.0	26.0	-0.903664	0
1.0	67.0	-0.902985	1
2.0	31.0	-0.900950	2
2.0	75.0	-0.899593	3
1.0	38.0	-0.903664	4
1.0	22.0	-0.902985	5
2.0	53.0	-0.900271	6
1.0	47.0	-0.902985	7
1.0	91.0	-0.902307	8
2.0	88.0	-0.899593	9
1.0	85.0	-0.902307	10
1.0	88.0	-0.902985	11
2.0	93.0	-0.899593	12
2.0	28.0	-0.902307	13
2.0	7.0	-0.902307	14
2.0	8.0	-0.905699	15
2.0	47.0	-0.900271	16
2.0	6.0	-0.906377	17
1.0	33.0	-0.902985	18
2.0	24.0	-0.902985	19
2.0	2.0	-0.907056	20
2.0	1.0	-0.898915	21
2.0	14.0	-0.904342	22
2.0	15.0	-0.901628	23
2.0	7.0	-0.902307	24
2.0	2.0	-0.907056	25
2.0	16.0	-0.902985	26
2.0	61.0	-0.899593	27
2.0	2.0	-0.907056	28
2.0	19.0	-0.901628	29
2.0	36.0	-0.901628	30
2.0	100.0	-0.899593	31
2.0	40.0	-0.901628	32
1.0	69.0	-0.902985	33
2.0	79.0	-0.899593	34
2.0	57.0	-0.900271	35
1.0	1.0	-0.900271	36
2.0	41.0	-0.900271	37
1.0	12.0	-0.905699	38

2.0	29.0	-0.900950	39
1.0	47.0	-0.902985	40
2.0	23.0	-0.901628	41
2.0	21.0	-0.901628	42
1.0	67.0	-0.902985	43
2.0	10.0	-0.905699	44
2.0	33.0	-0.900950	45
1.0	18.0	-0.905020	46
2.0	4.0	-0.907734	47
2.0	55.0	-0.900271	48
2.0	99.0	-0.898915	49

Note: Hyperopt aims to find the lowest loss score. Here loss is computed as the negative average accuracy over 10-fold cross-validation.

Table 7.5: XGBoost Classifier — 100 hyperparameter combinations with Hyperopt for the STAI-35-20 dataset split. Chosen Hyperparameter combination with loss and trial number is marked in blue.

