Master's thesis

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

Andrine Elsetrønning

Generalized workflow with uncertainty quantification for detecting abnormalities in lung sounds

Master's thesis in Cybernetics and Robotics
Supervisor: Adil Rasheed
Co-supervisor: Jon Bekker
June 2021

NTNU
Norwegian University of Science and Technology

Andrine Elsetrønning

Generalized workflow with uncertainty quantification for detecting abnormalities in lung sounds

Master's thesis in Cybernetics and Robotics
Supervisor: Adil Rasheed
Co-supervisor: Jon Bekker
June 2021

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

NTNU
Norwegian University of
Science and Technology

Abstract

Objective: The main objective for this master's thesis is to find a generalized approach for detecting abnormalities in a lung sound. This generalized workflow should be abstract enough to apply to other audio time-series data. Estimating the uncertainty in the final classification should be included in the workflow to meet the safety-critical model demand in the field of medical diagnosis.

Method: Tromsøundersøkelsen has gathered and labeled a substantial dataset of lung sounds, which we have been granted access to. Using this dataset a preprocessing procedure was found through an empirical investigation, where segmentation proved to be an important step. Furthermore, the state-of-the-art within time-series classification was identified, in terms of performance and scalability. Only the leading scalable algorithms were applied to the lung sound classification problem. Finally, a convolutional neural network with Monte Carlo dropout and a Deep Ensemble were employed to estimate uncertainty.

Results: Discriminating the anomalous lung sounds from normal lung sounds proved to be demanding, because of the noisy nature of the signal. This issue highlighted results in literature that achieved deceitfully high accuracy because of data leakage.
Utilizing simple statistics in the time domain and Mel-frequency domain, the proposed approach achieved on par performance results, compared to state-of-the-art time-series classification algorithms. The Deep Ensemble approach seems superior to the Monte Carlo dropout in estimating uncertainty, after conducting experiments with increasingly manipulated data.

Conclusion: The results suggest that there is still a long way to go before autonomous lung sound classification can be applied in a practical setting. Until classification methods achieve stronger results, the main focus should be toward utilizing models that "know what they do not know". Upcoming work ought to prioritize automatic segmentation of lung sounds, and design data acquisition methods that will simplify digital analysis.

Sammendrag

Mål: Hovedmålet med denne masteroppgaven er å finne en generaliserbar prosedyre for å detektere unormale lungelyder. Den foreslåtte prosedyren bør være generell nok til å kunne bli benyttet til andre lydbaserte klassifiseringsproblemer. Påliteligheten til den autonome klassifiseringen er spesielt viktig i medisinsk data, dermed vil en usikkerhetsanalyse være nødvendig.

Metode: Tromsøundersøkelsen, som er en av Norges største befolkningsundersøkelser, har samlet inn en rekke lungelyder, og angitt hvilke typer uregelmessigheter som finnes i disse lungelydene. Dette datasettet med lungelyder og markeringene; crackle, wheeze og normal, ble brukt for å utrede en forbehandlingsprosedyre. Eksperimenter viste et behov for segmentering av lungelyden. En rekke tidsserieanalyse-teknikker ble utredet, og sammenliknet basert på nøyaktighet og skalerbarhet. Kun de mest skalerbare algoritmene ble testet, ettersom lyddata ofte kan ha betraktelig størrelse. To metoder ble undersøkt for usikkerhetsanalyse; Convolutional Neural Network med Monte Carlo Dropout og Deep Ensemble.

Resultater: Å skille de atypiske lunge lydene fra de normale viste seg å være vanskeligere enn først antatt, grunnet det uregelmessige signalet. Dette problemet satte søkelyset på hvordan dagens forskning ofte oppgir upålitelige klassifiseringsresultater, på grunn av datalekkasje.
Ved å ekstrahere enkle statistiske egenskaper fra tidsdomenet og Mel-frekvensdomenet, og videre bruke dette til å kategorisere lungelyden, ble det oppnådd en nøyaktighet som er sammenlignbar med de fremste tidsserieanalyse-teknikkene.
Deep Ensemble viser tendenser til å gi bedre usikkerhets-estimater enn Monte Carlo Dropout.

Konklusjon: Resultatene foreslår at det enda er en stor jobb igjen før autonom lungelyd analyse kan bli brukt i praktiske situasjoner. Frem til en høyst nøyaktig metode er utviklet, vil usikkerhetsanalyse spille en viktig rolle. Fremtidig arbeid burde fokusere på automatisk segmentering av lungelyden, samt design av opptaksprosedyre.

Preface

This thesis is submitted for the degree of Master of Technology in Engineering Cybernetics, to the Department of Engineering Cybernetics at the Norwegian University of Science and Technology. In terms of learning, the goal of this thesis is to gain extensive knowledge in a specific field, and by doing so, one also acquires an understanding of running practical projects. This thesis was completed in the Spring of 2021, with background knowledge about lung sound classification from the specialization project, finished in the fall of 2020.

I would like to thank my supervisors Adil Rasheed and Jon Bekker, for their contributions of knowledge, guidance, and time. Through weekly meetings, my supervisors have given their inputs and motivated rigorous research. Being a doctor of medicine, Jon Bekker helped us cross the multidisciplinary boundary that often appears between engineering and medicine. His contributions facilitated a realistic perspective on the topic of ML in medicine. The extensive experience of Adil Rasheed has steered my attention into the right topics throughout the last year, giving pinpointers and input on which methods would be worthwhile.

Tromsøundersøkelsen, being Norway's most extensive and highly attended population-research initiative, has generously supplied us with lung sound recordings, along with corresponding labels.

The code for the work conducted was mainly implemented in *Python v3.7.9* [1]. To manage the audio data, the library *librosa* [2] was employed, while *sktime* [3] was used to explore time-series classification procedures. The code produced can be found at [4].

Contents

List of Figures

List of Tables

Nomenclature

$\mathbf{x}$	Time series
$\mathbf{z}$	Output of convolution operation
ω	Kernel
ρ	Padding
AI	Artificial Intelligence
ARC	Accuracy-Rejection Curve
b	Bias
$BOSS$	Bag Of SFA-Symbols
$Catch22$	CAnonical Time-series CHaracteristics
$cBOSS$	Contractable BOSS
CD	Critical Difference
CNN	Convolutional Neural Network
d	Dilation
DFT	Discrete Fourier Transform
DTW	Dynamic Time Warping
ECE	Expected Calibration Error
EE	Elastic Ensemble
$EEMD$	Ensemble Empirical Mode Decomposition
$HIVE-COTE$	Hierarchical Vote Collective of Transformation-based Ensembles

i	Index
l_ω	Kernel length
m	Time series length
MC	Monte Carlo
$MiniROCKET$	MINImally RandOm Convolutional KErnel Transform
ML	Machine Learning
n	Number of time series samples in a dataset
n_ω	Number of kernels
ppv	Proportion of Positive Values
$RISE$	Random Interval Spectral Ensemble
$ROCKET$	RandOM Convolutional KErnal Transform
S	Shapelet
SFA	Symbolic Fourier Approximation
ST	Shapelet Transform
TSC	Time Series Classification
TSF	Time Series Forest
w	Weight
$WEASEL$	Word ExtrAction for time SEries cLassification

Chapter 1

Introduction

Through a considerable exploration of the lung sound classification problem, an issue of data leakage is detected in scientific literature. Realistic data appears to be ill-performing as a result of the following: signal-to-noise ratio, and abnormality-to-segment ratio.

1.1 Motivation and Background

Rapid advancements in Machine Learning (ML) facilitates more complex decision-making in multiple fields. With minimal human interaction, machines can forecast the weather, control cars, and make medical diagnoses. We encounter several ML applications on a day-to-day basis. While checking Google Maps for traffic, unlocking our phones with face recognition, or asking Siri/Alexa to set a timer, we interact with complex ML technology. Clearly, big companies show little hesitation toward introducing their products to "smarter" algorithms, however not all fields have jumped on the artificial intelligence, AI, bandwagon. These revolutionary developments have opened the doors to a new age of digital medicine, yet the healthcare sector lags in putting the new tools to use [6]. A select few ML-based algorithms have made it into routine clinical care, such as ambulatory ECG monitoring, to detect arrhythmias [7], or ambulatory monitoring of blood glucose for diabetes patients [8]. Challenges that hinder the development of digital medicine tools involve the wide multidisciplinary nature of the field, international differences in standards, languages, expectations, and in general the regulatory environment.

To apply AI in medical settings, a layer of accountability has to be introduced. A critical model should be able to say "sorry, I do not know", which would be more socially acceptable, than making a wrong classification. When using ML to classify an observation, there is always a degree of uncertainty. This uncertainty could be due to an incorrect data-driven model or incorrect data. These two origins of uncertainty have different aids for solving. If an ML algorithm is to help a medical practitioner, then it would be useful if the algorithm can indicate when the recorded medical data is of insufficient quality. Furthermore, the algorithm could simply state that the user should re-record the data until the quality is approved. However, if the model is the origin

of uncertainty, then the algorithm needs to alert a doctor, to take further actions. It is clear that if one knows what kind of uncertainty is present in a prediction, this could lead to a greater acceptance of using AI in hospitals.

Monitoring respiratory behaviors can provide us with valuable information about lung health. Respiratory behaviors are characterized by the lung sound, which refers to the specific sound produced by air moving through the respiratory system. The stethoscope is the equipment of choice, in both cardiac and respiratory auscultation. Many have pinpointed the stethoscope as being an unreliable instrument, however, according to [9] this is not the case. [9] defends the stethoscope, and showcases its importance in respiratory analysis.

According to the World Health Organization, respiratory diseases is the third leading cause of death worldwide, as of 2019 [10]. New estimates, as of 2021, state that of the top five causes of death, three is due to chronic pulmonary disease, COVID-19, and lower respiratory infections respectively [11]. Unquestionably, methods for improving lung health, and treatment of respiratory conditions can help save lives. Today, respiratory health monitoring is limited to stethoscopes, which comes with a user dependency. Lack of experience, hearing loss, and the overall stress due to overfilled hospitals may result in practitioners making inconsistent decisions. Electronic auscultation can facilitate doctors in placing a diagnosis, as well as help manage the disease. With the ongoing COVID-19 pandemic, remote monitoring of the respiratory system could also be an important infection prevention tool. [12] discusses how a wireless stethoscope was used to diagnose COVID-19 pneumonia remotely, by detecting abnormalities in the lung sound. The study demonstrated how monitoring of the lung sound can reflect the disease status, thus guide the treatment. A cohort study of over 44,000 patients showed that illness related to COVID-19 becomes severe in 14 % of the cases, and critical in 5 % [13]. Of the critical cases, the fear is to develop acute respiratory distress syndrome. The timing of invasive mechanical ventilation plays an important part in the treatment, and can help improve the prognosis of severe COVID-19 cases [14]. Risk stratification encourages adequate care, moreover ensures the appropriate distribution of limited hospital resources.

In this thesis the discussion points presented will be grouped into the subject of automated analysis of lung sounds. Through a literature survey, the state-of-the-art and the knowledge gaps within will be specified.
Considering Table 1.1.1, there is a noticeable trend of firstly, applying different preprocessing techniques (noise reduction, down-sampling, and slicing) to the raw lung sound data, then de-

composing the data into sub-bands, which could carry more relevant information about the abnormalities, finally applying a collection of ML algorithms to make a classification. The approaches presented are very specific to lung sound classification, with little focus on *generalization*. The assumption is made that a solid preprocessing, feature extraction, and classification routine should be as good at adapting to new problems as the human ear. Furthermore, most papers read have not placed any work into *quantifying the uncertainty*. As stated above, in safety-critical tasks revolving around medicine, uncertainty estimates are indispensable. These statements will thus be the focus of this thesis.

In the research presented there seems to be a performance gap, depending on what dataset is employed. In [15], the reproducibility of research on digital medicine is emphasized. [15] states that challenges occur with fragile medical data, the heterogeneity of the diseased population as well as intentional- and unintentional biases. To make sure all results provided in this thesis are *reproducible*, a Github repository containing all produced code is made public.

Paper	Labels	Performance	Method
[16]	Crackle, Wheeze, None, Both	Accuracy = 49.86 %	Non-linear novel spectral feature extraction is utilized, along with a Support Vector Machine for classification.
[17]	Healthy, Pathological	Accuracy = 86.00 %	Compares the baseline method, which is spectral feature extraction with an SVM classifier, with Spectral Images fed into a CNN. The CNN approach achieved the best results.
[18]	Crackle, Wheeze, None, Both	Accuracy = 66.31 %. Patient specific classifier achieved accuracy = 71.81 %	Deep CNN-RNN model classifying lung sound based on Mel-Spechtrograms.
[19]	Crackle, Wheeze, Stridor, Squawk, Rhonchi, Normal	Accuracy = 94.82 %	A combination of time and frequency features are extracted from the Hilbert-Huang domain, furthermore a multilayer perceptron network is used for classification.
[20]	Normal, Abnormal	F-measure = 94.10 %	Several features were extracted from the spectral, fractal- and time- domain information. The main focus is dedicated to feature selection, comparing Random Subset Feature Selection to Sequential Forward Selection. An SVM, K-Nearest Neighbors and Naive Bayes were used for classification.
[21]	Crackle, Wheeze, None, Both	Sensitivity = 49.50, specificity = 39.370 %	Mel-Frequency Cepstal Coefficients are extracted, along with their first order derivatives, after preprocessing the lung sounds. Hidden Markov models in combination with Gaussian mixture models are used for classification.

Table 1.1.1: Literature survey on lung sound classification.

1.2 Research Objectives and research questions

1.2.1 Objectives

Primary Objective: Obtain a robust approach for classifying crackle, wheeze and normal lung sounds.
Secondary Objectives:

- Identify common issues and pitfalls in classifying lung sounds that can guide further research on the topic of lung sound classification.
- Quantify the model uncertainty, to make more robust and trustworthy classifications.

1.2.2 Research Questions

To the best of our knowledge, there has been little published work identifying the reason as to why there is a substantial performance gap in research when classifying lung sounds. To this end, the guiding questions governing the research can be stated as:

- Is it possible to find an *unbiased* procedure, that achieves sufficient classification results, that could be used in a *practical setting*?
- What *time series classification* technique delivers the best discriminating power between normal, crackle, and wheeze lung sounds?
- Could *uncertainty quantification* be used as a juncture between engineers and doctors, leading toward the usage of AI in clinical care?

1.3 Outline of Thesis

The thesis comprises of the following sections and content: Chapter 2 gives an introduction to the theoretic background on the topic of lung sound classification, as well as introducing the methods to be utilized in experiments; Chapter 3 dissects the employed dataset, as well as stating the proposed preprocessing procedure; while Chapter 4 presents and discusses the results obtained through experiments. Finally, the thesis is concluded in Chapter 5.

Chapter 2

Theory

This chapter discloses the theory behind approaches found in a literature survey. Preliminary, the topic of machine learning is introduced and narrowed down to fit the scope of this thesis. Afterward, state-of-the-art time series classification algorithms are presented. Towards the end of this chapter, methods for evaluating the performance of classification methods are presented, along with some uncertainty quantification theory. Finally, necessary knowledge about lung sounds and their common abnormalities is introduced.

2.1 Machine Learning

Machine Learning, ML, is a subfield of AI, that gives computer programs the ability to automatically learn and make decisions based on experience. With recent developments, such as cheap sensors and storage, a huge amount of data is easily accessible, spiking the need for new data-handling solutions. ML algorithms are derived from statistical principles, and they are currently being employed in an array of fields, such as; finance, forecasting, service personalization, speech recognition, and medical diagnosis. As the topic of ML is vast and diverse, only a select portion will be targeted in this thesis. It is assumed that the reader has some basic background knowledge about common ML algorithms, more in-depth theory can be found in the following literature [22, 23, 24].
ML can be applied to both regression and classification problems, where regression focuses on predicting a continuous output variable, while in classification the output is a categorical variable. In this thesis ML is narrowed down to time series classification, only considering supervised classification approaches.

2.2 Time series classification

Time series classification, TSC, has gained interest in recent years, with the buildup of cheap sensory data, together with the progressive field of ML. A site has been developed to provide a comprehensive repository for research on TSC. This website utilizes [25], which contains 128

TSC datasets. [26] presents the state-of-the-art approaches as of 2016. Some of these approaches are still relevant, however recent advancements in the field have yielded better performance in terms of accuracy and scalability. The most recent results, as of spring 2020, are displayed in A.2.1. The time used to train the different algorithms presented in Figure A.2.1, is displayed as a critical difference plot in Figure A.2.2. It is clear to see, in Figures A.2.1 and A.2.2, that some of the most promising algorithms in terms of accuracy are not sufficient in terms of time complexity. To get an overview of TSC, both historically and the current state-of-the-art, several common TSC approaches will be defined below.

2.2.1 DTW with K-NN

The benchmark procedure for many years has been using Dynamic Time Warping, DTW, as the distance measure in the nearest neighbor algorithm. Dynamic Time Warping measures similarity between temporal sequences of varying speed. DTW calculates the optimal match between two given sequences when a set of rules are applied. Given two time series, $\mathbf{x}_1$ and $\mathbf{x}_2$, the following rules holds:

- Every index from time-series $\mathbf{x}_1$ must be matched with at least one of the indices in $\mathbf{x}_2$
- The first index of $\mathbf{x}_1$ must be matched with the first index of $\mathbf{x}_2$, however, this does not have to be the only match. *This also holds for the last index.*
- Mappings from indices in $\mathbf{x}_1$, to indices in $\mathbf{x}_2$, must be monotonically increasing, and vice versa. This means that if there exists two indices, $j > i$ in $\mathbf{x}_1$, then there must not exist two indices $l > k$ in $\mathbf{x}_2$, such that i is matched with index l , and j is matched with k.

The optimal match of $\mathbf{x}_1$ and $\mathbf{x}_2$, is the one satisfying the rules while giving minimal cost. The cost is the sum of absolute differences, between the values of all matched pairs of indices.
As finding the optimal match can be a time-consuming procedure, a varying amount of restrictions are set. These restrictions might include setting a maximum allowable distance between indices or putting a weight penalty on warping distance, favoring reduced warping [27]. The time complexity of DTW with two time series of length m and l is $O(m \times l)$. For most TSC algorithms equal length is required, meaning $m = l$, resulting in a time complexity of $O(m^2)$.

Once the optimal path between a new instance and all training instances are found, a label is given to be that of the closest *k* neighbors.

2.2.2 Elastic Ensemble

Elastic Ensemble, EE, combines 11 nearest neighbors classifiers to make a TSC. EE uses different elastic distance measures in the time domain, along with the first-order derivatives [28]. There is a need for an elastic distance measure for TSC algorithms based on similarity. Noise or misalignment causing phase shifts are the main motivations for having an *elastic* distance measure. Research conducted in [28], stated that there were no statistically significant differences in accuracy, regarding different nearest-neighbor-based TSC algorithms. Moreover, the paper found that combining the 11 classifiers would lead to a significantly improved classification accuracy. The elastic distance measures investigated in the paper were: Euclidean distance (ED), dynamic time warping with full window (DTW), derivative DTW with full window (DDTW), DTW, and DDTW with window size set through cross-validation (DTWCV and DDTWCV), weighted DTW and DDTW (WDTW and WDDTW), Longest Common Sub-sequence (LCSS), Edit Distance with Real Penalty (ERP), Timewarp edit distance (TWED), and the Move-Split-Merge (MSM). To ensemble the classifiers, a weighting was given to each, based on the cross-validation accuracy. To counteract the effect of placing too much trust in the cross-validation accuracy, the weight was normalized over the number of transformations. Most of the classifiers in EE have a $O(m^2)$ time complexity, leading EE to inherit this.

2.2.3 BOSS

Bag Of SFA-Symbols, known as BOSS, is a dictionary-based method for doing time series analysis. The TSC algorithms mentioned above favor classification based on similarity in the time domain. Nevertheless, some tasks are better separated based on frequency. The BOSS algorithm utilizes sliding windows to detect patterns in data, these patterns are translated into discrete letters, forming words [29]. The words present in one time series form a histogram based on occurrence, furthermore, the distance between histograms is used to make a prediction. In further detail, the following steps are performed to obtain word histograms for a time series:

1. Set word length = l, and number of letters in the alphabet = c.
2. Discrete Fourier Transform (DFT) is performed on windows of the time series.
3. To obtain $\frac{l}{2}$ real Fourier coefficients, and $\frac{l}{2}$ imaginary Fourier coefficients, the DFT is low pass filtered to l.
4. The Fourier coefficients produced by each training sample are sorted, and placed into bins, via Multiple Coefficient Binning. The coefficients of the current window form a word, based on the bins.

5. The previous steps are performed for each window. Numerosity reduction is performed to avoid outweighing stable sections.
6. A histogram is created for each time series.

The bottleneck of the BOSS algorithm is the fitting procedure, which involves leave-one-out cross-validation with a 1-nearest-neighbor classifier, to find the best window length [29]. The resulting time complexity is $O(n^2 \times m^2)$.

2.2.4 Shapelet Transform

Shapelets are sub-sequences in a time series that allows for TSC based on local, phase-independant similarities in shape [30]. In the Shapelet Transform, ST, algorithm, the similarity between different shapelets and a time series is the discriminatory feature.
[30] proposes finding shapelets via an exhaustive search of every candidate between min and max. A candidate is here a subsequence of length l. The quality of the shapelet, S, is found by firstly calculating the minimum distance from S to all possible subsequences of a time series $\mathbf{x}$. The chosen S of the best quality is the one with the most information gain, between consecutive candidates of length i and $i+1$. For each time series in the training set the best S, and the quality is stored, and sorted. Clustering can be performed to remove similar shapelets. In the end, only the k best shapelets are selected. A feature vector is created from the distance of all k shapelets, to a time series $\mathbf{x}_i$ in the dataset. The resulting dataset of size $\mathbf{X} = (k, n)$, can be fed into any classifier, along with the labels $y = (1, n)$.
Searching the full space of all possible shapelets, with all possible lengths, has a time complexity of $O(n^2 \times m^4)$, where m is the time series length, and n is the number of time series in the dataset. This full-scale computation is infeasible for many classification problems, hence some speed-up techniques should be performed. These techniques include early abandonment of distance calculations/shapelets and precalculation of distance statistics, further details are found in [30].

2.2.5 HIVE-COTE

Hierarchical Vote Collective of Transformation-based Ensembles, known as HIVE-COTE, was the best performing TSC-algorithm for a multitude of years and has only recently been challenged [31]. The main limitation of HIVE-COTE is the extensive running time. HIVE-COTE is an improved version of Flat-COTE, introduced in [32]. The main difference between the improved approach and the old one is the voting scheme. Flat-COTE constitutes an ensemble voting of 35 classifiers in total, where 11 classifiers are whole-series classifiers, eight classifiers build upon

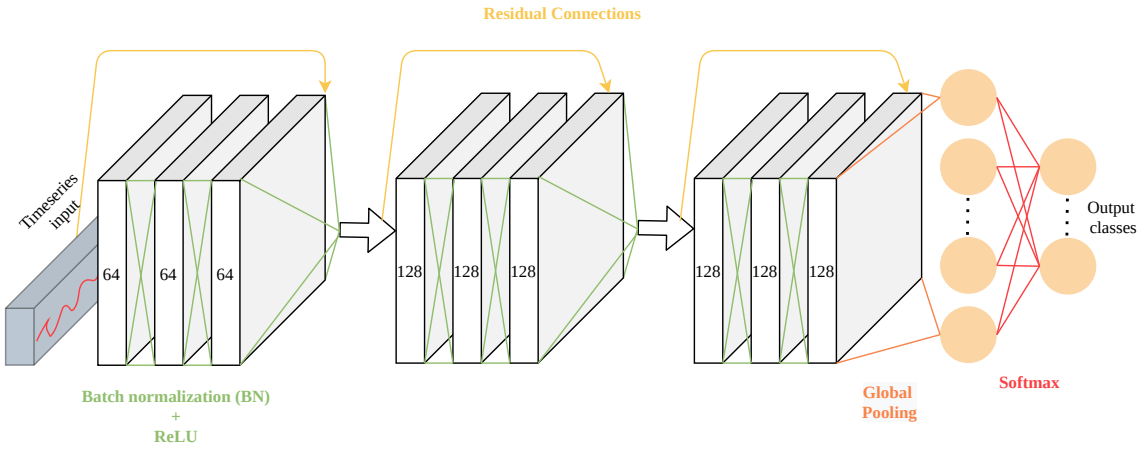


Figure 2.2.1: ResNet structure.

the shapelet transform, eight are based on auto-regressive features, and the remaining eight uses the power spectrum. It becomes apparent that when composing a weighted vote, based on the training set accuracy, the vote might become uneven. This uneven vote emerges when all classifiers are given similar weights, due to the uneven number of classifiers in each domain, a bias is introduced to the ensemble. To meet the difficulties of Flat-COTE, HIVE-COTE introduces modules. HIVE-COTE standardizes five classifier groupings: EE, ST, BOSS, TSF, and Random Interval Spectral Ensemble (RISE). Each module gives one vote, thus the collective probability is just the normalized weighted sum over modules.
As one of the modules in HIVE-COTE is ST, which has an established time complexity of $O(n^2 \times m^4)$, the time complexity for HIVE-COTE is bounded by this.

