UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MECÁNICA

# MODELO PARA IDENTIFICACIÓN DE MODOS DE FALLA DE MÁQUINAS EN BASE A VARIATIONAL AUTO-ENCODERS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MECÁNICO

GABRIEL ANTONIO SAN MARTÍN SILVA

PROFESOR GUÍA:
ENRIQUE LÓPEZ DROGUETT

MIEMBROS DE LA COMISIÓN:
VIVIANA MERUANE NARANJO
SEBASTIÁN MALDONADO ALARCÓN

SANTIAGO DE CHILE
2018

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: GABRIEL ANTONIO SAN MARTÍN SILVA
FECHA: 2018
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

## MODELO PARA IDENTIFICACIÓN DE MODOS DE FALLA DE MÁQUINAS EN BASE A VARIATIONAL AUTO-ENCODERS

Within the field of mechanical engineering, one of the areas that have grown the most in recent years is physical assets management and reliability. Together with the capability of building more complex machines and systems, the problem of early detection of faults in mechanical elements have become of the utmost importance. At the same time, the increasing in the availability and affordability of sensing technology have given engineers the faculty to measure a greater amount of operational variables, such as pressure, temperature or acoustic emissions, to name a few, at sample rates higher than ever. In that regard, it becomes a challenge in itself to process that operational data in an efficient way and extract useful information from it. One approach to tackle this problem is the development of dimensionality reduction techniques, which, if implemented correctly, could provide a better representation of the data in order to perform the diagnosis of health states.

The principal motivation for this thesis is the need for reliable models for the diagnosis of health states in mechanical elements using machine learning techniques. These types of models could result in major benefits for industrial sectors, both in terms of cost savings and operational safety.

The main objective of this thesis is to develop a model for diagnosis of health states in mechanical elements based on a dimensionality reduction approach using Variational Auto-Encoders (VAEs), and then evaluate and compare the results against a similar model that use principal component analysis (PCA) as a reduction method and a third model that does not perform a reduction of the databases at all.

The methodology used for this work consists mainly of five consecutive steps: First, a revision of the state of the art with respect to methodologies developed for diagnosis is presented. Second, the acquirement and preprocessing of the databases needed for the training and testing of the proposed models. Third, the model using PCA and the model with no reduction are implemented to create a baseline for comparison. Fourth, the model based on Variational Auto-Encoders is developed and implemented. Ultimately, the VAE model developed in this thesis is compared with the other two methods to draw conclusions about its applicability.

The principal conclusion of this thesis is that the VAE model results in better health states classification accuracies than the PCA based reduction for situations where the amount of labeled data is scarce, or when the reduction in dimensionality has to be very drastic. Also, the VAE model almost always outperforms the model where no reduction is done to the datasets, highlighting the utility of performing a reduction in dimensionality to the data prior the classification tasks.

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MECÁNICO
POR: GABRIEL ANTONIO SAN MARTÍN SILVA
FECHA: 2018
PROF. GUÍA: ENRIQUE LÓPEZ DROGUETT

## MODELO PARA IDENTIFICACIÓN DE MODOS DE FALLA DE MÁQUINAS EN BASE A VARIATIONAL AUTO-ENCODERS

Dentro del campo de la ingeniería mecánica, una de las áreas que más crecimiento ha mostrado en los últimos años es la de la gestión de activos físicos y confiabilidad. Junto con la capacidad de construir máquinas y sistemas más complejos, el problema de la detección temprana de fallas en elementos mecánicos se vuelve de suma importancia. Al mismo tiempo, el incremento en la disponibilidad de tecnología sensitoria ha dado a los ingenieros la capacidad de medir una gran cantidad de variables operacionales, como por ejemplo presión, temperatura o emisiones acústicas, a frecuencias de muestreo altísimas. Es ese aspecto, se vuelve un desafío en sí mismo el poder procesar esa cantidad de datos de una manera eficiente, con tal de extrar información útil a partir de ellos. Una metodología para enfrentar este problema es el desarrollo de técnicas de reducción de dimensionalidad, las cuales, si son implementadas de forma correcta, pueden generar una mejor representación de los datos con el fin de mejorar el diagnóstico posterior de los modos de falla presentes.

La motivación principal de este trabajo de título es la necesidad de desarrollar modelos confiables para el diagnóstico de modos de falla en elementos mecánicos utilizando técnicas de Aprendizaje de Máquinas. Estos modelos pueden resultar en grandes beneficios para los sectores industriales, tanto en términos de ahorros monetarios como seguridad operacional.

El principal objetivo de esta tesis es desarrollar modelos para el diagnóstico de fallas en elementos mecánicos basados en una reducción de dimensionalidad usando un Auto Encoder Variacional (VAE), y luego evaluar y comparar los resultados obtenidos con un modelo similar que usa Análisis de Componentes Principales (PCA) como método de reducción de dimensionalidad y un tercer modelo que no genera una reducción.