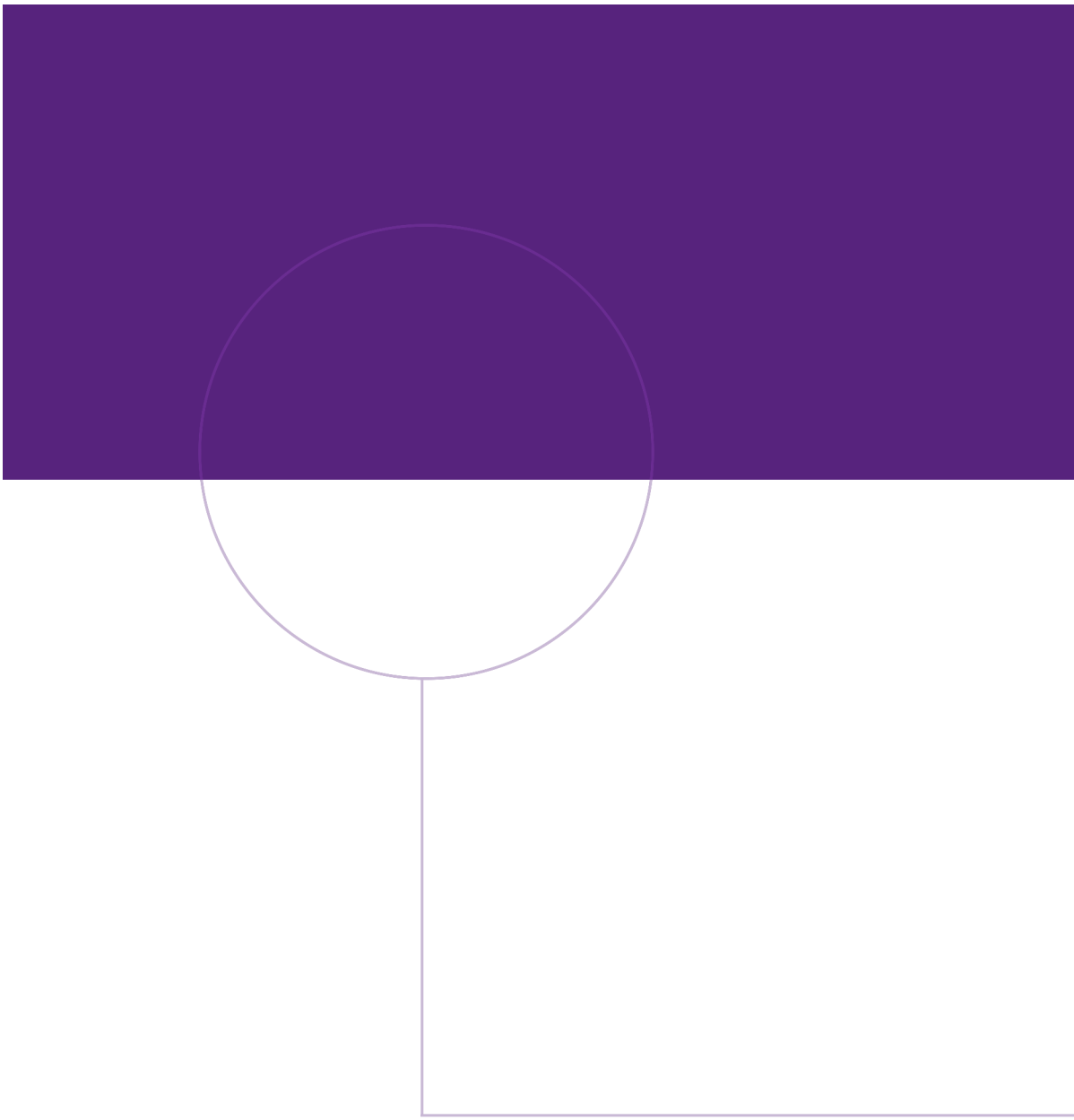
colsample_bytree	eta	gamma	max_depth	min_child_weight	n_estimators	reg_alpha	reg_lambda	loss	trial_number
0.526860	0.150781	8.795189	14.0	5.0	105.0	50.0	0.906121	-0.900950	0
0.695323	0.256928	7.400285	15.0	1.0	267.0	86.0	0.066324	-0.900271	1
0.705215	0.045989	4.989100	17.0	3.0	142.0	30.0	0.235608	-0.892809	2
0.912108	0.272791	5.972761	6.0	1.0	231.0	69.0	1.404551	-0.903664	3
0.650571	0.914347	1.328822	11.0	1.0	277.0	52.0	0.886490	-0.899593	4
0.753742	0.979531	4.148882	5.0	2.0	44.0	9.0	1.088445	-0.909091	5
0.874632	0.734150	1.648698	14.0	1.0	171.0	69.0	1.099255	-0.928765	6
0.828047	0.875197	3.535114	10.0	4.0	147.0	41.0	1.684680	-0.915875	7
0.978990	0.569390	9.011183	16.0	1.0	47.0	94.0	0.497325	-0.907056	8
0.718003	0.200596	5.083982	11.0	0.0	229.0	26.0	0.206981	-0.914518	9
0.860187	0.134671	10.967448	5.0	4.0	168.0	38.0	1.975253	-0.900950	10
0.610771	0.419678	3.472728	13.0	5.0	152.0	69.0	0.492498	-0.904342	11
0.880554	0.841777	1.650286	8.0	5.0	117.0	45.0	0.452994	-0.930122	12
0.657411	0.475835	0.819087	18.0	1.0	284.0	30.0	0.117979	-0.906377	13
0.638355	0.916115	7.812887	17.0	4.0	149.0	48.0	0.482901	-0.896879	14
0.689409	0.733186	3.774938	10.0	5.0	157.0	7.0	1.548429	-0.904342	15
0.554766	0.338789	1.342723	15.0	5.0	153.0	78.0	1.861290	-0.911805	16
0.594463	0.897771	5.739173	20.0	0.0	243.0	73.0	1.699918	-0.896201	17
0.744114	0.430455	8.003154	8.0	4.0	255.0	11.0	0.367516	-0.925373	18
0.955728	0.078247	9.059471	15.0	2.0	220.0	26.0	1.194191	-0.915197	19
0.821093	0.705984	2.228898	3.0	6.0	89.0	106.0	0.743626	-0.907734	20
0.903986	0.738911	0.013517	7.0	6.0	197.0	61.0	1.266926	-0.932836	21
0.932835	0.606012	0.021586	8.0	6.0	199.0	61.0	1.302207	-0.937585	22
0.939428	0.600185	0.099197	8.0	6.0	204.0	57.0	1.313712	-0.926730	23
0.988274	0.624332	0.281180	3.0	6.0	190.0	59.0	1.398737	-0.922659	24