2.2.6 ResNet

The Residual Network, ResNet in short, is a standard baseline deep neural network that performs end-to-end TSC without requiring heavy preprocessing and feature-engineering [33]. An overview of the network architecture is illustrated in Figure 2.2.1. ResNet, like most convolutional neural networks, CNNs, tends to overfit, because of the large number of parameters.

2.2.7 InceptionTime

InceptionTime is a CNN structure for TSC, based on the Inception module [34]. The InceptionTime module is based on the Inception v4 architecture [35]. To learn both long and short patterns, a multitude of filters/kernels are applied. InceptionTime ensembles five separate CNNs to stabilize the output. An overview of the architecture and the module is displayed in Figures 2.2.2 and 2.2.3, respectively. From Figure 2.2.2 it is clear that the Inception network uses Inception

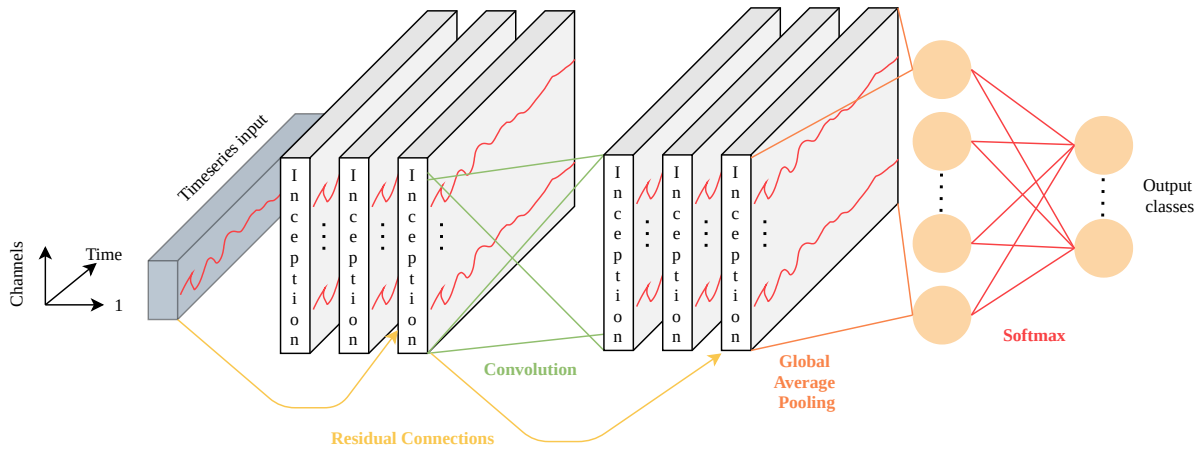


Figure 2.2.2: InceptionTime architecture, with the Inception module presented in 2.2.3

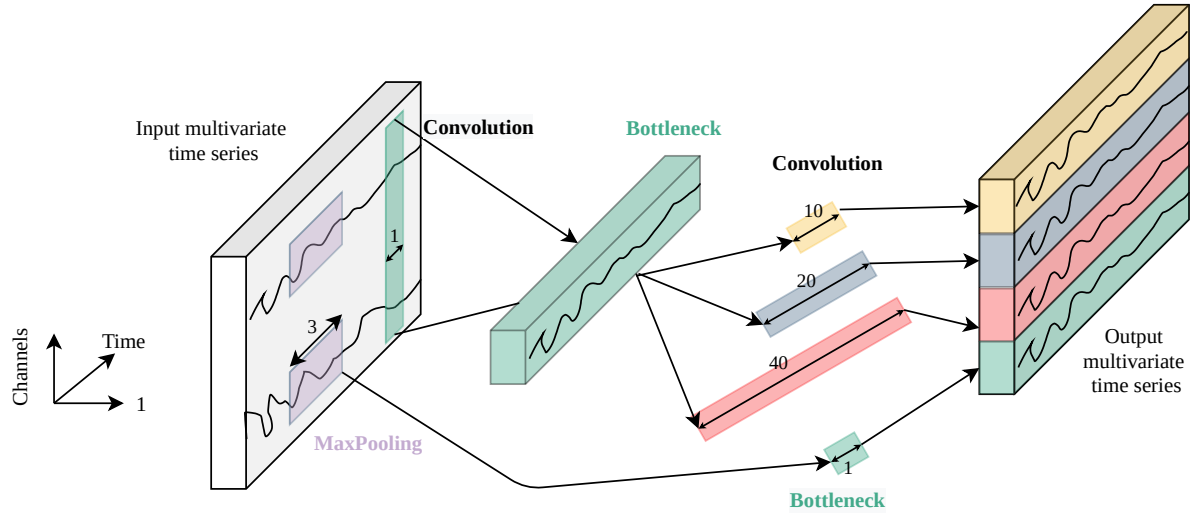


Figure 2.2.3: Inception module

modules, instead of pure convolutional layers. The yellow arrows mark residual connections, which are introduced to alleviate the vanishing gradient problem [36]. Figure 2.2.3 displays the usage of bottleneck layers, with a stride of one, to reduce the dimension of the input. A parallel MaxPooling operation is applied to make the model invariant to small perturbations, this is visualized as the purple windows in Figure 2.2.3. For more information about the choice of hyperparameters, see [34].

2.2.8 ROCKET

RandOM Convolutional KErnal Transform, is the state-of-the-art method for TSC, with the greatest achievement in terms of scalability. The ROCKET algorithm transforms the dataset using random convolution kernels [37]. These convolution kernels are the same as the ones found in a CNN. Because ROCKET only makes a single pass through a layer of kernels, without learning

any weights, a large number of kernels can be used. The default amount of kernels in ROCKET is 10 000, which results in 20 000 extracted features from each time series. The kernels, ω, are given a randomly initialized length (l_ω), weight (w), bias (b), dilation (d) and padding (ρ), while stride = 1. The result of applying the described kernel, to position i in the time series $\mathbf{x}$, is given by equation (2.2.1).

$$z_i = x_i * \omega = (\sum_{j=0}^{l_\omega - 1} x_{i+(j \times d)} \times \omega_j) + b \tag{2.2.1}$$

After convolving ω with a time series, two features are extracted: the *maximum* and the *proportion of positive values, ppv*. The maximum value is equal to a global max pool, while *ppv*, given in equation (2.2.2), indicates how to weigh the prevalence of a pattern, captured by the kernel.

$$ppv(\mathbf{z}) = \frac{1}{n} \sum_{i=0}^{n-1} [z_i > 0] \tag{2.2.2}$$

In equation (2.2.2), $\mathbf{z}$ is the output of the convolution. After applying the ROCKET transformation, any classifier can be employed. In [37] a *ridge regression classifier* is proposed for smaller datasets (n < 20 000). For large datasets (n $\gg$ 20 000), *logistic regression* with stochastic gradient descent is suggested. The randomness of the ROCKET classifier allows for capturing a wide range of information.

MiniROCKET

Even though ROCKET already is the fastest state-of-the-art TSC algorithm, a new implementation of ROCKET, MiniROCKET, had yielded similar accuracy, while being 75 times faster than the original ROCKET algorithm, on larger datasets [38]. The difference between ROCKET and MiniROCKET lies in the randomness of the kernel hyperparameters, which are compared in Table 2.2.1. In addition to the smaller hyperparameter grid, MiniROCKET only extracts *ppv* from each convolution output, hence only 10 000 features are obtained for each time series.

2.2.9 Catch22

CAnonical Time-series CHaracteristics, captures the dynamical properties of a time series concisely [5]. An excessive feature set can be found via the framework proposed in [39]. This framework is limited to *Matlab* users and consists of 7658 features, that can be extracted from a time series. [5] proposes a procedure for finding the 22 most promising features, via an extensive filtering process, comparing the feature performance across 93 TSC problems. The procedure

Table 2.2.1: Comparing hyperparameter grid for ROCKET and MiniROCKET.

Hyperparameter	ROCKET	MiniROCKET
l_ω	{7 , 9, 11}	9
w	$\mathcal{N}(0,1)$	{-1, 2}
b	$\mathcal{U}(-1,1)$	sampled from convolution output
d	random	fixed, default = 32
ρ	random	fixed

for selecting the overall most interesting features begins with removing features sensitive to normalization, as well as features that tend to give *NaN* values. After pre-filtering there were 4791 candidate features left.

The following stages are performed to compress the feature-set; statistical filtering, performance filtering, and redundancy minimization. Said stages are illustrated in Figure 2.2.4.

Statistical filtering finds significant features. In this section of the feature selection process, all labels are shuffled, to achieve a random guess accuracy Gaussian curve. An estimate of the null accuracy distribution curve is obtained for all feature-task combinations, via 1000 samples of random guessing. For each feature-task combination, the p-value is found through hypothesis testing with the estimated curve, as well as the true accuracy. Using the Fisher method and Holm-Bonferroni, a combined p-value across all 93 tasks is found. The statistical filtering is concluded by selecting the significant features ($p < 0.05$) , consequently 4646 candidate features remain.

Performance filtering consists of determining the combined mean accuracy across all tasks for one feature and sorting these accuracies to a be a Gaussian curve. A threshold for keeping features is selected to be one standard deviation above the mean, leaving 710 relevant features.

Finally, **redundancy minimization** is performed by clustering the Pearson correlation distance of a features accuracy vector. Hierarchical complete-linkage clustering with a threshold of 0.2 results in 22 clusters. One feature is selected from all clusters, which is either the best performing feature or the most interpretable feature in the cluster.

The remaining 22 features form a feature set that gives state-of-the-art performance while being faster than rivaling algorithms.

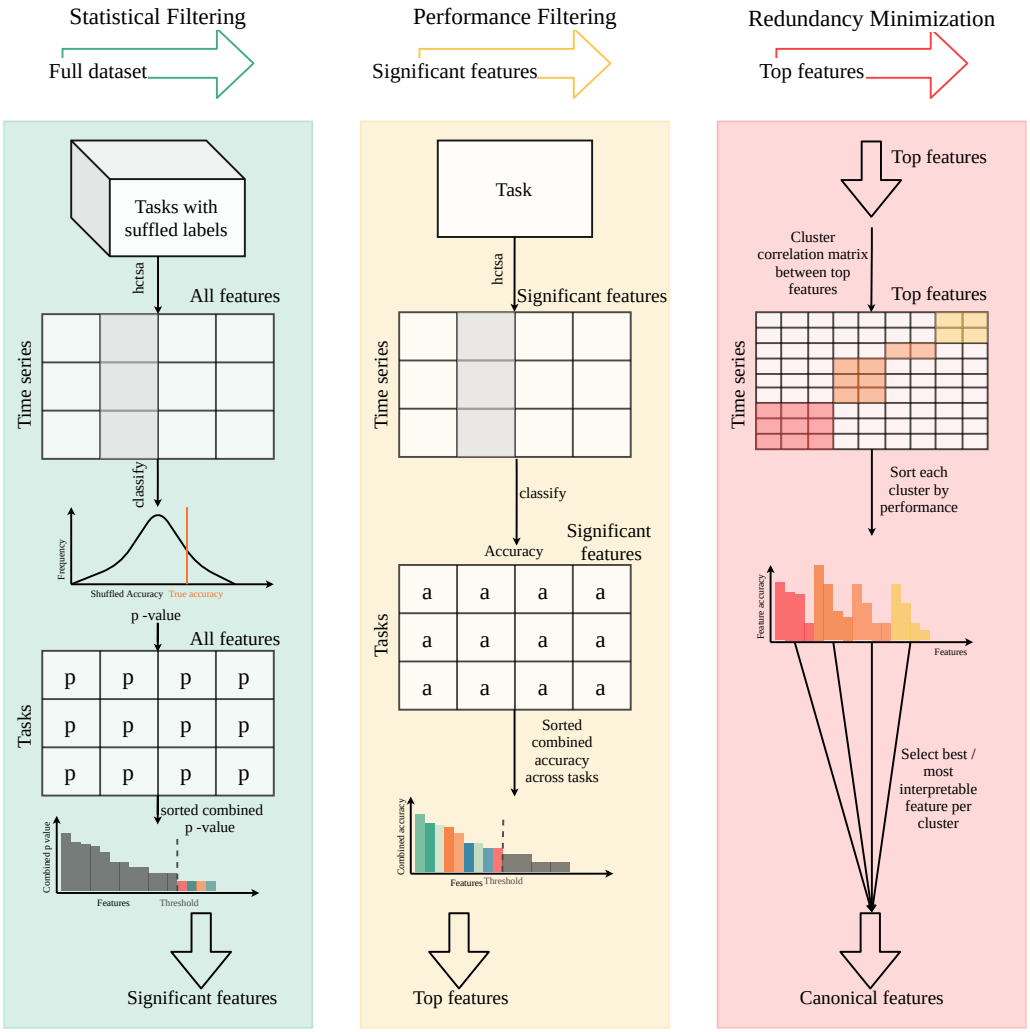


Figure 2.2.4: The three stages of selecting features; *Statistical prefiltering:* The features which give similar performance to random number generators are identified. *Performance filtering:* The top features according to the combined accuracy across all datasets is determined. *Redundancy minimization:* Clustering the top features with similar performance across tasks, and selecting one best performing feature from each cluster [5].

2.3 Evaluation of results

To compare and discuss the results that will be presented later on in this thesis, some evaluation methods will be established. In this section, various methods employed to compare different algorithms will be discussed.

2.3.1 Critical Difference

When the goal is to find a generalized procedure for classification, some basis for comparison needs to be formed. [40] proposes a critical difference, CD, plot as a method of comparison. CD-plots gathers the average ranking for multiple classifiers, over multiple data-sets, and illustrates

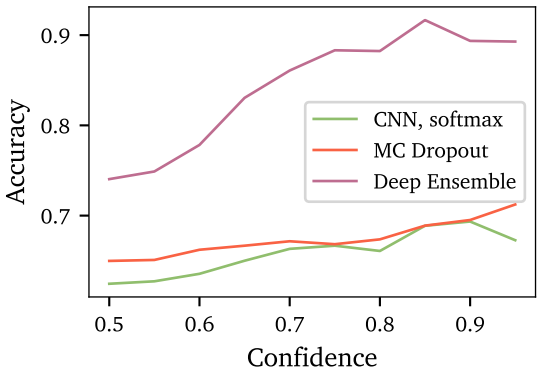


Figure 2.3.1: Accuracy-rejection/confidence curve, to illustrate the effect abstaining from classification has on the classification accuracy. This curve gives an indication of the classifiers knowledge about uncertainty. Classification is here performed between crackle and wheeze samples. The classifiers used will be discussed further in section 4.3.

this along with some significance tests. An example of a CD plot is displayed in Figure A.2.1. The solid horizontal lines mark cliques. These cliques are formed from statistical significance testing. If there is no solid horizontal line connecting two classifiers, then the classifiers perform significantly different. Common post-hoc statistical significance tests used for CD plots are the Nemenyi post-hoc test, Bonferroni-Dunn test or the Wilcoxon-Holm post-hoc test. See [41] for further explanation.

2.3.2 Accuracy-rejection curve

The accuracy-rejection curve, ARC, can give some indication of whether or not the classifier knows what it does not know [42]. The idea behind an ARC is that if the uncertainty calculated by the classifier is reliable, then this uncertainty should correlate with the probability of making a correct classification. If the classifier is allowed to abstain from classifying all samples that fall into a region of uncertainty, then the classification accuracy should improve on the remaining samples. Specifically, ARCs plots the accuracy of a classifier as a function of its rejection/confidence rate. An example an ARC is displayed in Figure 2.3.1.

2.3.3 Evaluation of uncertainty estimation

The interest in uncertainty quantification is in this thesis related to the real-world example of lung sound classification. Real-world settings often include a distribution shift from the training dataset, due to sample bias and non-stationarity [43]. [43] compares several state-of-the-art methods for model uncertainty quantification, when introduced to out-of-distribution samples and distribution shifts. The paper found that deep ensembles performed the best across most

metrics. Seeing as distribution shifts is a likely source of uncertainty in a practical lung sound classification setting, this will be explored in a later section. The metrics introduced in the discussed paper, Brier Score and Expected Calibration Error, will be utilized to measure the quality of the model uncertainty estimation. The Brier Score is a strictly proper scoring function that measures the accuracy of probabilistic predictions [44]. The score is the square error between the predicted probability vector, $p_\theta(y = c|\mathbf{x})$, and the one-hot encoding of the correct label, $\delta_{k=y}$, as stated in equation (2.3.1).

$$BS = C^{-1} \sum_{c=1}^{C} (\delta_{c=y} p_\theta(y = c|\mathbf{x}))^2 \tag{2.3.1}$$

In equation (2.3.1) C refers to the C number of classes and θ are the found parameters in a model.

Expected Calibration Error, ECE, measures the discrepancy between the accuracy and the confidence of a model. This metric is not a proper scoring rule, however it gives an intuitive understanding of the reliability of a model. To calculate ECE all predicted probabilities in the test set are put into a set amount of bins, usually $\Psi = 10$. For each bin the accuracy, $acc(B_\psi) = \frac{1}{|B_\psi|} \sum_{i \in B_\psi} (\hat{y}_i = y_i)$ and the confidence, $conf(B_\psi) = \frac{1}{|B_\psi|} \sum_{i \in B_\psi} p_\theta(\hat{y}_i|\mathbf{x}_i)$, is computed. ECE is defined in equation (2.3.2)

$$ECE = \sum_{\psi=1}^{\Psi} \frac{|B_\psi|}{n} |acc(B_\psi) - conf(B_\psi)| \tag{2.3.2}$$

In equation (2.3.2) B_ψ is the number of observations in bin ψ, while n is the number of samples in the test set.

2.4 Uncertainty Quantification

Most ML algorithms are black-box models, meaning that the processes taking place between input and output are opaque. To make informed decisions based on the numerical output of a model, we need to quantify the confidence that the model places in its predictions. Uncertainty quantification is an active field of research that focuses on capturing inaccuracies emerging from data-driven predictions. In the context of machine learning, a broad way to categorize sources of uncertainty would be the following sources [45]:

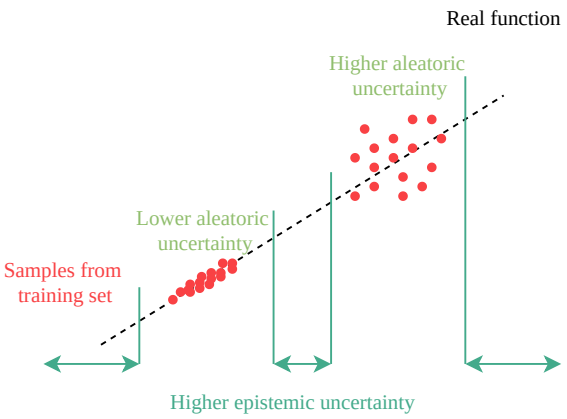


Figure 2.4.1: Aleatoric and epistemic uncertainty in a linear process.

- *Parameter Uncertainty* is connected to the input parameters of machine learning models. True model parameters are unknown, thus learning them is of interest.
- *Model Uncertainty* , also referred to as structural uncertainty or model inequality. The lack of knowledge about the underlying physics of a problem results in this type of uncertainty. Model uncertainty, along with parameter uncertainty, are both reducible when provided a sufficient amount of data.
- *Residual Variability* is concerned with the variation in predicted values after repeating a process. This variability may be due to the inherent stochasticity of a process, that the model could not eliminate.

2.4.1 Aleatoric vs Epistemic Uncertainty

Sometimes uncertainty is separated into the two following categories; aleatoric and epistemic. This categorization of uncertainty is commonly found within medical applications. **Aleatoric uncertainty** is uncertainty due to randomness in the data, which in practice means noise, or other inherently random effects that distort the data [46]. The uncertainty known as **Epistemic uncertainty**, refers to uncertainty caused by lack of knowledge about the best-suited predictor. This lack of knowledge can furthermore be divided into uncertainty over the model parameters, and uncertainty over model structure [47]. A visualization of aleatoric and epistemic, in a linear regression context, is displayed in Figure 2.4.1.

The two types of uncertainty are not necessarily strictly binary. One can assume that epistemic uncertainty is strongly dependant upon the amount of data introduced. The more data introduced, the less ignorant the classifier will be. However, when more data is added, the variability is likely to increase. Hence, roughly speaking, one would expect the epistemic uncertainty to

decrease, while the aleatoric uncertainty increases when feeding more data into a model.

Epistemic uncertainty is a reducible uncertainty, meaning that one can decrease the uncertainty by introducing more training data. Aleatoric uncertainty, on the other hand, can not be resolved by adding more observations. Distinguishing these two main types of uncertainty in a classification problem is of great interest, especially in safety-critical fields, such as medicine.

As stated one can not reduce aleatoric uncertainty by adding more data because this type of uncertainty comes from the fact that not all data is divisible. A solution in the case of aleatoric uncertainty is to add more features, thus uncovering information that could better discriminate between classes that overlap in the old feature space. Standard probabilistic classifiers give an estimate of aleatoric uncertainty, by modeling the probabilities $p(y|x)$. However, this probability is not always a solid measure of uncertainty, as it often fails to represent epistemic uncertainty. Points far away from the training data, which the model has no means for actually classifying correctly, tend to be given an unjustified high probability.

Estimating the epistemic uncertainty is not as straightforward as estimating the aleatoric uncertainty. Various research has been presented throughout the years, with quantification of this type of uncertainty in mind. [48] summarises the state-of-the-art in uncertainty quantification, dividing between aleatoric and epistemic uncertainty. Two key ideas discussed in this paper are probability estimation and generative models.

Probability estimation

Bayesian approaches to machine learning qualifies uncertainty in terms of probabilities. Given a hypothesis space, $\mathcal{H}$, which consists of possible probabilistic predictors, the Bayesian approach learns by replacing the prior distribution with the posterior distribution.

Several classification approaches deliver a probability for classes, given an observation. Well-established methods are; logistic and linear regression, various neural network approaches, and Bayesian approaches such as Bayesian Networks and Gaussian Processes [49, 50, 51, 52]. For some of the classifiers, the probability estimates are a vital part of the classification, while others, such as Naive Bayes or decision trees, rely on a post-processing step of Platt scaling or isotonic regression [53, 54]. By themselves, probabilistic predictors tend to mainly give a measure of aleatoric uncertainty, originating from the difficulty of representing lack of knowledge in

probability theory [55].

2.4.2 Generative models

Generative models learn the joint probability, $p(x, y)$, and predicts the most likely label using Bayes rules [56]. Consequently, the model includes the distribution of data along with the prediction, hence informing how likely an observation is. If the hypothesis space, $\mathcal{H}$, becomes larger, then the model uncertainty will diminish, however inevitably the approximation uncertainty will increase. In models with high flexibility, such as neural networks, the mapping from x to y has no explicit assumptions, thus finding a general model can be difficult. In the case of neural networks, one can expect the aleatoric uncertainty to appear in regions where the training data is overlapping, while epistemic uncertainty emerges in regions where the predictor has yet to encounter any data. With this information in mind, it becomes clear that generative models can indicate epistemic uncertainty.

The main approaches to estimate the density, $p(x)$, are; kernel density estimation [57] and gaussian mixture [58]. These density estimators have been combined with recent approaches, to detect outliers [59, 60, 61].

In [62] an intuitive understanding of uncertainty has been stated. The paper proposes dividing the classification problem into multiple binary classification problems, each equipped with a generative model. With this setup, one can distinguish between indifference and incomparability. Indifference will correspond to the conflict that arises when there is evidence in favor of more than one class, while incomparability reflects the level of ignorance. Ignorance appears when a new observation comes from a sparse region, where evidence lacks for all classes.

2.4.3 Model Uncertainty Estimation for Neural Networks

The topic of uncertainty quantification is vast and with many significant contributions. In this thesis, the problem of uncertainty quantification is narrowed down to epistemic uncertainty, with the main focus on model uncertainty estimation in neural networks. The two main approaches for estimating model uncertainty in neural networks are; Monte Carlo Dropout and Deep Ensembles [63, 64].

Monte Carlo Dropout

Dropout is a key concept used for regularization in deep neural networks [65]. The concept can simply be summarized as turning off some neurons at each training step. Each of the neurons in a neural network has a probability, p, of being ignored, this probability is a user-specified input

parameter, known as the *dropout-rate*. Since a neuron can be switched off at any moment, the weights are distributed more, leading to a model that generalizes better. Dropout is usually only performed at training time.
Monte Carlo, in mathematics, refers to the method of repeated random sampling leading to a distribution [66]. [63] combines the two discussed concepts to form an approach that can be interpreted as a Bayesian approximation of a Gaussian Process. By applying dropout at test time, one can think of all the generated networks as Monte Carlo samples from all possible models.

Deep Ensemble

In [64] an ensemble deep neural network approach is proposed as a simple and scaleable alternative to a Bayesian Neural Network [67]. The method combines a proper scoring rule, with ensembles and adversarial training, to produce smooth predictive estimates. Scoring rules assess the quality of predictive forecasts, by assigning a numerical score [68]. The goal of a forecaster is to maximize the scoring rule through optimization. Given a predictive distribution $p_\theta(y|\mathbf{x})$, along with an event $y|\mathbf{x} \approx q(y|\mathbf{x})$, where $q(y|\mathbf{x})$ represents the true distribution on dataset entries $y|\mathbf{x}$, the scoring rule is given as $S(p_\theta, q)$. If $S(q, q) \geq S(p_\theta, q)$ for all p_θ and q, then the scoring rule is considered proper. In [64], negative log-likelihood is deemed a suitable proper scoring rule, as defined in equation (2.4.1).

$$- log(p_\theta(y_i|x_i)) = log\frac{\sigma_\theta^2(\mathbf{x})}{2} + \frac{(y - \mu_\theta(\mathbf{x}))^2}{2\sigma_\theta^2(\mathbf{x})} + constant \tag{2.4.1}$$

Adversarial examples are added, using the fast gradient sign method [69]. Put simply, the fast gradient sign method modifies an input $\mathbf{x}$ slightly, along the direction which will increase the loss, $l(\theta, \mathbf{x}, y)$. Mathematically the new adversarial samples, $\mathbf{x}'$ are as expressed as in equation (2.4.2).

$$\mathbf{x}' = \mathbf{x} + \epsilon sign(\nabla_\mathbf{x} l(\theta, \mathbf{x}, y)) \tag{2.4.2}$$

In equation (2.4.2) ϵ is a small constant, tuned so that the new samples $(\mathbf{x}', y)$ are valid additional training samples. The addition of adversarial training creates smoother predictive distributions. The last contribution of the proposed approach was to take the average predictions of M neural-network-ensembles, after applying the proper scoring function and adversarial training to each neural network.

2.5 Lung Sound Classification

The lungs are cone-shaped organs, which main purpose is gas exchange. The goal of the gas exchange is to permit oxygen to move from the air into the systemic venous blood to sustain the metabolic needs of the cells. Furthermore, the lungs allow the waste product of cellular metabolism, which is carbon dioxide, to be released [70]. An example plot of a lung sound is displayed in Figure 2.5.1, here the inspiration and expiration phases are marked as blue and green respectively. [71] gives a guideline for common terminology in the field of computerized lung sound analysis. As the reach of this thesis is focused on abnormality detection in respiratory sounds, definitions of *crackles* and *wheezes* are supplied.

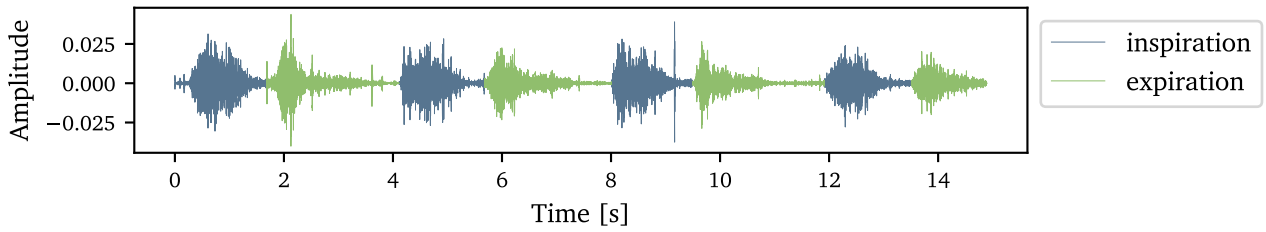


Figure 2.5.1: Inspiration and expiration phases of a normal lung sound marked with blue and green.

Crackles are short, discontinuous, explosive, and non-musical adventitious respiratory sounds that are related to the explosion of gas bubbles in pulmonary secretions or sudden opening of abnormally closed airways [72]. Crackles appear more often in inspiration than expiration and are in most cases best detected in the lowermost part of the lung [73]. Crackles are defined by intermittent discontinuous deflections that usually lasts less than 10 ms [74], furthermore, their appearance is often an early sign of a respiratory disease. The number of crackles per respiratory cycle and the timing within the cycle indicates both the severity and the type of disease [75]. Detection of crackles during a physical examination may lead to early diagnosis of important disorders such as pneumonia, heart failure, or pulmonary fibrosis. It is difficult to detect crackles in the power spectrum since the frequencies range from $200 - 2000Hz$, and the crackles take on a transient waveform [76]. Figure 2.5.2 illustrates crackle sounds appearing in the late inspiration phase of a lung sound.

Wheezes are music-like continuous adventitious sounds that usually last more than 100 ms, with a frequency greater than 100 Hz. Wheezes originate from obstructions in the airway, that vibrate as air passes through. In most cases, wheezes appear in the expiration phase of the respiratory cycle, superimposed on the normal lung sound. The characteristics of the wheeze, such as location, duration, and its relative placement to the breathing cycle can assist doctors in diagnosing

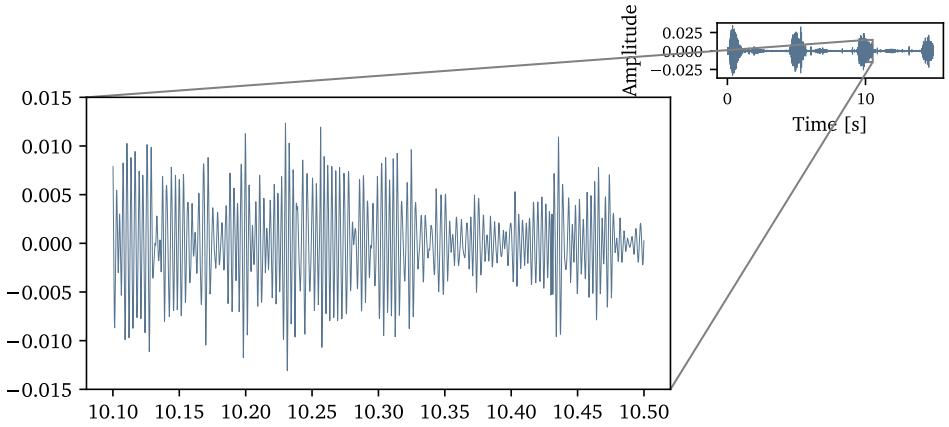


Figure 2.5.2: Zoomed in on crackle sounds appearing late in the inspiration phase of a lung sound.

and supervising diseases like chronic obstructive pulmonary disease, bronchiolitis, and asthma [77]. Figure 2.5.3 displays a wheeze appearing in the expiratory phase of the lung sound.

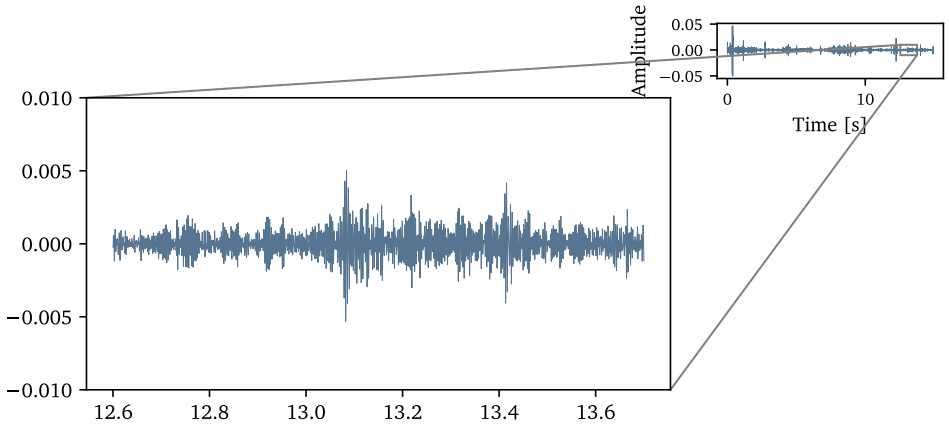


Figure 2.5.3: Zoomed in on wheeze appearing in the expiration phase of a lung sound.

Chapter 3

Method and set-up

To create a stable and robust generalized approach, several steps must be performed. The following chapter will go more in-depth on the data to be utilized, as well as explaining the preprocessing actions needed. A workflow will be proposed together with a set-up for evaluating the presented approach.

3.1 Data

The data to be utilized for the exploration of different time series classification approaches is taken from *Tromsøundersøkelsen*. The original goal of Tromsøundersøkelsen in 1976 was to deduce the cause for the high cardiovascular mortality in the region of Tromsø, Norway [78]. Since 1976 the study has been repeated multiple times, with the latest installation in 2016-2017. The lung sound of 6035 participants was recorded, where 45.2 % were male. Of the 6035 participants, all were older than 40 years, 60 % were aged 40-59 years, while 20 % were aged 60-84 years. To record the lung sound a microphone (MKE 2-EW) with a wireless system (EW 112-P G3-G), was placed in the tube of a Littmann Classic II stethoscope. To transmit the signal to the computer an external sound card was utilized. The recordings took place in a quiet environment. After initializing the recording at inspiration, the total duration was fixed to 15 seconds. Figure 3.1.1 illustrates the locations and order of the recordings. A sampling frequency of 44 100 Hz was utilized, and each lung sound was saved in a Wave (.wav) format. The dataset totals at 36 210 recordings, after gathering six lung sound recordings from all 6 035 participants. Two observers were sat down to classify the lung sounds. If the two observers disagreed, then a third observer would be brought in to discuss, and give a final classification. The lung sound was put into one of the following categories; *normal, wheeze during expiration, wheeze during inspiration, crackle during expiration, crackle during inspiration, other abnormal sound* or *unclassifiable*. A second round of classification was performed on all the recordings marked as containing an abnormality, here two observers marked the findings of the first round of classification as *certain, possible* or *absent*. Only the abnormalities marked as *certain* in the second round of observation were explored in this thesis.

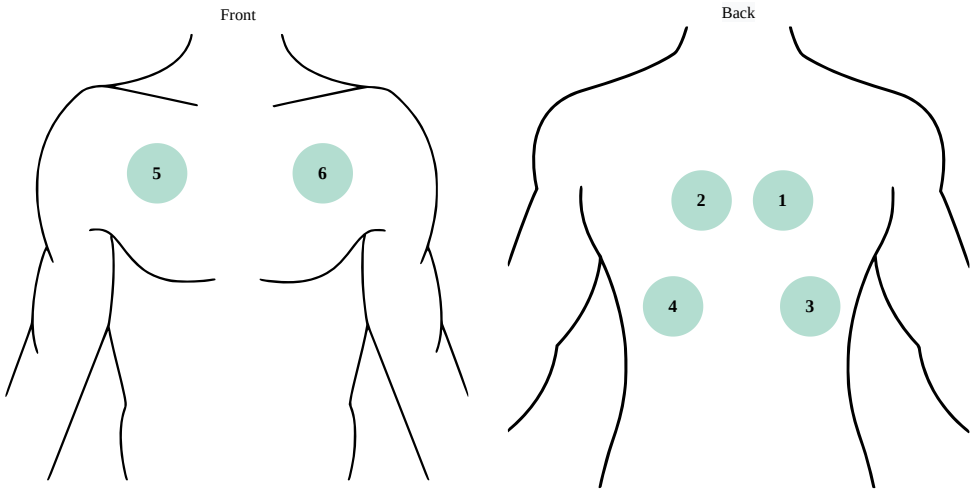


Figure 3.1.1: Recording locations, marked with order of collection.

For simplicity, only recordings where one type of abnormality is present are considered, to make sure that the TSC techniques summarize the characteristic behavior of each abnormality. Furthermore, slicing techniques will be utilized to narrow down the classification problem. Recordings with only one noted instance of an abnormality in the 15 s window, will be discarded.

3.1.1 Data exploration

To get a consensus of what kind of data one is dealing with, the dataset is explored further in this section. In total, after importing and processing the Wave-format files, there are 36 210 samples of lung sounds. Of these samples, there are 34 156 normal samples, with no marked abnormalities. Figure 3.1.2 displays the distribution of the labels in the dataset. The data is imbalanced, hence some measures need to be taken before introducing the dataset to a TSC algorithm.

To classify a lung sound, a medical practitioner needs to carefully listen to the audio to detect abnormalities. Looking at the visualization provided in Figure 3.1.3, there does not seem to be a clear visual difference between the various labels. This is to be expected, as both crackles and wheezes are short, compared to the total length of the recording. From the theory in section 2.5, together with Figure 3.1.3, it is clear that most crackles appear in the inspiration phase, while most wheezes appear in the expiration phase. As the scope of this thesis is to classify between crackle, wheeze and normal, the labels inspiration- and expiration-wheeze are concatenated, the same follows for the inspiration- and expiration-crackles.

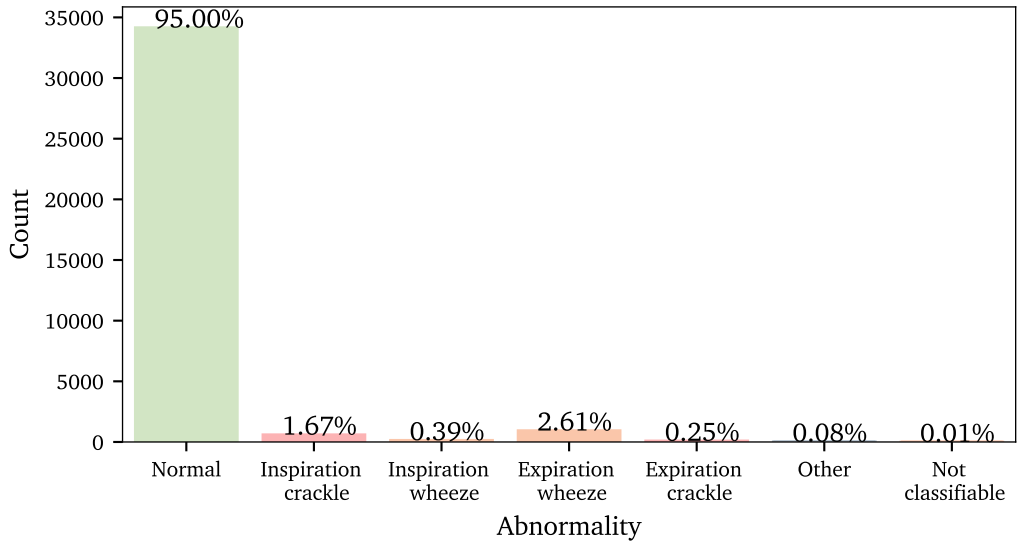


Figure 3.1.2: Histogram of the distribution of the samples, divided into labels. This distribution stems from the raw data provided by Tromsøundersøkelsen.

3.2 Preprocessing

The generalized workflow contains the important steps of downsampling, slicing and denoising, before splitting into train and test sets. These four preprocessing stages will be discussed in this section.

The main goal of the preprocessing is to sieve the raw data so that the remaining information has higher quality. Some compression and quality improving steps are essential, especially when dealing with audio data. Before performing the more problem-specific preprocessing steps, the following will steps are applied; downsampling, slicing, and denoising.

Downsampling: Downsampling is performed to limit the number of data points passed to various TSC algorithms, ensuring classification within justifiable time. From [79, 80] it is known that the dominant frequency range of the lung sound is between 150 Hz and 2000 Hz, hence by the Nyquist sampling criteria, the sampling frequency should at least be 4000 Hz to avoid aliasing. During experimentation, it was found that sometimes frequencies greater than 2000 Hz, were encountered. [81] states that the frequency range of lung sounds is 50 - 2500 Hz, moreover that frequencies as high as 4000 Hz can be found. Seeing as there are discrepancies in literature concerning the frequency range of lung sounds, a downsampled frequency of 8000 Hz is chosen for good measure.

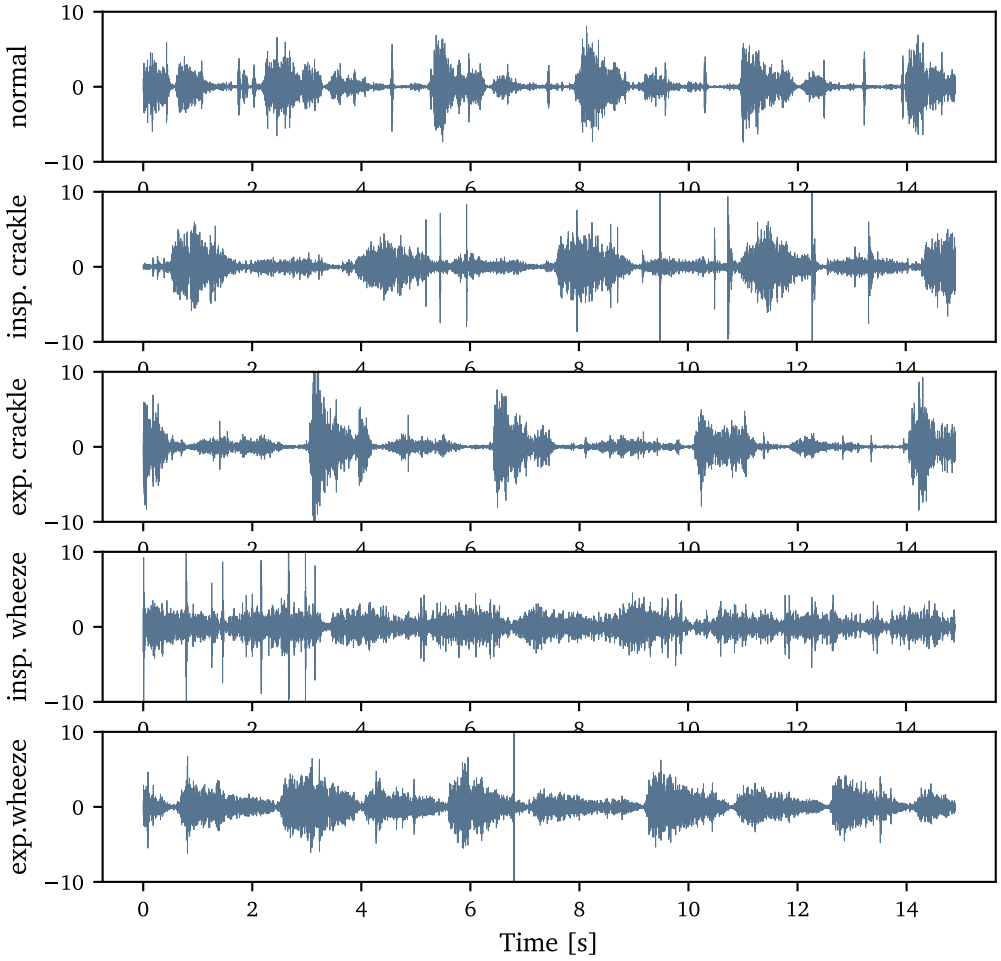


Figure 3.1.3: Samples of time series, containing different lung sounds; normal, inspiration crackle, expiration crackle, inspiration wheeze and expiration wheeze. Here the lung sound is filtered, z-normalised, and all samples are taken from recording location 5.

Slicing: A slice length of 500 ms was chosen, for reasons which will be discussed later in section 4. After running several steps of filtering out insufficient data, a total of 1 901 lung sound samples, with 500 ms duration, was gathered. These samples were found through a thorough labeling process, which involved making a customized GUI, with labeling the given dataset in mind. An important aspect of this compressed dataset is the number of patients, which is 632 patients, of various ages, with various medical backgrounds. The importance of the latter will also be discussed in section 4 .

Denoising: The denoising method proposed in [82] is performed on the slices, to diminish the noise from the surroundings. [82] utilizes wavelet denoising, which thresholds the signal in the wavelet domain to suppress noise. With inspiration from [83], the wavelet packet coefficients are modulated before the thresholding operation and demodulated afterward. The

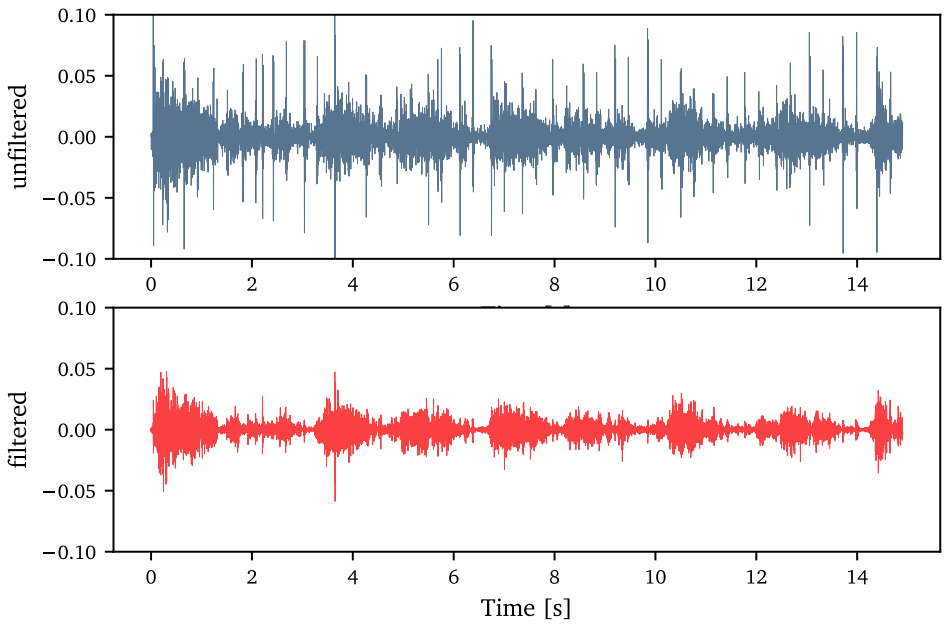


Figure 3.2.1: Recording containing wheeze abnormality, taken from recording location 6. *Blue* plot is before bandpass filtering, *red* plot is after bandpass filtering. A 12th order Butterworth bandpass filter is utilized. Low-cut frequency = 150 Hz, high-cut frequency = 2000 Hz.

modulation/demodulation transforms the data into an orthogonal domain, thus the thresholding does not distort the original lung sound. This process only diminishes the ambient noise, consequently, some additional denoising needs to be performed to remove the inevitable effect of the heart sound. Using a bandpass filter the heart sound was filtered out of the recording. With the knowledge that heart sounds usually reside in a lower frequency range, a 12th order Butterworth bandpass filter with a low-cut frequency of 150 Hz, and a high-cut frequency of 2000 Hz was applied to the lung sound. Butterworth filters are known for their flat frequency response in the passband, which is desired. The order of the filter is proposed by [84] and verified through experimentation. Applying the proposed filter with the suggested frequency range was found sufficient to remove the heart sound while keeping the abnormalities intact. A visualization of the lung sound before and after filtering out the heart sound is provided in Figure 3.2.1.

The distribution of the labels after the discussed preprocessing measures are taken is illustrated in Figure 3.2.2 .

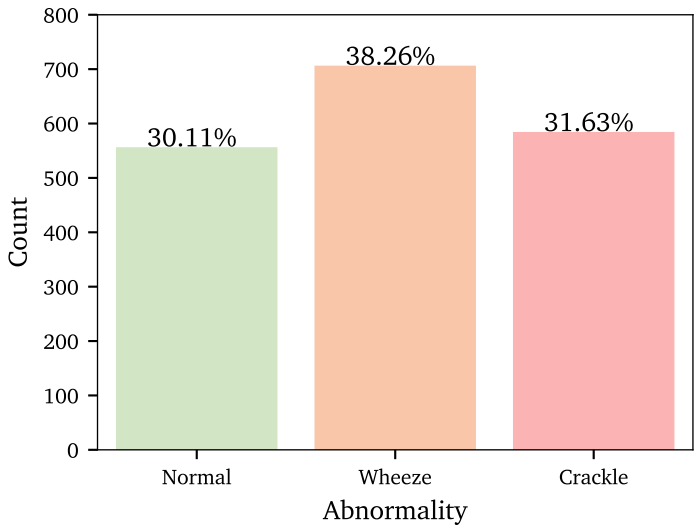


Figure 3.2.2: Distribution of labels after preprocessing the dataset. A summary of all the steps performed is provided in the appendix, section A.1.5.

3.3 Classification

While most of the mentioned TSC methods have achieved great success in terms of accuracy, the computational effort is a different story. In Table A.2.1, the computational complexity of all the TSC algorithms is presented. In Figure A.2.2 the mean computation time across all datasets in [25] is presented for the top TSC algorithms. ROCKET and ResNet stand out as the two most accurate, yet scalable algorithms. When working with audio data, which is the case with the lung sound, one can expect a long time series. To be able to process and classify a lung sound, one has to take into account how computationally complex the classification algorithm is, because some algorithms may be infeasible. Table 3.3.1 displays all the TSC algorithms that will be tested on the compressed lung sound dataset. In Table 3.3.1 the algorithms are divided into categories, based on common properties.