	0.782081	0.788943	2.668178	6.0	6.0	206.0	104.0	0.694275	-0.911126	25
	0.912659	0.546965	0.333944	9.0	3.0	185.0	83.0	1.267098	-0.921303	26
	0.992460	0.627905	11.075593	7.0	6.0	296.0	63.0	1.551810	-0.921303	27
	0.811918	0.667634	2.943957	4.0	5.0	119.0	94.0	0.906435	-0.918589	28
	0.930319	0.993850	11.960375	13.0	6.0	97.0	63.0	0.969036	-0.912483	29
	0.844398	0.785379	0.017014	12.0	4.0	261.0	91.0	0.768092	-0.909091	30
	0.507393	0.516300	4.456380	9.0	3.0	69.0	80.0	1.530615	-0.903664	31
	0.959862	0.386207	6.639317	6.0	6.0	127.0	18.0	1.736578	-0.907734	32
	0.908175	0.813023	2.406564	7.0	2.0	186.0	35.0	1.119943	-0.937585	33
	0.889251	0.306713	2.304581	5.0	2.0	218.0	1.0	1.104652	-0.912483	34
	0.770991	0.979012	0.870760	10.0	2.0	179.0	35.0	1.441486	-0.898236	35
	0.806938	0.672502	5.021727	4.0	2.0	133.0	17.0	1.060349	-0.923338	36
	0.858025	0.829672	2.005672	7.0	3.0	81.0	52.0	0.631727	-0.943012	37
	0.855106	0.947858	6.855381	4.0	3.0	28.0	49.0	0.602245	-0.926730	38
	0.795554	0.827106	4.216955	7.0	3.0	49.0	34.0	0.850553	-0.934871	39
	0.737262	0.865891	1.865127	12.0	2.0	82.0	43.0	0.304865	-0.901628	40
	0.840401	0.996875	3.031135	9.0	1.0	61.0	52.0	0.009401	-0.928087	41
	0.971139	0.509942	9.731051	11.0	0.0	22.0	100.0	0.607213	-0.919267	42
	0.873197	0.779778	5.513401	6.0	1.0	168.0	27.0	0.631348	-0.911805	43
	0.692439	0.230456	6.365093	5.0	3.0	102.0	39.0	0.197755	-0.903664	44
	0.769157	0.936475	4.612588	13.0	4.0	36.0	21.0	0.822010	-0.939620	45
	0.668728	0.930271	4.545544	18.0	4.0	35.0	17.0	0.368523	-0.907056	46
	0.716001	0.012108	3.545270	14.0	4.0	75.0	21.0	0.839929	-0.890773	47
	0.759357	0.888867	7.288705	20.0	4.0	59.0	3.0	0.983566	-0.917910	48
	0.585387	0.952171	0.921533	16.0	4.0	38.0	73.0	0.151699	-0.904342	49
	0.729957	0.861853	5.515433	13.0	3.0	111.0	13.0	0.555566	-0.916554	50
	0.630985	0.460862	8.181674	16.0	5.0	91.0	53.0	0.423306	-0.900950	51
	0.670920	0.743658	3.861144	19.0	3.0	138.0	6.0	0.276937	-0.911126	52
	0.557112	0.670577	3.256460	10.0	5.0	20.0	46.0	0.780625	-0.900950	53
	0.785129	0.696169	1.852506	11.0	4.0	57.0	31.0	0.691725	-0.921981	54