The Python library *sktime* [3], was employed for the following TSC methods; *ResNet*, *InceptionTime*, Contractable BOSS (*cBOSS*), Word ExtrAction for time SEries cLassification (*WEASEL*), Random Interval Spectral Ensemble (*RISE*), *Catch22*, *ROCKET*, *MiniROCKET* and Time Series Forest (*TSF*). Default settings for each method were applied. For the *preproject* and *preproject EEMD* routines, the preprocessing and feature extraction performed in the specialization project was performed [85]. A random forest was utilized for classification. The simple CNN structure proposed in Figure 3.3.1 was used in several experiments. For all deep learning implementations, the Python library *Keras* is applied [86]. Keras has the advantage of being simple to use,

Category	Algorithm
Deep Learning	ResNet, CNN, InceptionTime
Dictionary	cBOSS, WEASEL
Frequency Based	RISE, preproject, preproject-EEMD
Hybrid	Catch22, ROCKET, MiniROCKET
Interval	TSF

Table 3.3.1: TSC algorithms which will be explored for lung sound classification in this thesis. The algorithms are grouped by common properties.

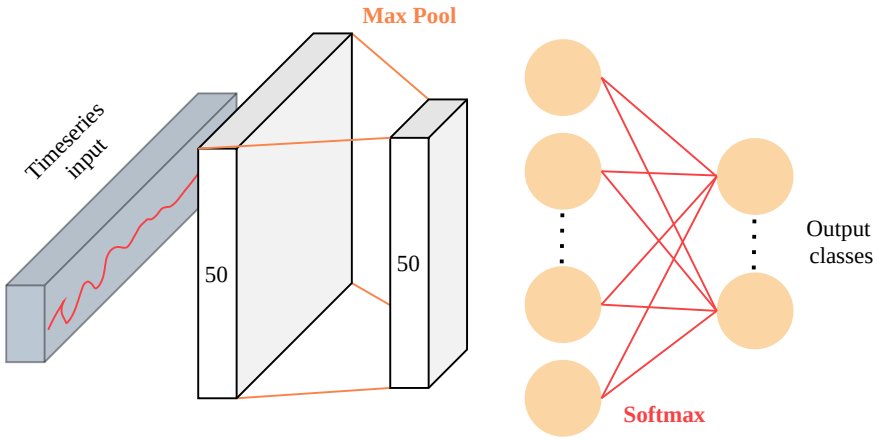


Figure 3.3.1: Simple CNN structure, used for TSC.

as well as having a great community for inspiration and support, therefore it is opted for in this thesis. A batch size of 100, as well as 100 epochs, were selected, through experimentation. This batch size gave the best tradeoff between computational effort and reliable gradient approximations.

3.4 Uncertainty Quantification

Uncertainty estimation will be included to add reliability to the classification. To quantify the model uncertainty two common approaches will be explored; one deep ensemble approach [64], and one Monte Carlo, MC, dropout approach [63]. The simple CNN illustrated in Figure 3.3.1, is used as the basis for the mentioned approaches, keeping the same batch size and number of epochs as before. In the deep ensemble method, the number of ensembles is set to 10, as suggested in [64].

3.5 Set-up

An overview of the proposed workflow is illustrated in Figure 3.5.1.

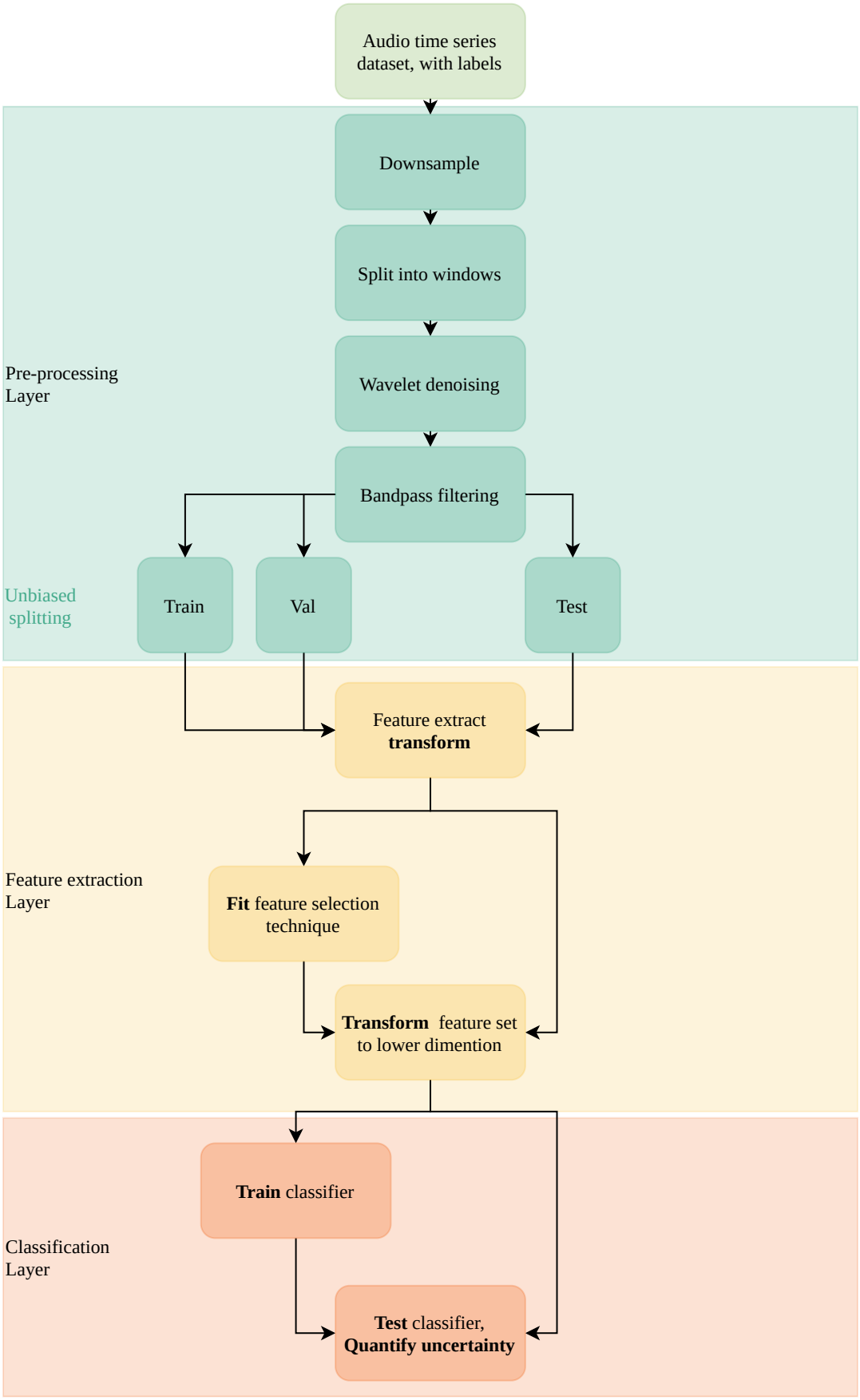


Figure 3.5.1: Generalised workflow for TSC with audio signals.

Chapter 4

Results and Discussions

In the following chapter, the results from several experiments will be presented. Firstly a synthetic classification problem is created, to evaluate the solvability of the lung sound classification problem. Afterward, solutions to the lung sound classification problem will be proposed and discussed.

4.1 Synthetic Dataset

Synthetic datasets are created to assess the solvability of the task at hand. After doing some basic exploration of the introduced dataset, the classification problem proved to be quite complex. To see if the poor performance is due to insufficient labeling, or because of the complexity of the data, several synthetic datasets were created. A pure cosine signal with a fixed frequency and fixed duration was introduced to the lung sound recordings. The placement of the pure tone within the lung sound recordings was selected at random. 4000 lung sounds classified as 'normal' was gathered and split into unbiased train, test, and validation datasets. In this context unbiased means that no same patient will appear in the train, test, and validation datasets. 50 % of all the recordings were corrupted with the pure tone, furthermore, the label was set to *'synthetic'*.

To evaluate the impact of the frequency, duration, and amplitude of the synthetic sound, a multitude of combinations were tested. One synthetic sound was randomly introduced to both entire 15 s recordings, and 5 s slices. The simple CNN proposed in 3.3.1 was utilized for classification, to determine the basic capability of discriminating between synthetic and normal samples. A convolution operation, with a set amount of filters, can be conceived as applying some statistical filtering, to summarize the data. Thus applying a convolution layer is a simple, yet efficient way of determining if the time series can be summarized by statistics. Figure 4.1.1 gives a summary of how the characteristics of the abnormality affect the performance when classifying between synthetic and normal lung sounds, in a 15 s recording. While Figure 4.1.2 represents the same for the 5 s recording.

Interestingly the performance when increasing the duration of the abnormality is similar for both the 15 s and the 5 s recordings. For the human ear, increasing the duration makes the abnormality even more distinguishable, yet the CNN is unable to capture this. A reason for this could be the set amplitude = 1 and frequency = 400 Hz. Both of these characteristics place the synthetic sound in the range of the lung sound, in terms of both amplitude and power. In the CNN, overlapping signals together with abnormality to duration ratio seems to be the main problem.

In the 15 s recording increasing the amplitude was the only characteristic which leads to an increase in performance. Considering that the CNN acts similar to a filter, averaging over the entire recording, increasing the amplitude would have the most impact on the statistics in the time-domain.

For the 5 s recording, raising the frequency of the abnormality had a notable impact. From 4.1.2c it seems that once the frequency is increased to outside the range of the expected lung sound, the CNN can separate the two classes.

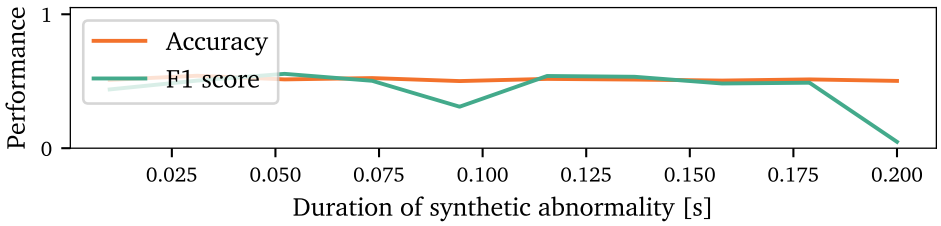
The recording duration is a crucial constituent, seeing an increase in performance just by going from a 15 s recording to a 5 s slice. This increasing performance suggests that segmentation is vital. Dividing the classification problem into sub-segments separates the complex lung sounds into simpler segments, where the noise statistics have less effect on the statistics of the abnormalities.

4.2 Lung Sound Classification

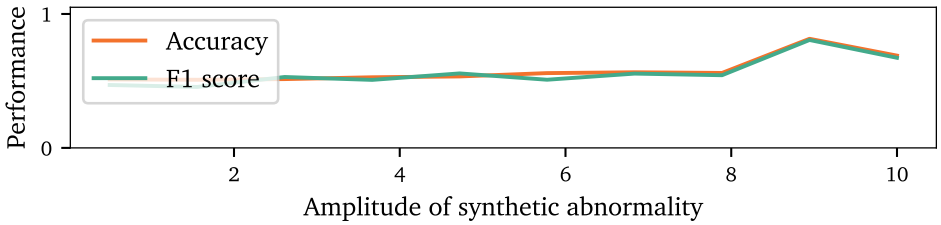
After feeding all observations through the preprocessing steps discussed, several TSC algorithms were explored and compared. In this section the results, in terms of classification, scalability, and interpretability, will be discussed further, however, firstly one path-changing matter will be addressed.

4.2.1 Data Leakage problem

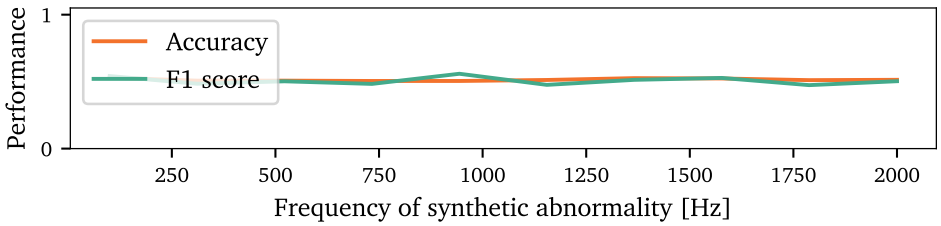
One problem encountered in the process of establishing the generalized workflow for lung sound classification was the data leakage problem. Data leakage is when information from outside the training dataset is used to create the predictive model. In the case of medical data, one has to make sure that no same patient appears in both the training set and the testing set, as this is a source of bias. Having the same patient appear in both datasets may lead to a predictor that recognizes the relationship between patient and label, instead of the correlation between data and label. To establish that this intricacy can lead to unjustified performance, two cases will be



(a) Increasing duration. Frequency is fixed to 400 Hz , and amplitude to 1.



(b) Increasing amplitude. Duration is fixed to 100 ms , and frequency to 400 Hz.

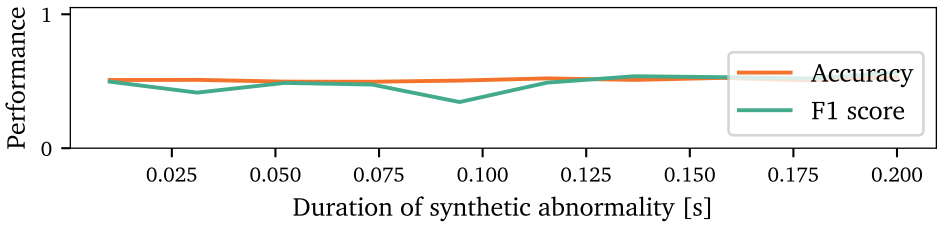


(c) Increasing frequency. Duration is fixed to 100 ms , and amplitude to 1.

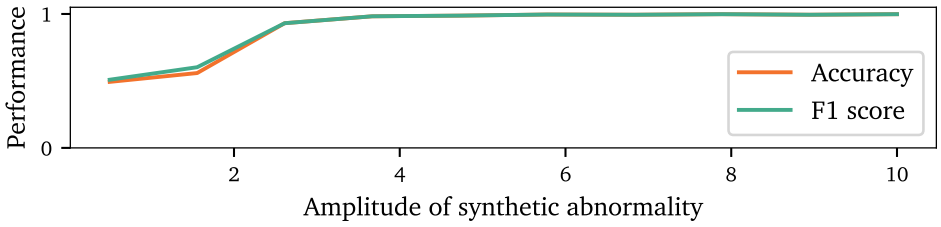
Figure 4.1.1: Introducing synthetic abnormality with increasing duration, amplitude and frequency, to 50 % of the normal samples. The abnormality is added to the 15 s lung sound window. A simple CNN is used for classification.

compared. In the first case, the patient ID is *not* taken into account when splitting into train and test set. For the second case, the patient ID is used as a basis when splitting into train and test sets, hence there is no overlap between the patients in the training and testing sets. The second case is more realistic, in terms of diagnosing new unseen recordings. The two cases are compared in terms of performance in Figure 4.2.1. Clearly *case one* achieves unrealistic results.

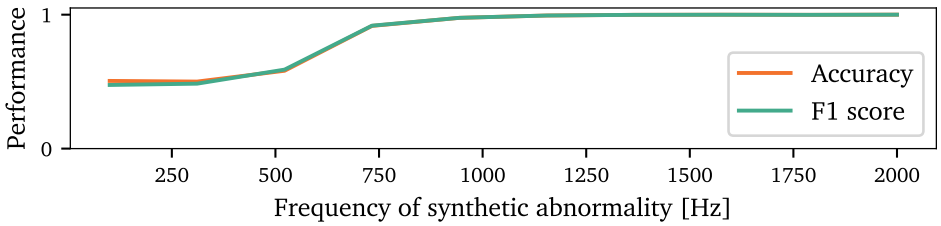
From Figure 4.2.1b, poor classification performance is met when making sure that the train, test, and validation datasets are impartial. The dive into the synthetic classification problem clarified that the problem with classifying lung sounds is many-fold. Firstly, an issue in the case of the crackle/wheeze/normal dataset is the labeling and window length. If the windows are extracted automatically to a set slice length, and each slice inherits the label of the original window, then there might be an issue of having uncertain labels. Secondly, there is a clear pattern of not being able to classify sounds that are inside the scope of the lung sound. From the bestowed



(a) Increasing duration. Frequency is fixed to 400 Hz , and amplitude to 1.



(b) Increasing amplitude. Duration is fixed to 100 ms , and frequency to 400 Hz.

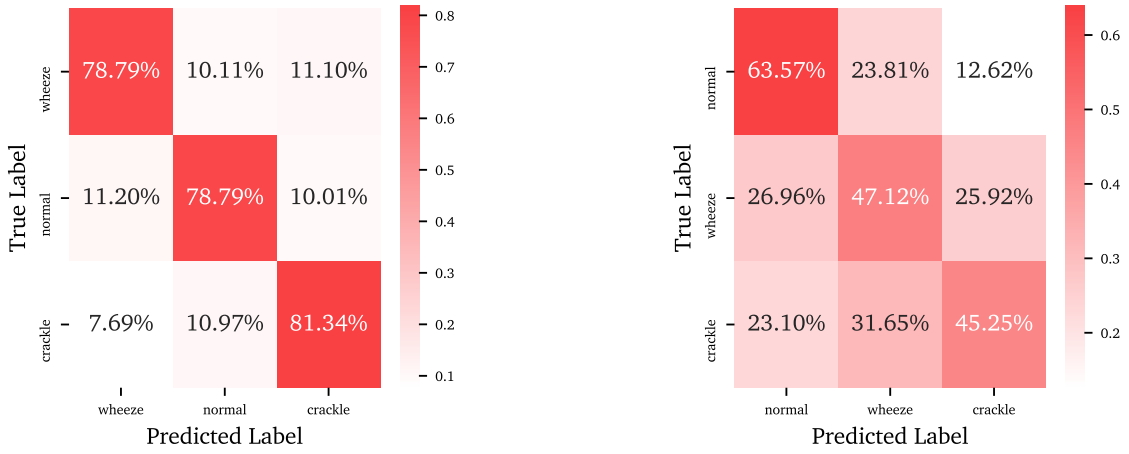


(c) Increasing frequency. Duration is fixed to 100 ms , and amplitude to 1.

Figure 4.1.2: Introducing synthetic abnormality with increasing characteristics to 50 % of the normal samples. The abnormality is added to the 5 s lung sound window. A simple CNN is used for classification.

knowledge about wheezes and crackles, this is an issue, seeing as both wheezes and crackles can appear in the expected frequency range of normal lung sounds.

For further work, a simplified variant of the lung sound classification problem is created. To explore the effect a large patient group has, a new dataset was constructed. Using Python, a simple GUI was designed, with the intent of filtering the dataset. One sample recording was extracted from each of the patients left after filtering out flawed data and balancing the classes; crackle, wheeze and normal. The customized GUI navigated the user through all samples, playing a 500 ms window at a time. Weak recordings, or recordings with a high amount of noise were discarded during this process. For each patient, a maximum of 3 recordings was archived. This measure was taken to make sure that the classifier does not overfit a particular individual during training. The *compressed dataset* created gives insight into the effect segmentation has. Unless specified otherwise, the compressed dataset is used to obtain the results which will be



(a) *Case 1:* Biased dataset

(b) *Case 2:* Unbiased dataset.

Figure 4.2.1: Confusion matrix after classifying the data extracted using the specialization project routine. In *case 1* the patient id is not taken into account when splitting into train,test and validation sets, while in *case 2* this issue is considered. The datasets originate from the processing steps summarized in the appendix, section A.1.3. A fixed window length of 5 seconds is set. Classifications are provided using a random forest.

presented and discussed below.

4.2.2 Comparing classification algorithms

In the specialization project leading to this thesis, approaches for classifying lung sounds were reviewed. The data leakage problem was not taken into account in the specialization project, hence the results were unrealistic. Consequently, more reasonable classification results will now be compared to several scalable state-of-the-art TSC algorithms. The feature extraction technique proposed in the specialization project has the advantage of being tailored for audio TSC. Expectantly the performance should not differ much when comparing the leading TSC methods with the proposed procedure. In the specialization project, an abundance of features was proposed. These features should be filtered through feature selection, before classification, to avoid redundancy. Another advocate for the specialization project procedure is the ability to extract features, without having to equalize the length of the audio signal. For all other TSC methods, a step of zero-padding has to be introduced, before applying the transformation/classification algorithms. Zero-padding may interfere with the classification, by smoothing the information contained in the signal. Regarding the compressed dataset, all samples are specified to be of equal length, thus avoiding this issue. However, for future work, where the signal may be segmented into inspiration and expiration phases, the specialization project feature extraction procedure is preferred.

Considering that data leakage corrupted the results obtained in the specialization project, an additional feature extraction procedure will be employed. An Ensemble Empirical Mode Decomposition, EEMD, approach is used to decompose the signal into Intrinsic Mode Functions. The ensemble approach is utilized to avoid mode mixing, which may lead to inconsistent decompositions. EEMD did not perform well in the specialization project, however, this might be due to data leakage.

Figures 4.2.2 and 4.2.3 summarizes the performance of several TSC algorithms. Figure 4.2.2 displays the accuracies and f1-scores obtained when classifying between crackles and wheezes. In the plot, the solid bars represent the accuracy, while the more transparent bars represent the f1-scores. From the figure, it is clear that the difference between the best-performing algorithms is not substantial. In Figure 4.2.4a the computation times is compared for the same classification problem. Combining the results presented in Figures 4.2.2 and 4.2.4a, the specialization project procedure and MiniROCKET comes out as winners. The specialization project routine is one of the fastest algorithms, while still delivering acceptable predictions. The same reasoning can be given for the classification problem classifying between crackle, wheeze, and normal. Here the performance is given in Figure 4.2.3, while the computation time is displayed in Figure 4.2.4b. When classifying between the three classes, the performance plummets. One cause for the drop in performance is thought to be the variability in the normal lung sound.

A notable observation from Figures 4.2.2 and 4.2.3 is that the two leading deep TSC approaches, ResNet and InceptionTime, deliver substantially worse results than other TSC algorithms. Differences present in the results are thought to be because of under-fitting, this statement is also motivated by the gap between the accuracy and f1-score. The assumption of under-fitting is further strengthened by low training accuracies for both algorithms. Under-fitting could be due to the complex pattern that is present in lung sounds, which the network is not able to capture. Another issue could be too short training time. Strangely, the simple CNN delivers better performance than the complex ResNet and InceptionTime algorithms. One would expect that the more complex models could summarize the complex lung sound better, however, this does not seem to be the case. A reason for this unusual issue may be the noisy nature of the lung sound data, which can be summarized better by simple statistics, rather than deep models.

During exploration, the intent was to test the state-of-the-art algorithm HIVE-COTE, however, this was infeasible. Shaplet Transform, which is a vital part of the HIVE-COTE method, could not be employed for the lung sound classification problem. The algorithms struggled with finding

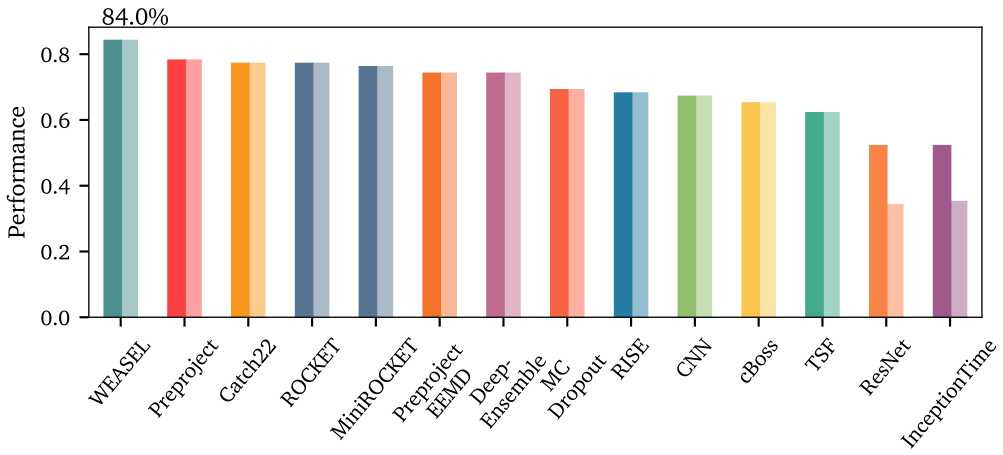


Figure 4.2.2: Comparing the accuracy and f1 score of the different TSC algorithms, when classifying between crackle and wheeze. The more transparent colored bar represents the f1 score. The dataset presented in section A.1.6 is utilized.

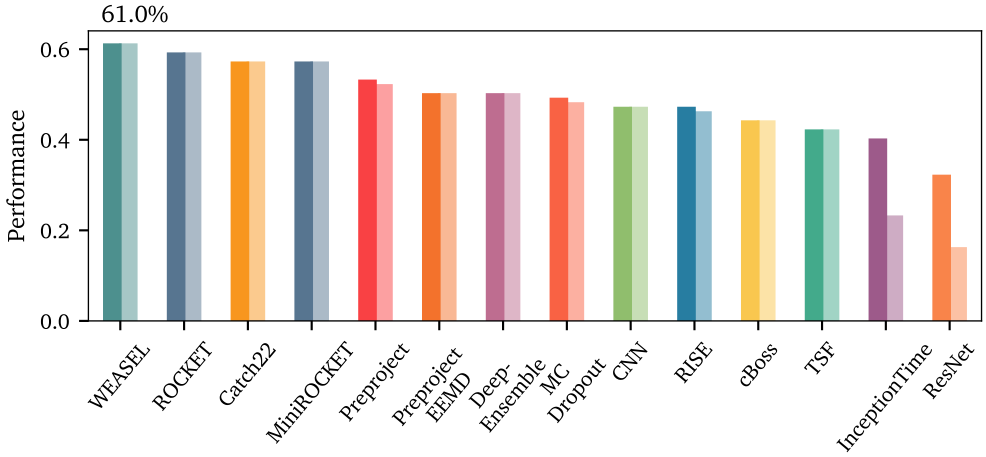
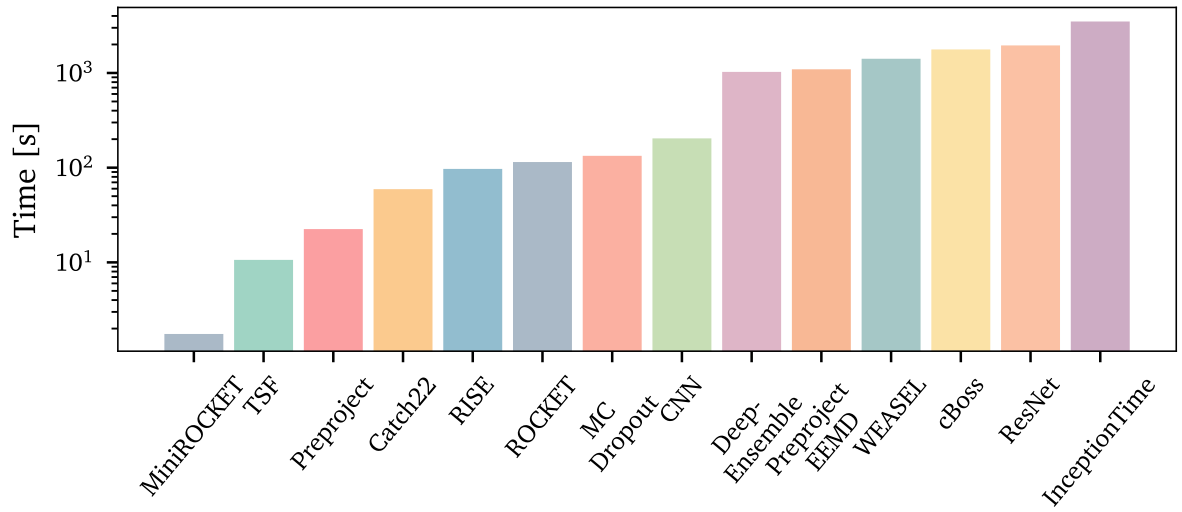


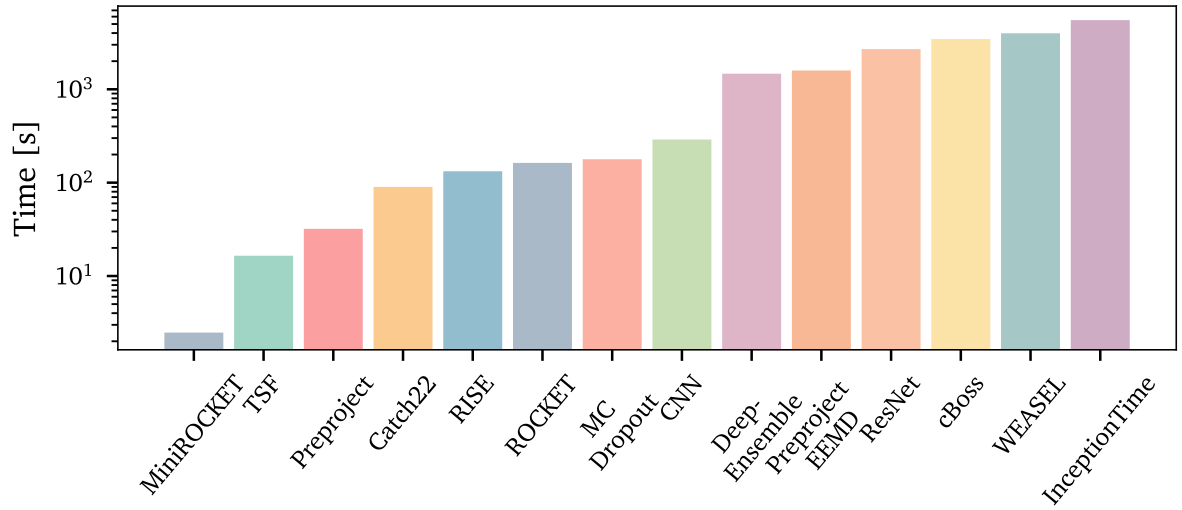
Figure 4.2.3: Comparing the accuracy and f1 score of the different TSC algorithms, when classifying between crackle, wheeze and normal. The more transparent colored bar represents the f1 score. The dataset presented in section A.1.5 is utilized.

common shapelets, which is to be expected for a naturally noisy signal, such as the lung sound. As a result of this lacking ability to extract valuable information, both the Shapelet Transform and HIVE-COTE methods were rejected.

To give a better understanding of the performance, and the struggles of each classifier, confusion matrices for each approach are provided in the Appendix, section A.3.



(a) Classifying between crackle and wheeze

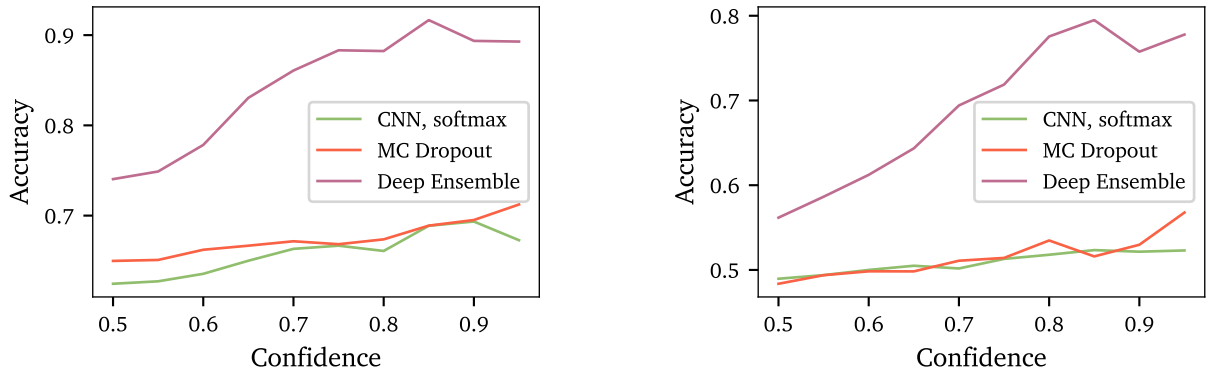


(b) Classifying between crackle, wheeze and normal

Figure 4.2.4: Comparing computation time for different TSC algorithms, when using the compressed lung sound dataset.

4.3 Model Uncertainty Quantification

One solution to the issue of having a faulty classifier due to the complexity of the classification problem is to introduce uncertainty quantification. With reliable uncertainty estimates, one can trust that the classifier at least knows what it does not know. In this section, uncertainty estimates provided using MC-dropout, and a deep ensemble is going to be presented and considered. The compressed dataset will be introduced to an increasing amount of noise, pitch, and shift-augmentation. Uncertainty estimates will be evaluated based on the accuracy, Brier score, ECE, box plots, and accuracy-rejection/confidence curves. As the 500 ms slices of normal lung



(a) Crackle vs. Wheeze. Section A.1.6 summarizes the creation of the dataset.

(b) Crackle vs. Wheeze vs. Normal. Section A.1.5 summarizes the creation of the dataset.

Figure 4.3.1: Accuracy-rejection/confidence curve, to illustrate the effect abstaining from classification has on the classification accuracy. This curve gives an indication of the classifiers knowledge about uncertainty.

sounds are selected at random, they will have more varying characteristics than crackles and wheezes, who mainly appear in the inspiration/expiration phases. With this issue in mind, presumably one will encounter worse discrimination capacity on the normal lung sound. To study if the uncertainty estimation methods can discriminate more consistent data, only the wheezes and crackles are focused on in this section.

Figure 4.3.1 shows the accuracy-rejection/confidence curves obtained after using the three stated approaches for classifying the lung sounds. Here *no* manipulation is performed on the test set. From Figure 4.3.1 it is evident that the deep ensemble can capture the uncertainty in such a way that abstaining from classification would increase the accuracy. Compared to the CNN and the MC-dropout methods, the deep ensemble seems much more reliable, where the accuracy is increasing with the growing confidence.

4.3.1 Expected Results

When increasing the extent of data manipulation, one expects the accuracy of the classifier to degenerate.

The *Brier Score*, as discussed in Chapter 2, measures the accuracy of probabilistic predictions. A lower Brier score indicates a sounder calibration of the predictions. When increasing the amount of augmentation applied to the test set, one can expect the Brier Score to rise, because it grows increasingly unsure.

ECE, summarizes the relationship between accuracy and confidence of a model. Larger values of

ECE hints that the model is inadequately calibrated for the task at hand. ECE should grow when entering highly augmented data as the model is not trained on such data.
When a model is well-calibrated, the *accuracy-rejection/confidence* curve should increase, meaning that the more confident predictions are truly correct. Finally, a boxplot of the predictions is expected to become more uncertain for a well-calibrated classifier, once augmented data is introduced.

4.3.2 Augmented noise

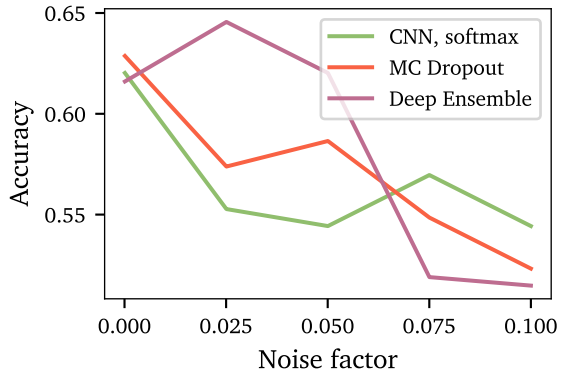
Firstly random noise is added, with increasing amplitude. For each escalation, a simple CNN, a simple CNN with added MC-dropout, and an ensemble of simple CNNs are employed to classify the manipulated data. The classifiers are trained on the normal training and validation sets, while the test data is manipulated. Manipulating the data with random noise should not have a huge effect on the performance of the predictions. A well-calibrated classifier can filter out the added noise. Figures 4.3.2 - 4.3.4 confirm this expectation. Confusion matrices obtained after increasing the noise effect show that all classifiers are more likely to output crackle predictions, for larger noise increments.

4.3.3 Augmented pitch

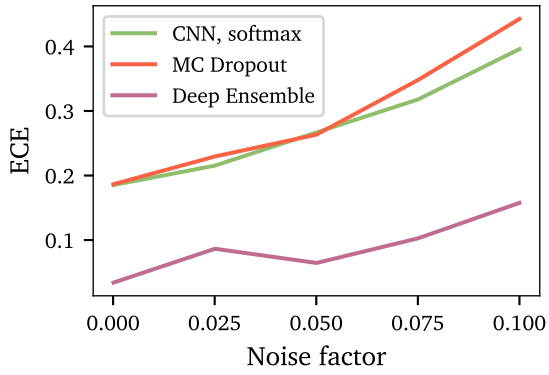
The pitch of the lung sounds in the test set is augmented with an increasing factor. The classifiers utilized for the escalating noise experiment are also employed for this experiment. Interestingly, when amplifying the pitch of the lung sound, the accuracy of the CNN with and without MC-dropout went slightly up, as seen in Figure 4.3.5a. Figures 4.3.5b and 4.3.5c indicate that the calibration of the classifications is not drastically affected by the pitch augmentation.
The box-plot provided in Figure 4.3.7 indicates that increasing the pitch steers the samples towards becoming more similar, hence the predicted probabilities also become more alike. The similarity of the augmented data quickly pushes the simple CNN to become overly confident in all predictions. MC-dropout delays this effect to some extent. Deep ensemble seems to be the most robust against pitch augmentation. Confusion matrices for the CNN with and without MC-dropout show that when increasing the pitch, the classifiers become overly confident in classifying all samples as crackles, the same applies for the deep ensemble.

4.3.4 Augmented shift

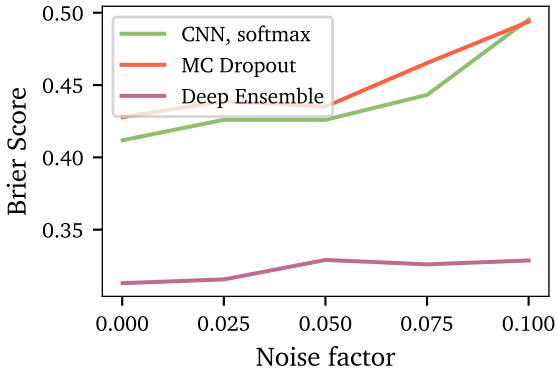
The test set is shifted using the variance of the observations, along with a rising shift factor. Shifting the data leads to a test set where all samples resemble each other, thus producing limited



(a) Accuracy



(b) ECE



(c) Brier score

Figure 4.3.2: Comparing metrics when increasing the amount of noise added to the test set. A CNN, CNN with MC-dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

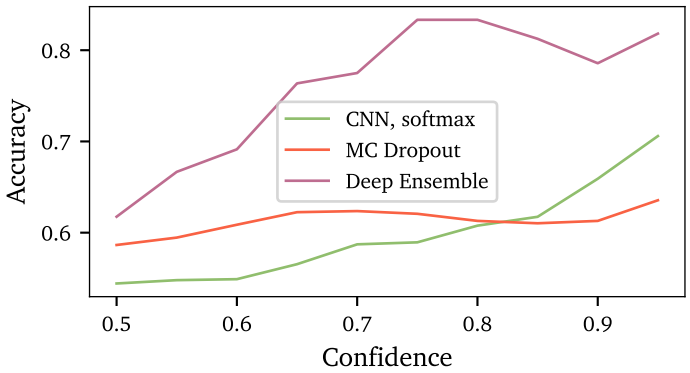


Figure 4.3.3: Accuracy-rejection curve when the noise factor is set to 0.005. A CNN, CNN with MC-dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

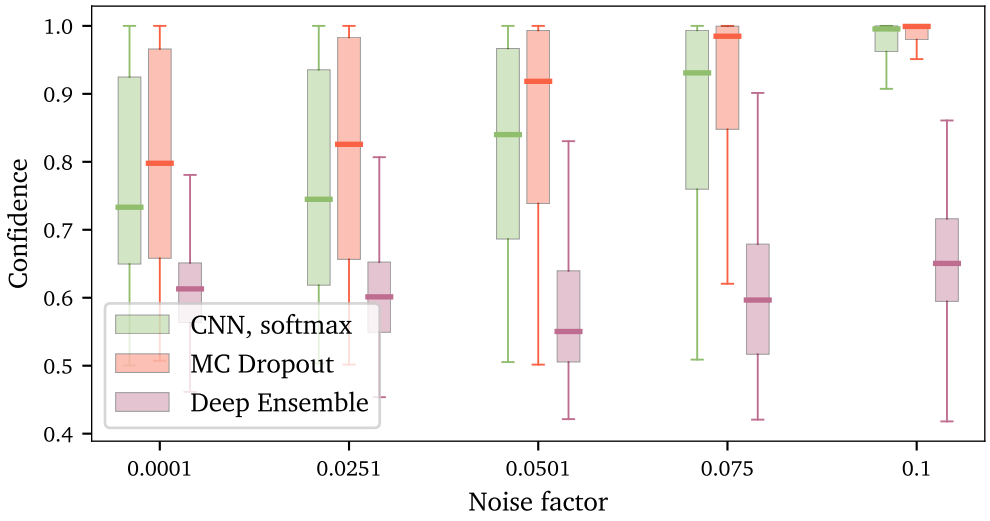
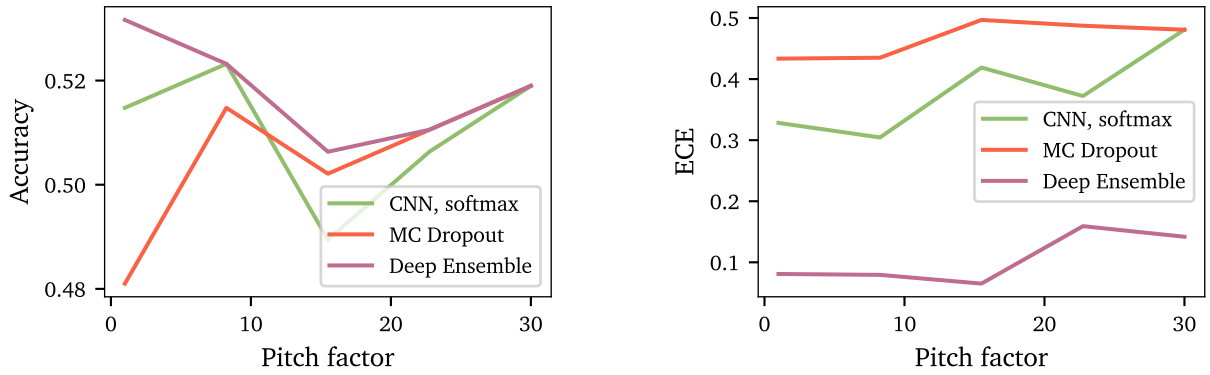
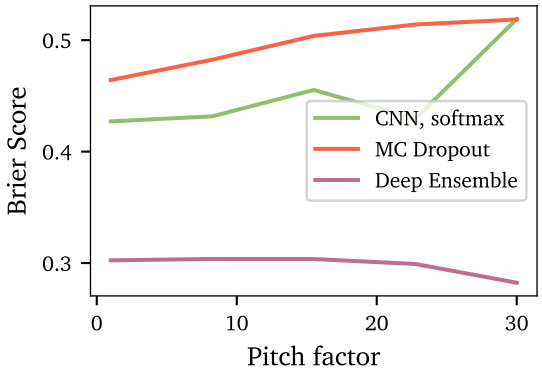


Figure 4.3.4: Boxplot of the confidence of the different classifiers, when increasing the amount of noise added to the test set. The dataset presented in section A.1.6 is utilized.



(a) Accuracy

(b) ECE



(c) Brier score

Figure 4.3.5: Comparing metrics when increasing the pitch of the test set. A CNN, CNN with MC-dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

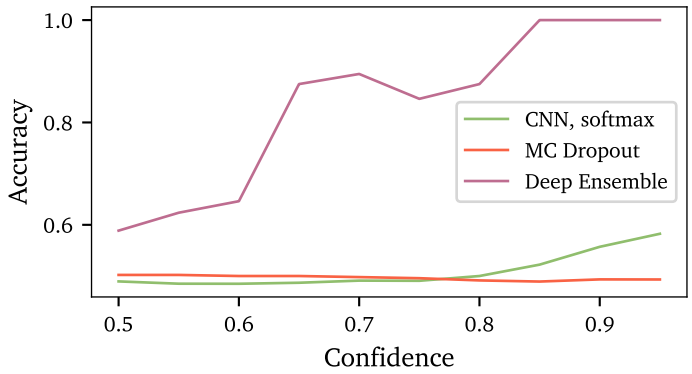


Figure 4.3.6: Accuracy-rejection curve when the pitch factor is set to 15.5. A CNN, CNN with MC-dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

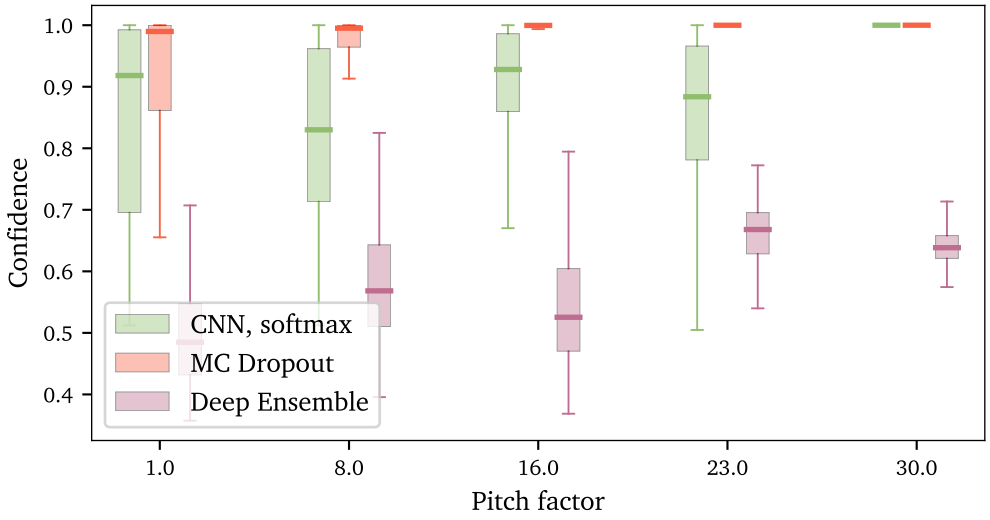
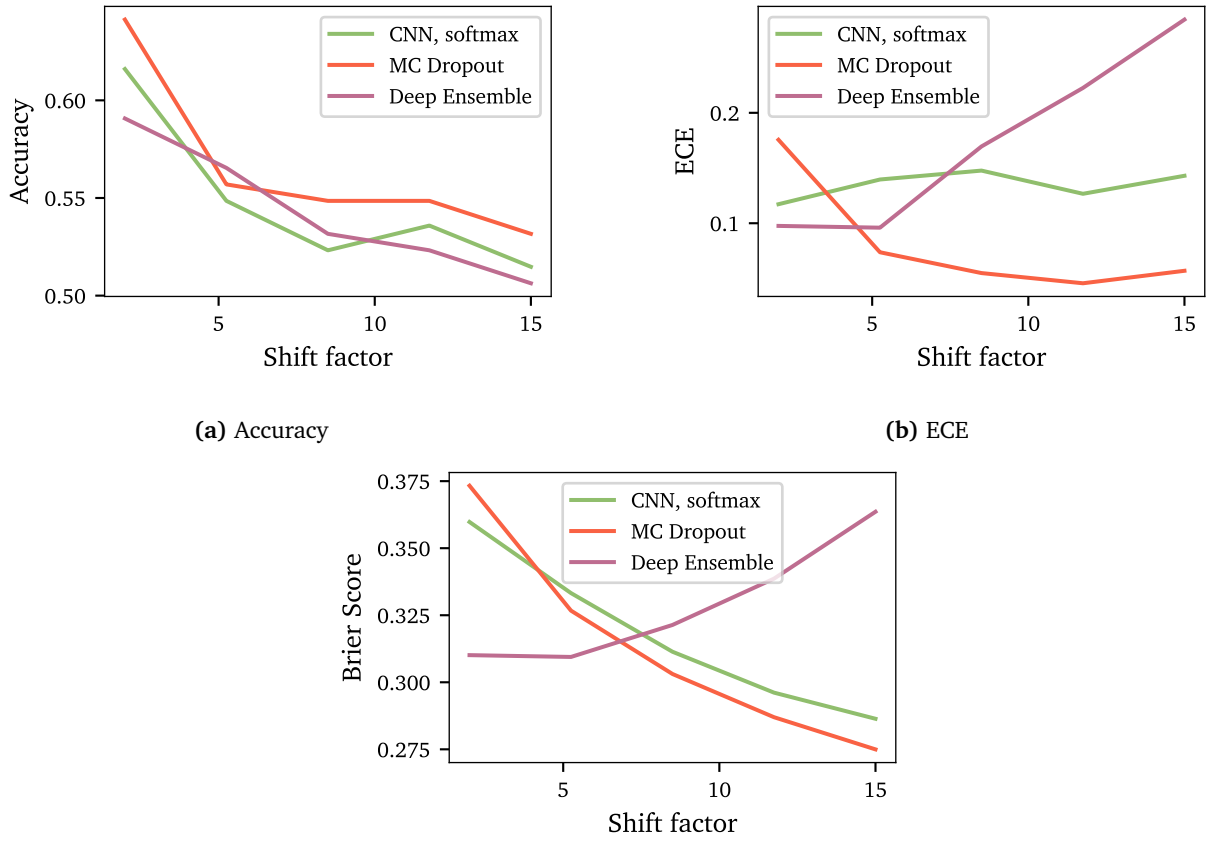


Figure 4.3.7: Boxplot of the confidence of the different classifiers, when increasing the pitch of the test set. The dataset presented in section A.1.6 is utilized.

variability in the predictions. Curiously, the Brier score of the CNN with and without MC-dropout seems to decline, while the ECE stays the same, as displayed in Figure 4.3.8. This indicates that the relationship between confidence and accuracy stays the same, while the predicted probabilities are closer to the actual output. The reason for this behavior appears to be that all samples grow more alike, motivating similar predictions. As seen in Figure 4.3.10, the predictions on augmented data stabilize at around 70 % confidence, thus few predictions are overly confident, leading to a lower Brier score. The decreasing Brier score does not mean that the accuracy stays intact, Figure 4.3.8a clearly states oppositely. After observing the confusion matrices resulting from each increment of dataset shift, it is clear that all of the three approaches are prone to classify most samples as wheezes.