0.759405	0.547563	4.687492	15.0	2.0	73.0	72.0	1.989008	-0.907056	55
0.831148	0.171695	6.001886	14.0	5.0	123.0	23.0	0.928807	-0.910448	56
0.702607	0.371663	1.464178	17.0	3.0	48.0	68.0	1.200690	-0.902985	57
0.888375	0.912712	8.599692	12.0	4.0	107.0	56.0	0.522428	-0.906377	58
0.619704	0.967889	9.309902	10.0	5.0	31.0	41.0	0.067507	-0.902307	59
0.864127	0.586825	7.592924	9.0	4.0	144.0	66.0	1.057343	-0.917232	60
0.718995	0.828021	2.754266	13.0	3.0	86.0	89.0	0.425684	-0.902307	61
0.677670	0.101384	4.042349	11.0	0.0	41.0	10.0	0.828692	-0.892130	62
0.646109	0.719036	0.523918	18.0	2.0	20.0	14.0	0.690933	-0.905699	63
0.999231	0.762597	5.259388	15.0	1.0	164.0	78.0	0.336162	-0.922659	64
0.925164	0.633045	1.187596	8.0	5.0	246.0	110.0	1.873770	-0.924695	65
0.902881	0.833351	2.279693	7.0	2.0	273.0	33.0	1.143901	-0.926730	66
0.946271	0.812270	2.636113	5.0	1.0	176.0	38.0	1.367259	-0.936906	67
0.814001	0.893533	3.473987	7.0	2.0	213.0	29.0	1.190384	-0.909769	68
0.913815	0.806683	6.208945	4.0	3.0	115.0	22.0	1.020245	-0.921981	69
0.846724	0.854348	4.884860	10.0	3.0	156.0	45.0	1.463136	-0.912483	70
0.873879	0.931284	1.933658	3.0	2.0	186.0	49.0	0.910137	-0.907056	71
0.774402	0.766978	3.242657	8.0	1.0	225.0	37.0	0.803804	-0.922659	72
0.799515	0.641388	0.638513	6.0	6.0	205.0	83.0	1.618052	-0.932157	73
0.829511	0.561115	1.185315	12.0	2.0	235.0	25.0	0.714866	-0.918589	74
0.983173	0.307787	6.910842	9.0	5.0	66.0	60.0	1.862904	-0.914518	75
0.969056	0.999651	4.212467	13.0	3.0	132.0	41.0	1.141050	-0.904342	76
0.944181	0.479235	0.257179	5.0	4.0	98.0	65.0	1.284444	-0.917232	77
0.885878	0.691883	2.535622	6.0	4.0	80.0	52.0	0.561238	-0.926052	78
0.749043	0.965131	2.094192	9.0	0.0	192.0	6.0	1.349924	-0.903664	79
0.791006	0.795927	1.517666	14.0	3.0	53.0	19.0	0.630886	-0.909091	80
0.856417	0.878064	5.310743	12.0	1.0	296.0	29.0	1.026898	-0.920624	81
0.898535	0.753578	3.853386	3.0	2.0	150.0	14.0	0.887443	-0.906377	82
0.924123	0.912162	4.483201	8.0	3.0	286.0	32.0	0.961294	-0.928765	83
0.776214	0.735729	2.986400	14.0	2.0	174.0	58.0	1.217193	-0.910448	84

0.820149	0.438332	5.678913	11.0	4.0	27.0	36.0	1.783084	-0.930801	85
0.962370	0.609736	10.063953	8.0	6.0	256.0	76.0	1.492620	-0.913840	86
0.865882	0.397093	0.675044	7.0	5.0	195.0	94.0	0.239759	-0.933514	87
0.735863	0.935077	4.872624	17.0	1.0	181.0	47.0	0.485439	-0.915197	88
0.804190	0.526460	3.183221	10.0	3.0	140.0	0.0	1.116647	-0.932836	89
0.840794	0.717923	3.691866	4.0	4.0	64.0	27.0	0.863603	-0.902307	90
0.915186	0.648677	0.932474	5.0	2.0	162.0	55.0	0.731390	-0.934871	91
0.763320	0.841897	1.735581	11.0	3.0	126.0	42.0	1.603734	-0.903664	92
0.896762	0.998512	2.337004	16.0	4.0	92.0	8.0	0.659979	-0.902307	93
0.728947	0.688377	2.864441	6.0	2.0	235.0	3.0	0.948837	-0.933514	94
0.681066	0.576611	7.052811	9.0	4.0	43.0	24.0	0.387584	-0.903664	95
0.700191	0.776201	6.336557	15.0	3.0	105.0	44.0	1.247371	-0.895522	96
0.952603	0.881213	4.324337	13.0	1.0	212.0	19.0	0.568393	-0.905699	97
0.851553	0.947065	4.025491	4.0	3.0	77.0	51.0	0.784794	-0.915197	98
0.658127	0.980359	4.691975	12.0	2.0	244.0	34.0	1.059845	-0.908412	99

Note: Hyperopt aims to find the lowest loss score. Here loss is computed as the negative average accuracy over 10-fold cross-validation.



NTNU

Norwegian University of
Science and Technology