(a) Accuracy

(b) ECE

(c) Brier score

Figure 4.3.8: Comparing metrics when increasing the shift of the test set. A CNN, CNN with MC-dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

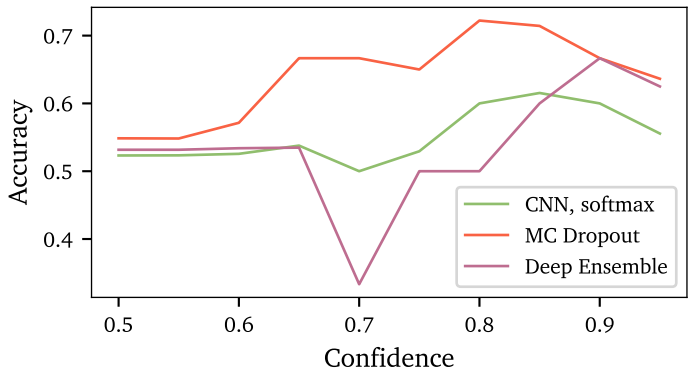


Figure 4.3.9: Accuracy-rejection curve when the shift factor is set to 15.5. A CNN, CNN with MC-dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

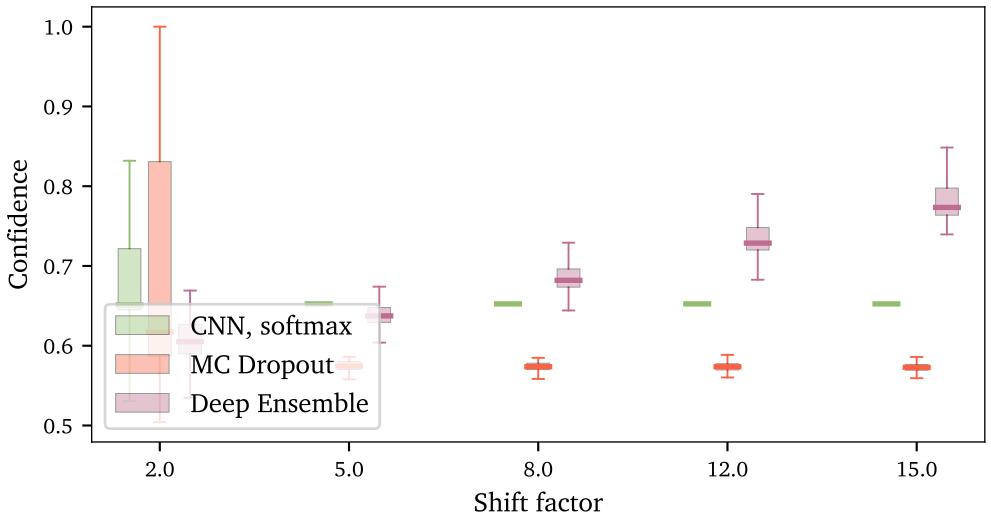


Figure 4.3.10: Boxplot of the confidence of the different classifiers, when increasing the shift of the test set. The dataset presented in section A.1.6 is utilized.

4.4 Discussion

Following the manual labeling of the dataset, performed by a person without a medical background, the dataset is introduced to a new type of bias. After compressing the dataset into 1 901 samples of 500 ms, followed by an unbiased dataset split, the classifier is still not ready for a real-world setting. Seeing as random 500 ms segments of lung sounds were extracted to define the normal lung sound, the classifier is not yet trained objectively. The fact that most wheezes were extracted from the expiration phase, and that most crackles were extracted from the inspiration phase, could trigger the classifier to recognize the pattern of inspiration and expiration, rather than crackles and wheezes. A new type of data leakage is therefore injected.

Unquestionably the classification accuracy when using the dataset without any type of bias is not sufficient enough to be used in a real-world setting. Lung sounds vary significantly from individual to individual, hence this poor performance is to be expected. One solution which could be interesting is to calibrate a classifier to one patient. Performing the calibration would improve the classification performance. This solution still requires doctors to record and label lung sounds, yet, this is only necessary a limited amount of times per patient. In acute or primary care, this method will be inapt, however for continuous monitoring or follow-ups of chronically ill patients, a calibrated classifier could ease the workload of medical practitioners.

The noisy nature of lung sounds makes classification a difficult task. Both the quality and characteristics of the lung sound depend a great deal on the placement of the stethoscope on the

body, and the unique build of the patient being recorded. When doctors detect abnormalities in lung sounds, they consider the timing within the breathing phase closely. To that end, comparing crackles, wheezes, and normal sounds without the context of the entire breathing cycle would in any case be difficult. This matter could be an explanation as to why the three-class classification problem seems to be strenuous. It is however interesting that humans find diagnosis simpler when being provided longer duration lung sounds, while computers have the opposite notion. This poses an interesting question of why tasks that humans find effortless, can be all the more difficult for computers to solve, and vice versa. The observation suggests that AI still has a long way to go.

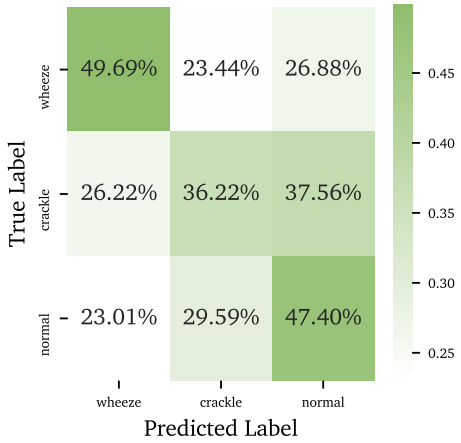
Encountering the data leakage problem motivated another deep dive into the current research on the topic of lung sound classification. Whereas previous literature surveys performed focused on the preprocessing, feature extraction, and classification techniques, the new focus was now the origin of the data. Interestingly, several papers which are deemed as state-of-the-art, seem to have overlooked the bias introduced by having the same patients appear both in the train and test dataset. A summary of the findings in the follow-up literature survey is presented in Table 4.4.1. From Table 4.4.1 it seems that segmentation and train/test-split plays a significant role when it comes to the resulting accuracy. The unbiased procedure that provided the best accuracy, of 96.02 %, had the advantage of only dealing with short segments. In [87, 88], whole breathing cycles were utilized with unbiased data, this provided classification results similar to the ones experienced in this thesis, when using 15 s lung sound recordings.

To experiment with the impact segmentation has on the results, different window lengths were explored. Here the original Tromøundersøkelsen dataset was utilized, meaning the dataset that had not been through the process of manual labeling. The data was segmented using three procedures; fixed window split *without* overlapping windows, fixed window split *with* 50 % overlapping windows, and a customized split on silence procedure. One issue with the proposed automatic splitting procedures is that there is no guarantee that the actual abnormality falls into the new window. Some lung recordings have abnormalities that do not appear periodically, thus resulting in inaccurate labels once the recording is segmented. Algorithm 1 describes the simplified procedure used for detecting breathing cycles in lung sounds. Splitting the lung sound into segments based on silence is considered a valid approach for identifying breathing cycles. It is observed that most expiration and inspiration cycles are separated by a short window of silence. This segmentation approach is somewhat simplified, and not all segments will contain

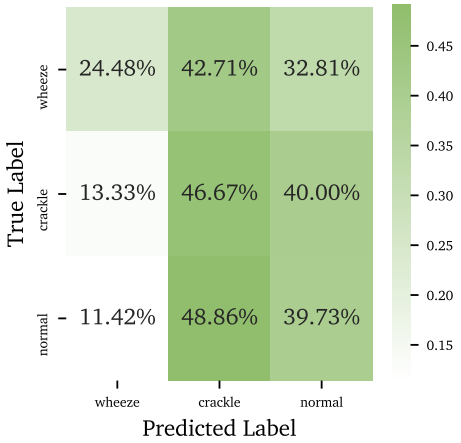
Paper	Biased	Performance	Comment
[17]	Yes	Accuracy = 86.00 %	17,930 lung sounds from 1630 subjects. Samples were composed of 10s slices.
[89]	Yes	Accuracy = 94.02 %	One sample is one breathing cycle. The dataset is quite small.
[90]	Yes	Accuracy = 97.67 %	One sample has a duration of between 80 ms and 200 ms. Used Leave-One-Out-Cross-Validation, with multiple samples from the same patient, to achieve the given accuracy.
[91]	Yes	Accuracy = 98.34 %	Signal segmentation is applied to shorten the duration.
[92]	Yes	Accuracy = 97.20 %	The samples consist of 256 point segments. With 26 patients, there were 13 healthy and 13 not healthy. Clearly this is biased
[93]	No	Accuracy = 96.02 %	100 ms segments were classified manually, and fed to a classification algorithm.
[87]	No	Sensitivity = 49.50% , Specificity = 39.37%	Segmented into breathing cycles.
[88]	No	Sensitivity = 35.50%, Specificity = 62.55%	Segmented into breathing cycles.

Table 4.4.1: Literature survey looking for data leakage

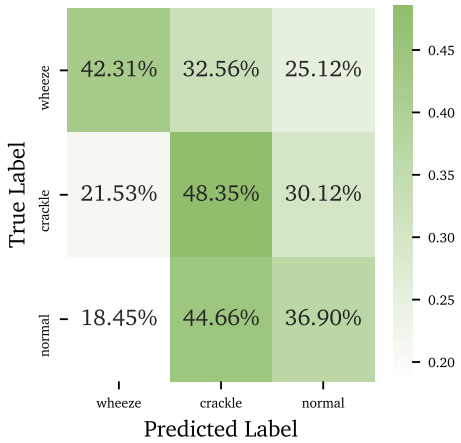
perfect cycles. From empirical testing, the optimal max dB audio level is defined, as well as the frame length and hop length to calculate RMSE, which furthermore is used to indicate silence. Datasets with all the windowing procedures described are introduced to the simple CNN architecture, from Figure 3.3.1. Figure 4.4.1 displays the resulting confusion matrices. Window lengths of both 3 s and 5 s are utilized, both with and without overlap, in Figures 4.4.1a - 4.4.1d. In Figures 4.4.1e and 4.4.1f, a minimum window length of 2 s and 2.5 s is given to algorithm 1. The minimum window length is set as an input to algorithm 1, after noting that the split based on RMSE sometimes became too eager. There does not seem to be a substantial difference in performance when comparing the proposed automatic windowing methods, after examining the results in Figure 4.4.1.



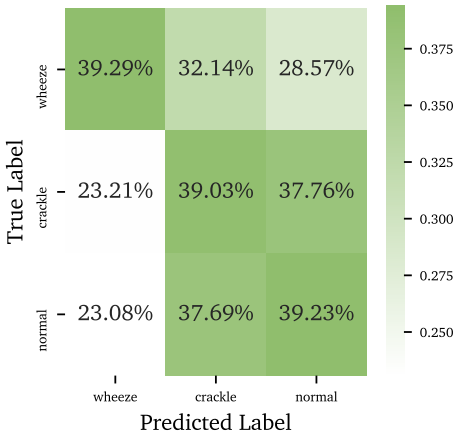
(a) 3 second window used with *no* overlap.



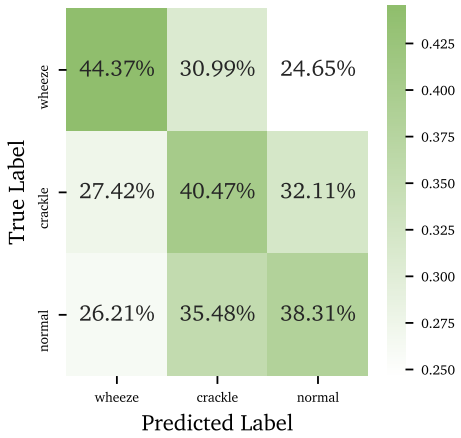
(b) 5 second window used with *no* overlap.



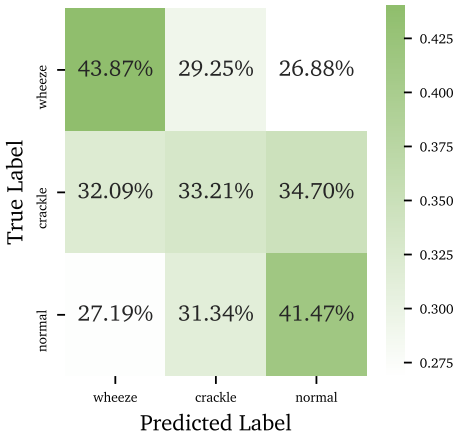
(c) 3 second window used with 50% overlap.



(d) 5 second window used with 50% overlap.



(e) Silence detection split, min 2 s window.



(f) Silence detection split, min 2,5 s window.

Figure 4.4.1: Confusion matrix after classifying the preprocessed data using specialization project features and a random forest for classification. All filtering steps included (bandpass filering, wavelet denoising and downsampling). The dataset is split into unbiased train, test and validation sets.

Algorithm 1 Split lung sound into segments based on silence

```
procedure SPLIT_ON_SILENCE(LS, min_l)        ▷ min_l is the minimum length of the new segment
    subs ← SPLIT_RMSE(LS)                    ▷ Use RMSE windows to detect silence.
    segs ← None
    curr_seg ← None
    for sub ← subs do
        add sub to curr_seg
        if len(curr_seg) > min_l then
            segs ← curr_seg
            curr_seg ← None
        end if
    end for
    return segs
end procedure
```

Designing experiments with the intent of being utilized for machine learning, is an important step moving forward. The abundance of data available today has the disadvantage of being recorded by, and primarily for, humans. If the lung sounds used in this project were recorded by a team of both doctors and data analysts, then the recording procedure could be designed to simplify automatic data analysis. Such alterations could be; simple labeling of the inspiration and expiration phases during real-time recording, rejecting samples that contain too much noise, and more detailed labeling. Another quality improvement could be to record the lung sound from multiple locations simultaneously. Simultaneous recordings could simplify lung sound and heart sound segmentation, allowing for methods such as Independent Component Analysis to be applied.

When it comes to uncertainty estimation, the deep ensemble classifier seems to handle augmentation the best. All classifiers become increasingly sure of one class, as the augmentation increases. Both the CNN with and without MC-dropout became overly confident when introducing noise and pitch augmentation, while the deep ensemble managed to stay seemingly critical for all augmentation tasks. One advocate for using a deep ensemble is that disagreements due to low-quality data may be captured by an individual network, thus increasing the robustness for each network added to the ensemble. To choose the optimal number of networks one has

to weigh the computational effort against the robustness. Once the saturation point is reached, adding more networks will not necessarily improve the performance.

Chapter 5

Conclusion and future work

In this master's thesis, we have introduced the problem of lung sound classification, intending to detect abnormalities in lung sounds. Several state-of-the-art approaches within TSC were identified, along with a proposed feature extraction procedure based on statistical properties. Conducted experiments have found that a large portion of research has unrealistic results, moreover that the classification task at hand is more complex than initially thought.

The complexity of the problem was determined by producing a synthetic dataset. The synthetic dataset proved that the signal-to-noise ratio present in lung sounds makes it difficult to distinguish the synthetic data from the normal data. Separating the synthetic abnormality and the normal lung sound becomes even more demanding when the characteristic properties of the abnormality overlap with the lung sound. We demonstrated that segmenting the lung sound would decrease the signal-to-noise ratio, thus increasing the classification performance. With a manually labeled lung sound dataset, composed of 500 ms segments, a classification accuracy of 84% was reached, when discriminating between crackles and wheezes. These classification results are not valid in a practical setting, because of the bias introduced by the manual labeling though they still highlight the importance of segmentation.

Uncertainty estimation was suggested as a solution to the lung sound classification problem. With acceptable uncertainty estimates the predictor can alert the user when it has insufficient knowledge, which would be more socially justifiable than making a wrongful classification.

Summarized the main conclusions of this thesis are:

- **Data leakage was identified as a flaw in current research about abnormality detection in lung sounds**. Patients appearing in both the training and testing datasets appear to be a significant predicament in several articles on lung sound classification. To stress this issue a new dataset was created focusing on having a diverse patient selection. The new dataset does not yet meet impartial data demand, due to the bias introduced by having the manual

labeling performed by a person without a medical background.

- **Segmentation of the lung sound is necessary to detect abnormalities within the lung sound.** Segmenting the lung sounds into shorter slices, that were thoroughly labeled, showed that a greater classification accuracy can be achieved. This observation motivates future research to include segmentation as a necessary preprocessing step.
- **The signal-to-noise ratio is proved to be one of the major drawbacks of lung sound classification.** As proved by inspecting the synthetic classification problem, one of the issues with lung sound classification is the noisy nature of the lung sound. The synthetic classification problem was solvable if the imposed abnormality has properties that reside outside characteristical lung sound.

5.1 Future Work

The following points should be the focus of future research in the field of lung sound classification:

- **Unbiased and accurate segmentation of the lung sounds should be performed.** The work conducted in this thesis highlights several research flaws. Results presented in this thesis are not unbiased, thus they do not reflect the ability to discriminate between lung sound abnormalities in a practical setting. To this end, future research on lung sound classification should focus on automatic segmentation, dividing the classification problem into sub-problems. With the knowledge that wheezes mostly appear during the expiration phase, and crackles during the inspiration phase, an inspiration/expiration segmentation would be reasonable. Automatic segmentation should be prioritized, as this could simplify the labeling procedure, and make the lung sound classification more applicable in a practical setting.
- **Detailed labels can result in classifications that are more useful in clinical care.** The timing of the abnormalities within the inspiration/expiration phases can indicate different issues, as stated in section 1.1. A more thoroughly labeled dataset could be beneficial for detailed classification, which may lead to an improved quality of treatment.
- **Design of the experimental setup is a necessary step for simplifying digital analysis.** As touched upon in section 4.4, multidisciplinary crossovers are necessary for moving forward in the age of big data. If engineers with experience in data analysis can assist doctors in gathering relevant data, then this might lead to more efficient and accurate digital monitoring.

5.2 Self-reflection

Lastly, I would say that the most interesting observation for me concerns the deviances in published literature. The initial impression was that research on the topic of lung sound classification was vast, and an impressive amount of articles have been written. A substantial takeaway from working on this thesis has thus been the lack of reliability in published research. I went from thinking that the solution to the lung sound classification problem was found, to understanding that research is underdeveloped. The majority of the papers studied focused on data processing, and classification techniques, however, little attention was given to data acquisition. Maybe, in this age of innovative processing techniques appearing around every corner, we should take a step back, and consider the seemingly obvious; that the data itself is the problem, not our algorithms. Developing new recording techniques, and protocols could improve the performance of ML in an array of medical problems, not only lung sound classification. With more comprehensive data-centric research, we can push AI from academic research to become applicable in practical settings. Recording techniques we use to gather medical data today were not intended for machines to analyze, so maybe we should contemplate updating the structure, rather than the final layer.

Bibliography

[1] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[2] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, pages 18–25. Citeseer, 2015.

[3] Markus Löning, Anthony Bagnall, Sajaysurya Ganesh, Viktor Kazakov, Jason Lines, and Franz J Király. sktime: A Unified Interface for Machine Learning with Time Series. In *Workshop on Systems for ML at NeurIPS 2019*, 2019.

[4] Andrine Elsetrønning. Autonomous lung sound classification. `https://github.com/AndrineElse/autonomousLungSoundClassification`, 2021.

[5] Carl H Lubba, Sarab S Sethi, Philip Knaute, Simon R Schultz, Ben D Fulcher, and Nick S Jones. catch22: Canonical time-series characteristics. *Data Mining and Knowledge Discovery*, 33(6):1821–1852, 2019.

[6] Andrea Coravos, Jennifer C Goldsack, Daniel R Karlin, Camille Nebeker, Eric Perakslis, Noah Zimmerman, and M Kelley Erb. Digital medicine: a primer on measurement. *Digital biomarkers*, 3(2):31–71, 2019.

[7] Mina K Chung, Steven J Szymkiewicz, Mingyuan Shao, Edwin Zishiri, Mark J Niebauer, Bruce D Lindsay, and Patrick J Tchou. Aggregate national experience with the wearable cardioverter-defibrillator: event rates, compliance, and survival. *Journal of the American College of Cardiology*, 56(3):194–203, 2010.

[8] Federico Boscari, S Galasso, G Acciaroli, A Facchinetti, MC Marescotti, A Avogaro, and D Bruttomesso. Head-to-head comparison of the accuracy of abbott freestyle libre and dexcom g5 mobile. *Nutrition, Metabolism and Cardiovascular Diseases*, 28(4):425–427, 2018.

[9] Raymond LH Murphy. In defense of the stethoscope. *Respiratory Care*, 53(3):355–369, 2008.

[10] The top 10 causes of death. `https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death`. Accessed: 2021-05-26.

[11] Just how do deaths due to covid-19 stack up? `https://www.thinkglobalhealth.org/article/just-how-do-deaths-due-covid-19-stack`. Accessed: 2021-05-26.

[12] Akinari Noda, Takeshi Saraya, Kikuko Morita, Masaoki Saito, Teppei Shimasaki, Daisuke Kurai, Keitaro Nakamoto, and Haruyuki Ishii. Evidence of the sequential changes of lung sounds in covid-19 pneumonia using a novel wireless stethoscope with the telemedicine system. *Internal Medicine*, 59(24):3213–3216, 2020.

[13] Kelvin Kai-Wang To, Owen Tak-Yin Tsang, Wai-Shing Leung, Anthony Raymond Tam, Tak-Chiu Wu, David Christopher Lung, Cyril Chik-Yan Yip, Jian-Piao Cai, Jacky Man-Chun Chan, Thomas Shiu-Hong Chik, et al. Temporal profiles of viral load in posterior oropharyngeal saliva samples and serum antibody responses during infection by sars-cov-2: an observational cohort study. *The Lancet Infectious Diseases*, 20(5):565–574, 2020.

[14] Xu Li and Xiaochun Ma. Acute respiratory failure in covid-19: is it "typical" ards? *Critical Care*, 24:1–5, 2020.

[15] Aaron Stupple, David Singerman, and Leo Anthony Celi. The reproducibility crisis in the age of digital medicine. *NPJ digital medicine*, 2(1):1–3, 2019.

[16] Gorkem Serbes, Sezer Ulukaya, and Yasemin P Kahya. An automated lung sound preprocessing and classification system based onspectral analysis methods. In *International Conference on Biomedical and Health Informatics*, pages 45–49. Springer, 2017.

[17] Murat Aykanat, Özkan Kılıç, Bahar Kurt, and Sevgi Saryal. Classification of lung sounds using convolutional neural networks. *EURASIP Journal on Image and Video Processing*, 2017(1):1–9, 2017.

[18] Jyotibdha Acharya and Arindam Basu. Deep neural network for respiratory sound classification in wearable devices enabled by patient specific model tuning. *IEEE transactions on biomedical circuits and systems*, 14(3):535–544, 2020.

[19] Yun-Xia Liu, Yang Yang, and Yue-Hui Chen. Lung sound classification based on hilbert-huang transform features and multilayer perceptron network. In *2017 Asia-Pacific Signal*

and Information Processing Association Annual Summit and Conference (APSIPA ASC), pages 765–768. IEEE, 2017.

[20] S Don. Random subset feature selection and classification of lung sound. *Procedia Computer Science*, 167:313–322, 2020.

[21] Nikša Jakovljević and Tatjana Lončar-Turukalo. Hidden markov model based respiratory sound classification. In *International Conference on Biomedical and Health Informatics*, pages 39–43. Springer, 2017.

[22] Giuseppe Bonaccorso. *Machine learning algorithms*. Packt Publishing Ltd, 2017.

[23] Yoshua Bengio. *Learning deep architectures for AI*. Now Publishers Inc, 2009.

[24] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23(1):89–109, 2001.

[25] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. The ucr time series classification archive, October 2018. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`.

[26] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31(3):606–660, 2017.

[27] Young-Seon Jeong, Myong K Jeong, and Olufemi A Omitaomu. Weighted dynamic time warping for time series classification. *Pattern recognition*, 44(9):2231–2240, 2011.

[28] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015.

[29] Patrick Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

[30] Jon Hills, Jason Lines, Edgaras Baranauskas, James Mapp, and Anthony Bagnall. Classification of time series by shapelet transformation. *Data mining and knowledge discovery*, 28(4):851–881, 2014.

[31] Jason Lines, Sarah Taylor, and Anthony Bagnall. Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles. *ACM Trans. Knowl. Discov. Data*, 12(5), July 2018.

[32] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2522–2535, 2015.

[33] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[34] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.

[35] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[36] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[37] Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

[38] Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. *arXiv preprint arXiv:2012.08791*, 2020.

[39] Ben D. Fulcher and Nick S. Jones. hctsa: A computational framework for automated time-series phenotyping using massive feature extraction. *Cell Systems*, 5(5):527–531.e3, 2017.

[40] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[41] Alessio Benavoli, Giorgio Corani, and Francesca Mangili. Should we really use post-hoc tests based on mean-ranks? *The Journal of Machine Learning Research*, 17(1):152–161, 2016.

[42] Malik Sajjad Ahmed Nadeem, Jean-Daniel Zucker, and Blaise Hanczar. Accuracy-rejection curves (arcs) for comparing classification methods with a reject option. In *Machine Learning in Systems Biology*, pages 65–81. PMLR, 2009.

[43] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*, 2019.

[44] Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

[45] Marc C Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[46] Robin Senge, Stefan Bösner, Krzysztof Dembczyński, Jörg Haasenritter, Oliver Hirsch, Norbert Donner-Banzhoff, and Eyke Hüllermeier. Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences*, 255:16–29, 2014.

[47] Lotta Meijerink, Giovanni Cinà, and Michele Tonutti. Uncertainty estimation for classification and risk prediction in medical settings. *arXiv preprint arXiv:2004.05824*, 2020.

[48] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, pages 1–50, 2021.

[49] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. *Logistic regression*. Springer, 2002.

[50] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[51] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2):131–163, 1997.

[52] J Bernardo, J Berger, APAFMS Dawid, A Smith, et al. Regression and classification using gaussian process priors. *Bayesian statistics*, 6:475, 1998.

[53] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[54] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699, 2002.

[55] Scott Ferson and Lev R Ginzburg. Different methods are needed to propagate ignorance and variability. *Reliability Engineering & System Safety*, 54(2-3):133–144, 1996.

[56] Andrew Y Ng and Michael I Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848, 2002.

[57] Simon J Sheather. Density estimation. *Statistical science*, pages 588–597, 2004.

[58] Jonathan Q Li and Andrew R Barron. Mixture density estimation. In *NIPS*, volume 12, pages 279–285, 1999.

[59] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.

[60] Shehroz S Khan and Michael G Madden. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(3):345–374, 2014.

[61] M. Perello-Nieto, T. D. M. E. S. Filho, M. Kull, and P. Flach. Background check: A general technique to build more reliable and versatile classifiers. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1143–1148, 2016.

[62] Eyke Hüllermeier and Klaus Brinker. Learning valued preference structures for solving classification problems. *Fuzzy Sets and Systems*, 159(18):2337–2352, 2008.

[63] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[64] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016.

[65] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[66] Malvin H Kalos and Paula A Whitlock. *Monte carlo methods*. John Wiley & Sons, 2009.

[67] Igor Kononenko. Bayesian neural networks. *Biological Cybernetics*, 61(5):361–370, 1989.

[68] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[69] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[70] Steven Lehrer. *Understanding lung sounds*. Steven Lehrer, 2018.

[71] ARA Sovijarvi, F Dalmasso, J Vanderschoot, LP Malmberg, G Righini, and SAT Stoneman. Definition of terms for applications of respiratory sounds. *European Respiratory Review*, 10(77):597–610, 2000.

[72] Sonia Charleston-Villalobos, G Martinez-Hernandez, Ramón Gonzalez-Camarena, Georgina Chi-Lem, José G Carrillo, and Tomás Aljama-Corrales. Assessment of multichannel lung sounds parameterization for two-class classification in interstitial lung disease patients. *Computers in Biology and Medicine*, 41(7):473–482, 2011.

[73] Paul Forgacs. Crackles and wheezes. *The Lancet*, 290(7508):203–205, 1967.

[74] Raymond LH Murphy Jr, Stephen K Holford, and William C Knowler. Visual lung-sound characterization by time-expanded wave-form analysis. *New England Journal of Medicine*, 296(17):968–971, 1977.

[75] T Kaisia, A Sovijärvi, P Piirilä, HM Rajala, S Haltsonen, and T Rosqvist. Validated method for automatic detection of lung sound crackles. *Medical and Biological Engineering and Computing*, 29(5):517–521, 1991.

[76] Nandini Sengupta, Md Sahidullah, and Goutam Saha. Lung sound classification using cepstral-based statistical features. *Computers in Biology and Medicine*, 75:118 – 129, 2016.

[77] T. Richard Fenton, Hans Pasterkamp, A. Tal, and Victor Chernick. Automated spectral characterization of wheezing in asthmatic children. *IEEE Transactions on Biomedical Engineering*, BME-32(1):50–55, 1985.

[78] Juan Carlos Aviles-Solis, Cristina Jacome, A Davidsen, R Einarsen, Sophie Vanbelle, Hans Pasterkamp, and Hasse Melbye. Prevalence and clinical associations of wheezes and crackles in the general population: the tromsø study. *BMC pulmonary medicine*, 19(1):1–11, 2019.

[79] Rajkumar Palaniappan, Kenneth Sundaraj, and Nizam Uddin Ahamed. Machine learning in lung sound analysis: a systematic review. *Biocybernetics and Biomedical Engineering*, 33(3):129–135, 2013.

[80] Volker Gross, Anke Dittmar, Thomas Penzel, Frank Schuttler, and Peter Von Wichert. The relationship between normal lung sounds, age, and gender. *American journal of respiratory and critical care medicine*, 162(3):905–909, 2000.

[81] Sandra Reichert, Raymond Gass, Christian Brandt, and Emmanuel Andrès. Analysis of respiratory sounds: state of the art. *Clinical medicine. Circulatory, respiratory and pulmonary medicine*, 2:CCRPM–S530, 2008.

[82] M. Bahoura, M. Hubin, and M. Ketata. Respiratory sounds denoising using wavelet packets. In *Proceedings of the 2nd International Conference on Bioelectromagnetism (Cat. No.98TH8269)*, pages 11–12, 1998.

[83] David L Donoho. De-noising by soft-thresholding. *IEEE transactions on information theory*, 41(3):613–627, 1995.

[84] Bruno M Rocha, Dimitris Filos, Luís Mendes, Gorkem Serbes, Sezer Ulukaya, Yasemin P Kahya, Nikša Jakovljevic, Tatjana L Turukalo, Ioannis M Vogiatzis, Eleni Perantoni, et al. An open access database for the evaluation of respiratory sound classification algorithms. *Physiological measurement*, 40(3):035001, 2019.

[85] Andrine Elsetrønning. Analysis of lung sound data for abnormality detection, 2020. TTK4550 specialization project report.

[86] François Chollet et al. Keras. `https://keras.io`, 2015.

[87] N. Jakovljević and T. Lončar-Turukalo. Hidden Markov model based respiratory sound classification. In Nicos Maglaveras, Ioanna Chouvarda, and Paulo de Carvalho, editors, *Precision Medicine Powered by pHealth and Connected Health*, pages 39–43, Singapore, 2018. Springer Singapore.

[88] Gaëtan Chambres, Pierre Hanna, and Myriam Desainte-Catherine. Automatic detection of patient with respiratory diseases using lung sound analysis. In *2018 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–6. IEEE, 2018.

[89] Arumugam Kandaswamy, C Sathish Kumar, Rm Pl Ramanathan, S Jayaraman, and N Malmurugan. Neural classification of lung sounds using wavelet coefficients. *Computers in biology and medicine*, 34(6):523–537, 2004.

[90] Sezer Ulukaya, Gorkem Serbes, and Yasemin P Kahya. Overcomplete discrete wavelet transform based respiratory sound discrimination with feature and decision level fusion. *Biomedical Signal Processing and Control*, 38:322–336, 2017.

[91] Shengkun Xie, Feng Jin, Sridhar Krishnan, and Farook Sattar. Signal feature extraction by multi-scale pca and its application to respiratory sound classification. *Medical & biological engineering & computing*, 50(7):759–768, 2012.

[92] Gorkem Serbes, C Okan Sakar, Yasemin P Kahya, and Nizamettin Aydin. Feature extraction using time-frequency/scale analysis and ensemble of feature sets for crackle detection. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3314–3317. IEEE, 2011.

[93] Kevin E Forkheim, David Scuse, and Hans Pasterkamp. A comparison of neural network models for wheeze detection. In *IEEE WESCANEX 95. Communications, Power, and Computing. Conference Proceedings*, volume 1, pages 214–219. IEEE, 1995.

Appendix A

Appendix

A.1 Dataset creation methods

A.1.1 15 s recording, crackle/wheeze/normal

15 s lung sounds are extracted directly from the Tromsø Dataset, and a target value is set to be either wheeze, crackle or normal, depending on the annotation given by the Tromsø Study group. The lung sounds are put through the steps of downsampling, denoising and bandpass filtering, before being fed directly into a classifier. When training a classifier the train, test and validation split of the dataset took the patient ID into account, hence no same patient appears across datasets.

A.1.2 15 s recording, crackle/wheeze

15 s lung sounds are extracted directly from the Tromsø Dataset, and a target value is set to be either wheeze or crackle, depending on the annotation given by the Tromsø Study group. The lung sounds are put through the steps of downsampling, denoising and bandpass filtering, before being fed directly into a classifier. When training a classifier the train, test and validation split of the dataset took the patient ID into account, hence no same patient appears across datasets.

A.1.3 Fixed length slices, no overlap, crackle/wheeze/normal

Lung sounds are extracted directly from the Tromsø Dataset, and a target value is set to be either wheeze, crackle or normal, depending on the annotation given by the Tromsø Study group. The lung sounds are put through the steps of downsampling, denoising and bandpass filtering, before being split into fixed length, non-overlapping windows. Each window inherits the same label as the original recording. When training a classifier the train, test and validation split of the dataset took the patient ID into account, hence no same patient appears across datasets.

A.1.4 Fixed length slices, no overlap, crackle/wheeze

Lung sounds are extracted directly from the Tromsø Dataset, and a target value is set to be either wheeze or crackle, depending on the annotation given by the Tromsø Study group. The lung sounds are put through the steps of downsampling, denoising and bandpass filtering, before being split into fixed length, non-overlapping windows. Each window inherits the same label as the original recording. When training a classifier the train, test and validation split of the dataset took the patient ID into account, hence no same patient appears across datasets.

A.1.5 Manually labeled 500 ms slices, crackle/wheeze/normal

A compressed dataset is composed of 500 ms samples extracted from 632 recordings of 632 patients. Recordings are marked as containing crackle (194), wheeze (254) or normal (186) lung sounds. A maximum of 3 samples were taken from each of the 632 recordings. Inheriting the recording label of the entire recording is inaccurate, thus manual labeling was performed. A customized GUI was created with the sole purpose of extracting valuable 500 ms samples from the 632 recordings. A GUI navigates the user through the 15 s recording, playing one 500 ms slice at a time. The original label of the recording is stated, so that the user knows what to look for. If a 500 ms slice contains the abnormalitiy indicated by the original label, then the user can save the new sample, along with the target value. When a maximum of 3 samples are extracted from one recording the program moves on to the next patient. The new 500 ms lung sounds are put through the steps of downsampling, denoising and bandpass filtering. When training a classifier the train, test and validation split of the dataset took the patient ID into account, hence no same patient appears across datasets.

A.1.6 Manually labeled 500 ms slices, crackle/wheeze

The compressed dataset is composed of 500 ms samples extracted from 448 recordings of 448 patients. Recordings are marked as containing crackle (194) or wheeze (254) lung sounds. A maximum of 3 samples were taken from each of the 632 recordings. Inheriting the recording label of the entire recording is inaccurate, thus manual labeling was performed. A customized GUI was created with the sole purpose of extracting valuable 500 ms samples from the 448 recordings. A GUI navigates the user through the 15 s recording, playing one 500 ms slice at a time. The original label of the recording is stated, so that the user knows what to look for. If a 500 ms slice contains the abnormalitiy indicated by the original label, then the user can save the new sample, along with the target value. When a maximum of 3 samples are extracted from one recording the program moves on to the next patient. The new 500 ms lung sounds are put

through the steps of downsampling, denoising and bandpass filtering. When training a classifier the train, test and validation split of the dataset took the patient ID into account, hence no same patient appears across datasets.

A.2 UEA & UCR Time Series Classification Results

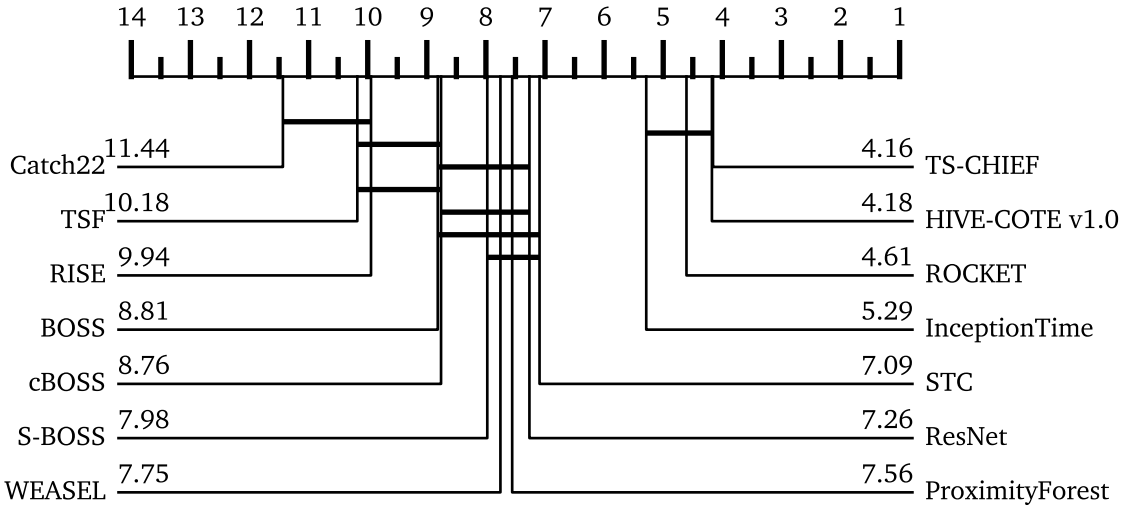


Figure A.2.1: Critical difference plot of 14 classifiers, tested on 112 TSC problems. Solid bars indicate cliques, where there is no significant difference in rank. Ranking is based on mean accuracy.

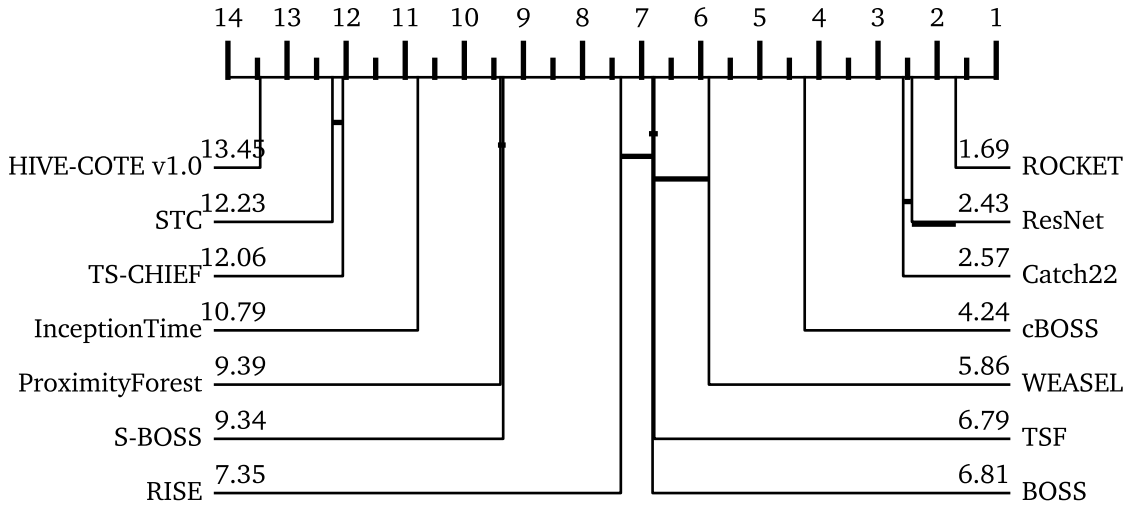


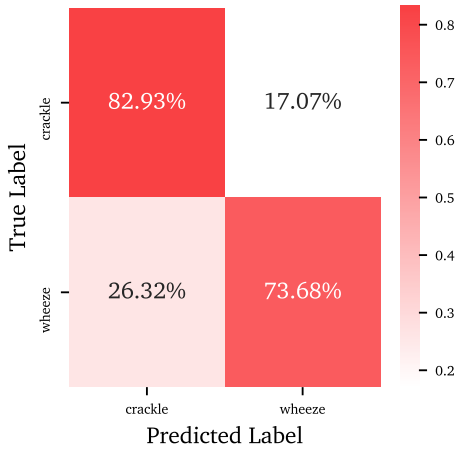
Figure A.2.2: Critical difference plot of 14 classifiers, tested on 112 TSC problems. Solid bars indicate cliques, where there is no significant difference in rank. Ranking is based on time used to train. Higher rank, means shorter computation time.

Table A.2.1: Big O computational complexity of TSC approaches presented in chapter 2.

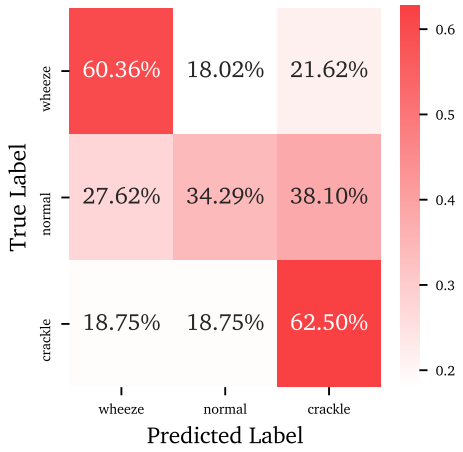
Algorithm	Computational Complexity	References
DTW-KNN	$\mathcal{O}(m^3)$	[27]
EE	$\mathcal{O}(n^2 \times m^2)$	[28]
Shapelet Transform	$\mathcal{O}(n^2 \times m^4)$	[30]
BOSS	$\mathcal{O}(n^2 \times m^2)$	[29]
HIVE-COTE	$\mathcal{O}(n^2 \times m^4)$	[31]
Catch22	$\mathcal{O}(n^{1.16} \times m)$	[5]
ROCKET, MiniROCKET	$\mathcal{O}(m \times n \times n_\omega)$	[37], [38]

A.3 Confusion Matrices TSC

A.3.1 Preproject



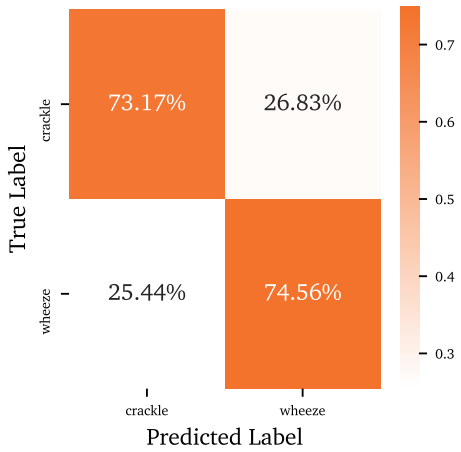
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



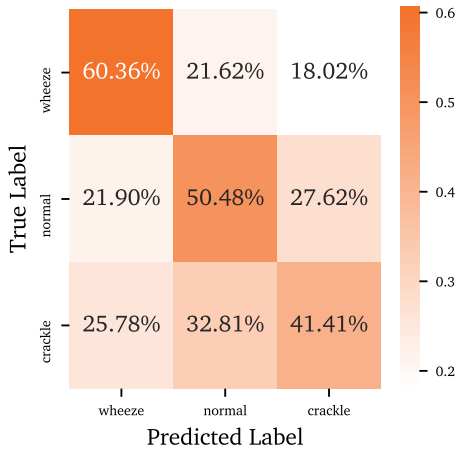
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.1: Confusion matrix after classifying the data extracted, using the preproject feature extraction routine.

A.3.2 Preproject with EEMD



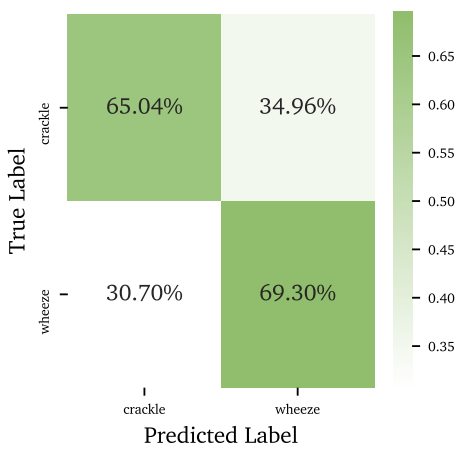
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



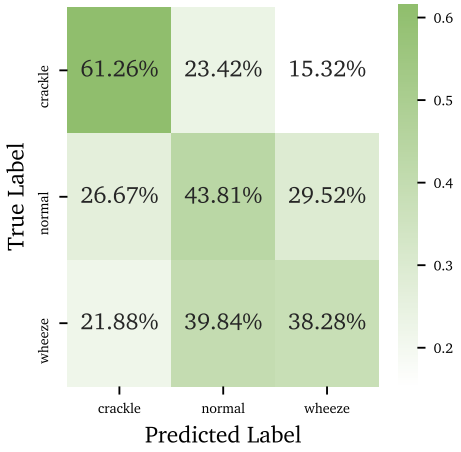
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.2: Confusion matrix after classifying the data extracted, using the preproject feature extraction routine, with EEMD features.

A.3.3 Simple CNN



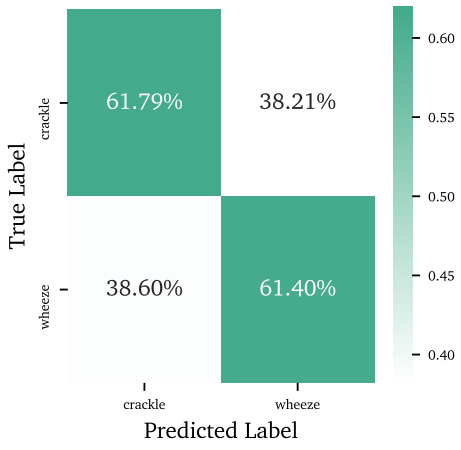
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



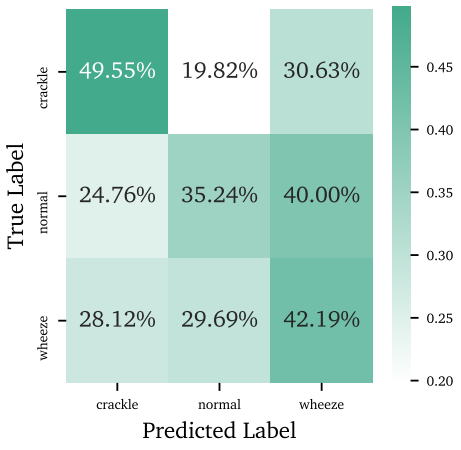
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.3: Confusion matrix after classification using a simple CNN architecture.

A.3.4 TSF



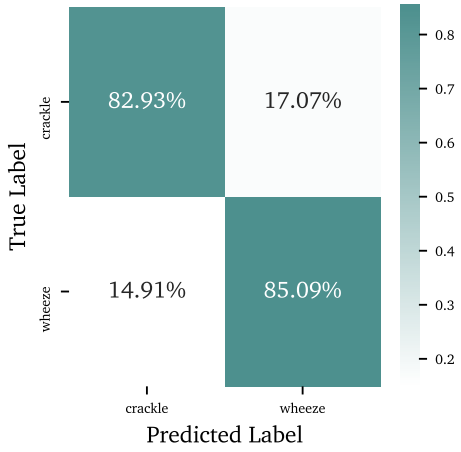
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



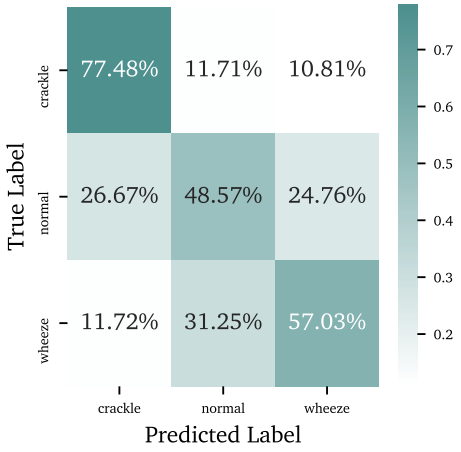
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.4: Confusion matrix after classifying the data extracted, using a Time Series Forest (TSF) for classification.

A.3.5 WEASEL



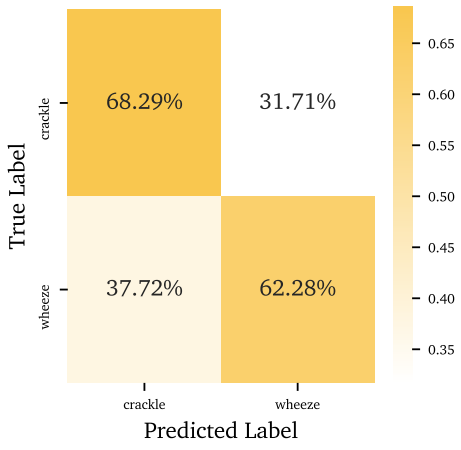
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



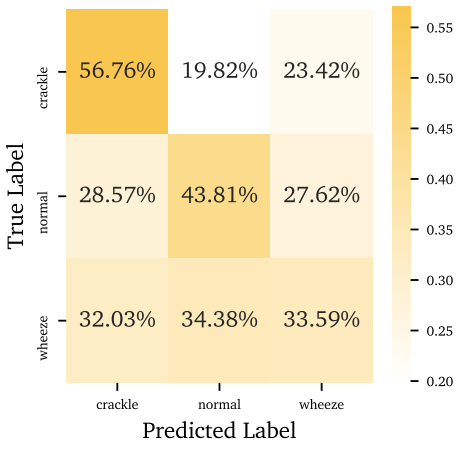
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.5: Confusion matrix after classifying the data extracted, using the WEASEL algorithm for classification.

A.3.6 cBOSS



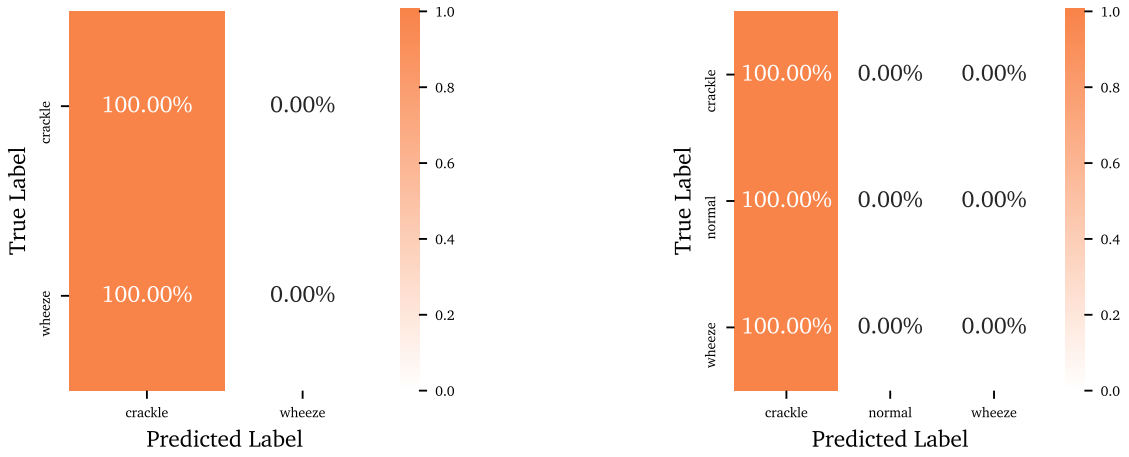
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.6: Confusion matrix after classifying the data extracted, using the cBOSS algorithm for classification.

A.3.7 ResNet

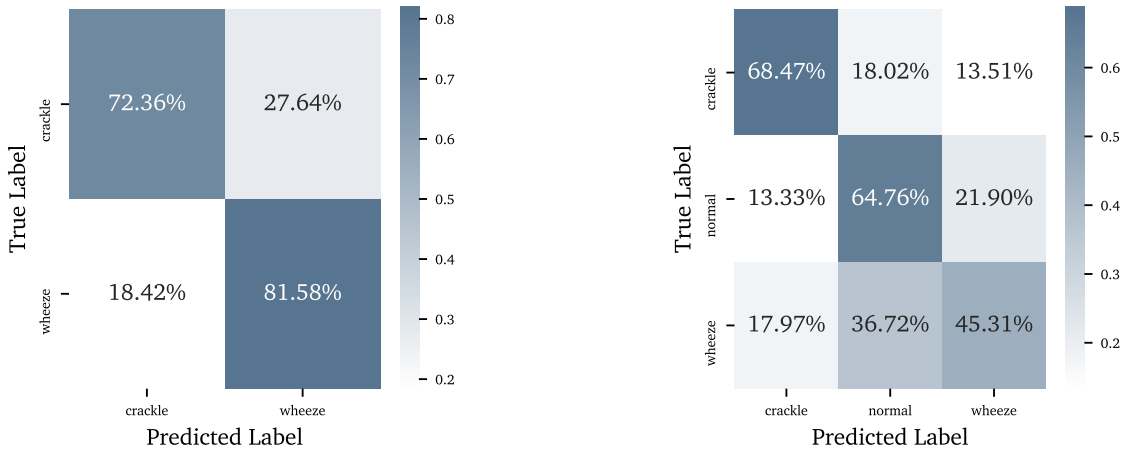


(a) 2 class classification, the dataset presented in section A.1.6 is utilized.

(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.7: Confusion matrix after classifying the data extracted, using the ResNet algorithm for classification.

A.3.8 ROCKET

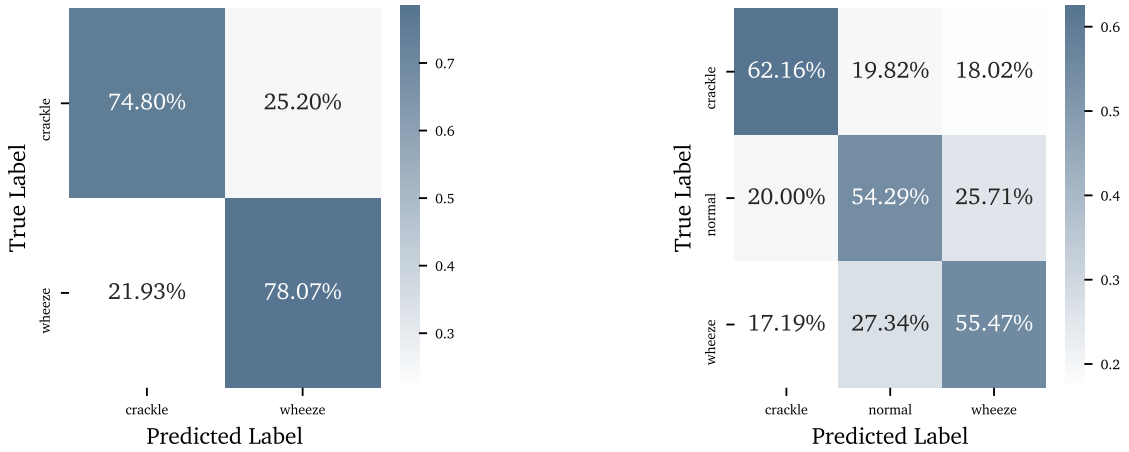


(a) 2 class classification, the dataset presented in section A.1.6 is utilized.

(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.8: Confusion matrix after classifying the data extracted, using the ROCKET algorithm for classification.

A.3.9 MiniROCKET

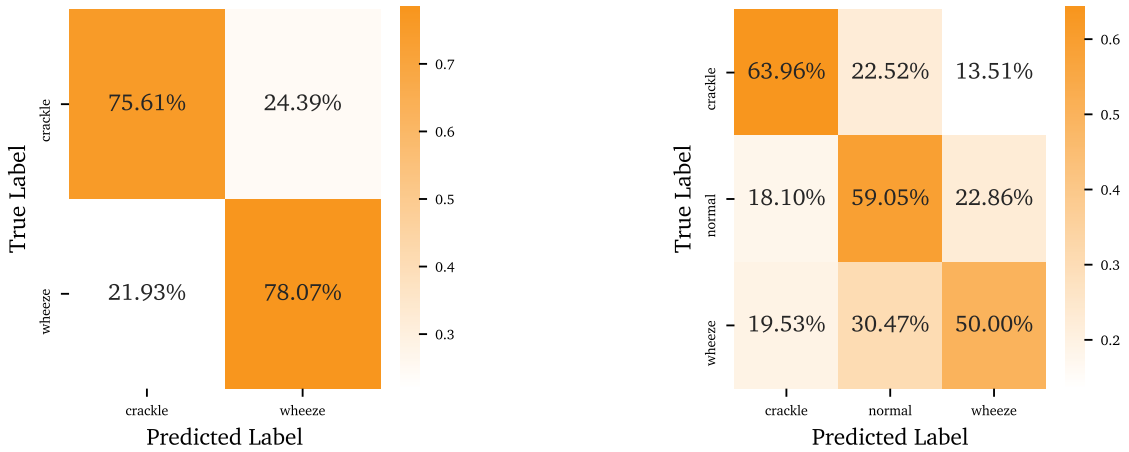


(a) 2 class classification, the dataset presented in section A.1.6 is utilized.

(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.9: Confusion matrix after classifying the data extracted, using the MiniROCKET algorithm for classification.

A.3.10 Catch22

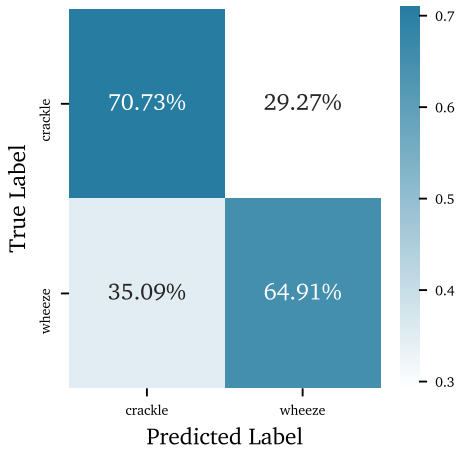


(a) 2 class classification, the dataset presented in section A.1.6 is utilized.

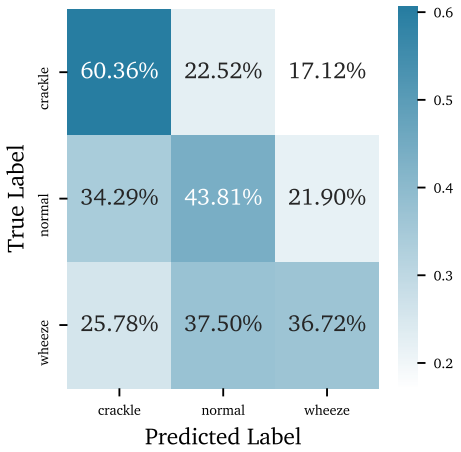
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.10: Confusion matrix after classifying the data extracted, using the Catch22 feature extraction routine

A.3.11 RISE



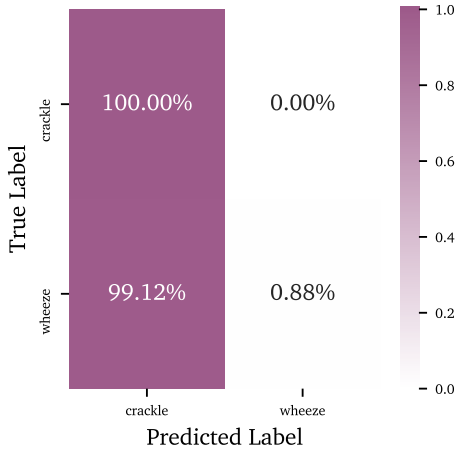
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



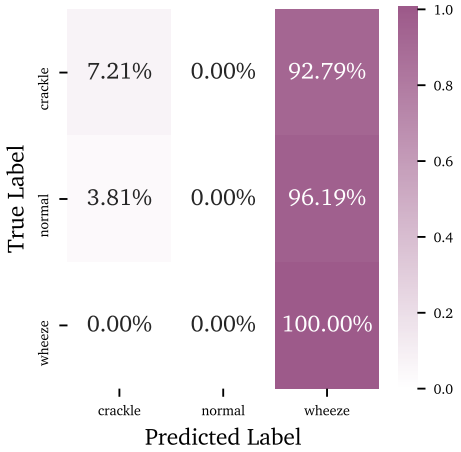
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.11: Confusion matrix after classifying the data extracted, using the RISE algorithm for classification.

A.3.12 InceptionTime



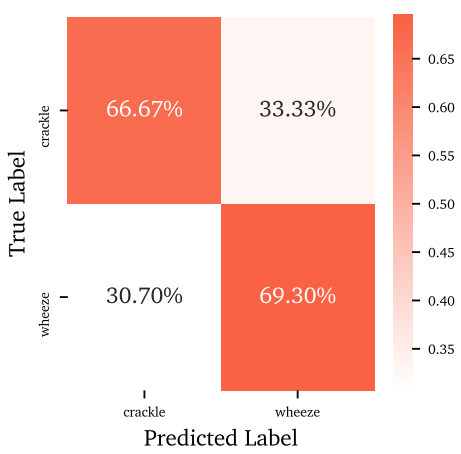
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



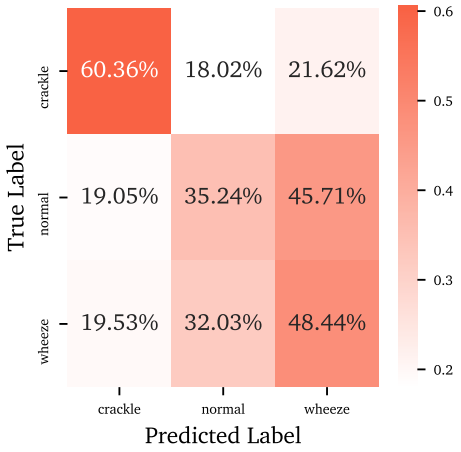
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.12: Confusion matrix after classifying the data extracted, using the InceptionTime algorithm for classification.

A.3.13 CNN with MC dropout



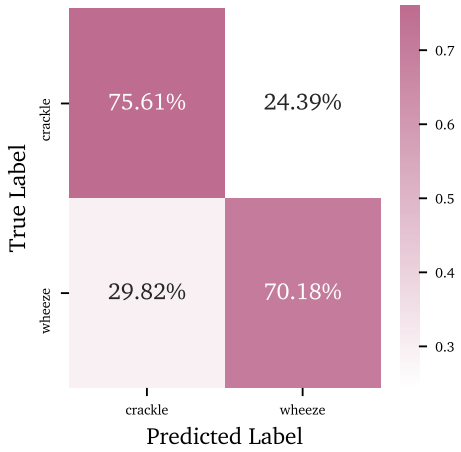
(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



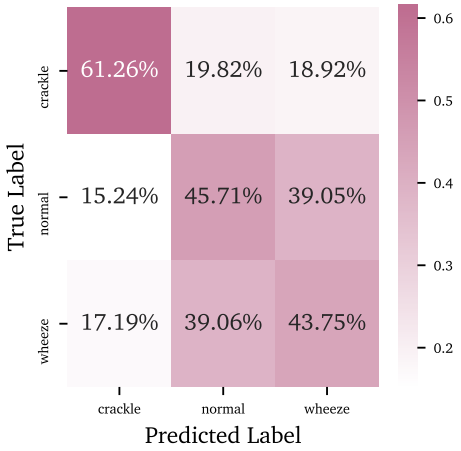
(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.13: Confusion matrix after classifying the data extracted, using a CNN with MC dropout for classification.

A.3.14 Deep Ensemble



(a) 2 class classification, the dataset presented in section A.1.6 is utilized.



(b) 3 class classification, the dataset presented in section A.1.5 is utilized.

Figure A.3.14: Confusion matrix after classifying the data extracted, using Deep Ensemble algorithm for classification.

A.4 GUI for lung sound segmentation

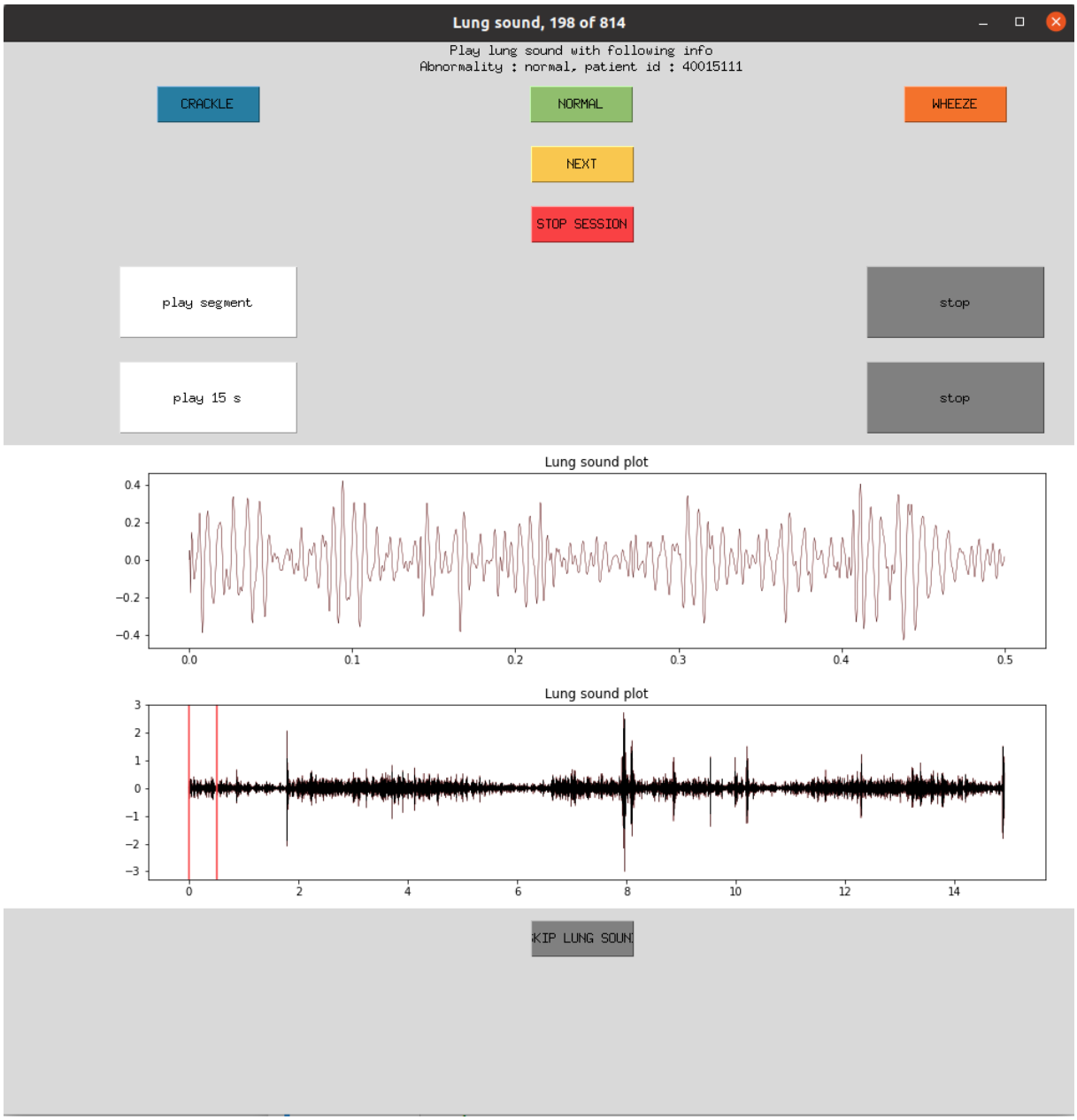
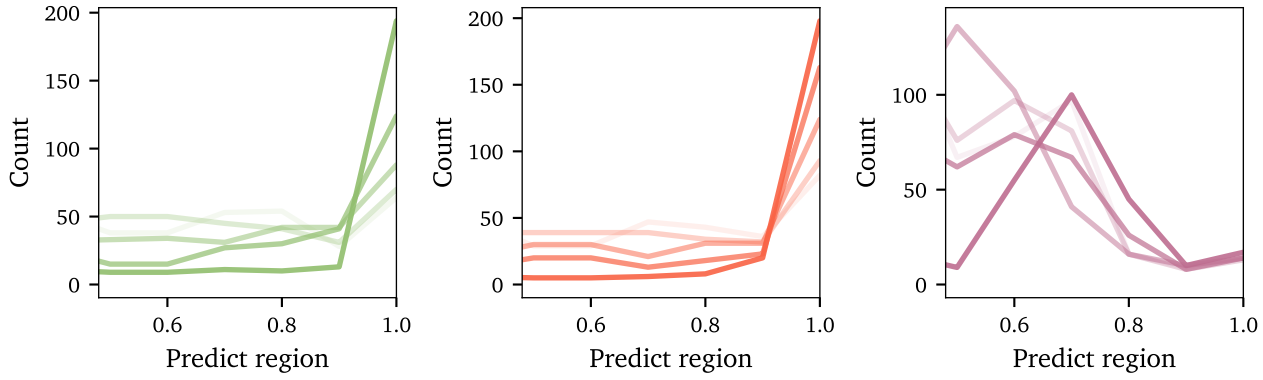


Figure A.4.1: GUI for filtering out short segments of wheeze and crackles

A.5 Count confidence plots, uncertainty quantification

A custom-made plot visualizes the change in confidence as the data is introduced to more manipulation. This plot, named the *count-confidence* plot, groups all predictions into bins, based on confidence, and counts the number of probabilities in each bin. For increasing augmentation, the opaqueness of the curve will increase. Expectantly, the *count-confidence* plot should have greater confidence in the more transparent curves, seeing as the more opaque curves are increasingly corrupted.

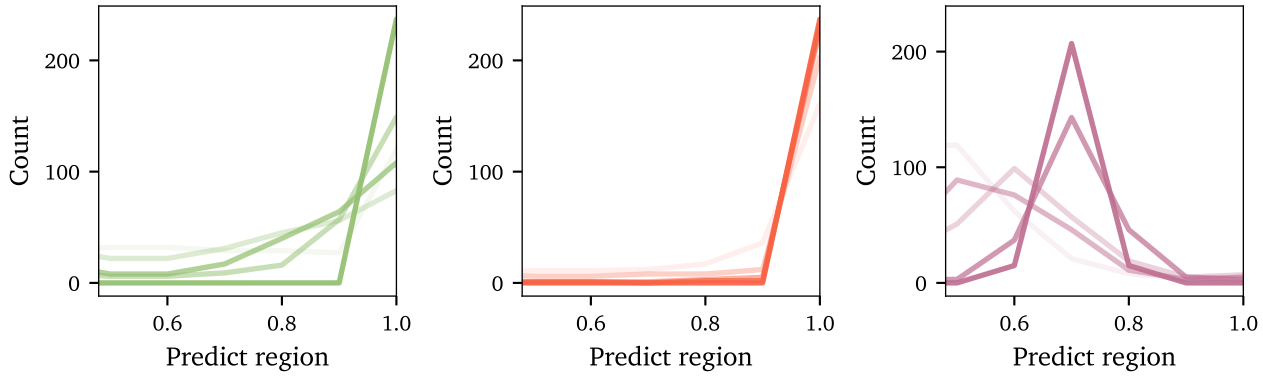
A.5.1 Noise



(a) CNN **(b)** CNN with MC dropout **(c)** Ensemble of CNN's

Figure A.5.1: Comparing the count confidence plots when increasing the amount of *noise* added to the test set. A CNN, CNN with Monte Carlo Dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

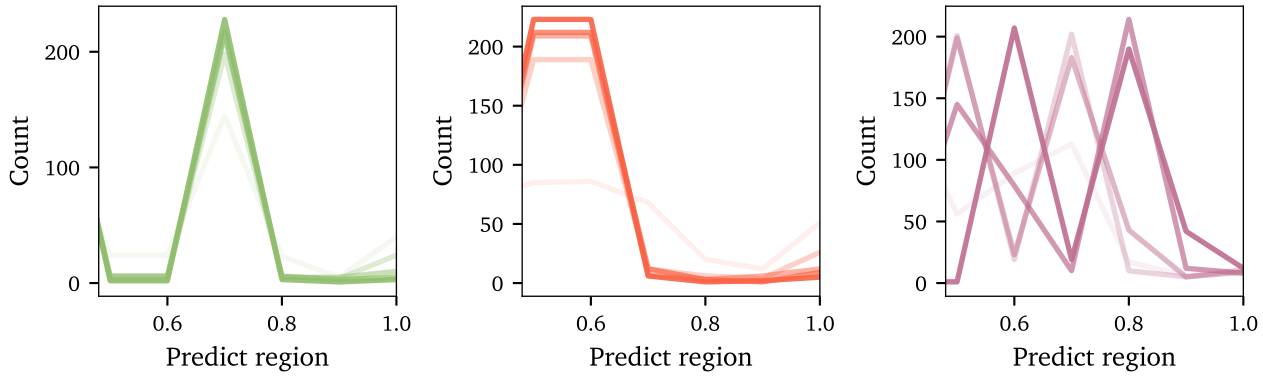
A.5.2 Pitch



(a) CNN **(b)** CNN with MC dropout **(c)** Ensemble of CNN's

Figure A.5.2: Comparing the count confidence plots when increasing the *pitch* of the test set. A CNN, CNN with Monte Carlo Dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

A.5.3 Shift



(a) CNN **(b)** CNN with MC dropout **(c)** Ensemble of CNN's

Figure A.5.3: Comparing the count confidence plots when increasing the *shift* of the test set. A CNN, CNN with Monte Carlo Dropout and an ensemble of CNN's are employed for classification. The dataset presented in section A.1.6 is utilized.

NTNU

Norwegian University of
Science and Technology