La metodología usada para este trabajo consiste principalmente de cinco etapas. Primero, una revisión del estado del arte respecto a metodologías existentes para el diagnóstico de fallas es desarrollada. Luego, la adquisición y preprocesamiento de datos operacionales que serán utilizados para entrenar y evaluar los modelos desarrollados. Tercero, el modelo que usa PCA y el modelo que no realiza reducción de dimensionalidad es implementado. Cuarto, el modelo que utiliza VAE es desarrollado e implementado. Por último, el modelo que usa VAE es comparado con los otros dos modelos para extraer conclusiones sobre su aplicabilidad.

La principal conclusión de este trabajo es que el modelo que utiliza VAE es mejor en el diagnóstico de modos de falla que el que utiliza PCA para situaciones donde la cantidad de datos etiquetados es escasa, o para los casos cuando una reducción de dimensionalidad muy drástica es requerida. Tambien, el modelo que utiliza VAE casi siempre presenta mejores resultados que el modelo que no genera reducción en los datos, mostrando la importancia de reducir la dimensionalidad de los datos previo a una operación de diagnóstico o clasificación.

*To Sandra, for all the patience, kindness and for being such a positive role model in my life.*
*We will meet again someday.*
*And to Laura and Raphael, for all the happiness that await ahead of us.*

# Acknowledgments

First of all, I wish to express my sincere thanks to my thesis advisor, Prof. Enrique López, who's office door was always (really, always) open to solve all my questions, guide me through this work and help me to get into the amazing (and sometimes scary) world of research and investigation. It was an incredible journey, one that I really enjoyed. Thank you very much.

I would also like to acknowledge all the efforts that my family have done for me in order to make this milestone possible. My mother, who's love, kindness and honesty was always there to encourage me to give my best. My father, who somewhat manage to be always present in my life, teaching me valuable lessons and giving me all his support even though what life could throw at us. My brother Raphael, who's silence is always a reminder of hard work and consequence to me, and who's laugh is extremely rewarding every time I manage to produce it. My sister Laura, who's existence was the most precious gift that life could have given me. My grandparents Jorge and Norma, because their unconditional love against all odds is an example of what I want for my life. And last, but not least important, both Verónica (my step mom) and Rubén (my step dad). Life, not happy with giving me two amazing parent, gave me another extra couple of amazing parental figures to learn from.

Also, I would like to thank all my closest friends: Walter, Sebastián, Jean, Andrés, Danilo, Ricardo, Vicente and Francisco. Your continuous jokes about my personality forged me into a (I hope) better person. Thanks for that.

And finally, to Mirella, Sandra, Nora and Sofía. I know that you are together now enjoying non-filtered cigarettes, puzzles, fashion magazines and soft cushions. Thanks for raising me, loving me and making my infancy a happy experience. You are the best.

# Contents

**List of Tables** **viii**

**List of Figures** **x**

**1 Introduction** **1**
1.1 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
1.2 General Objective . . . . . . . . . . . . . . . . . . . . . . . . . . 3
1.3 Specific Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . 3
1.4 Scope of This Work . . . . . . . . . . . . . . . . . . . . . . . . . 4

**2 Methodology** **5**
2.1 Data Acquisition . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
2.2 Gathering Information . . . . . . . . . . . . . . . . . . . . . . . . 5
2.3 Development of the Models for the Diagnosis of Health States . . . . . . . 6
2.4 Definition and Execution of Experiments . . . . . . . . . . . . . . . 6
2.5 Analysis of the Results and Concluding Remarks. . . . . . . . . . . . . 6
2.6 Resources Used for this Thesis . . . . . . . . . . . . . . . . . . . . 7

**3 Theoretical Background** **8**
3.1 Short-Time Fourier Transform (STFT) and the Spectrogram . . . . . . . . 8
3.2 Principal Component Analisys . . . . . . . . . . . . . . . . . . . . 10
3.3 Machine Learning . . . . . . . . . . . . . . . . . . . . . . . . . . 11
3.4 Classification Problems . . . . . . . . . . . . . . . . . . . . . . . 12
3.5 Artificial Neural Networks . . . . . . . . . . . . . . . . . . . . . . 13
3.5.1 Activation Functions for the Intermediate Layers . . . . . . . . . . . 15
3.5.2 Activation Function for the Output Layer . . . . . . . . . . . . . . 17
3.5.3 Training of Artificial Neural Networks . . . . . . . . . . . . . . . 17
3.5.4 Convolutional Neural Networks . . . . . . . . . . . . . . . . . . 18
3.6 Variational Inference . . . . . . . . . . . . . . . . . . . . . . . . 19
3.7 Variational Auto-Encoders . . . . . . . . . . . . . . . . . . . . . . 22
3.7.1 Prior over the latent variables $p(z)$ . . . . . . . . . . . . . . . . 23
3.7.2 The original data conditioned by the latent variables $p(x|z)$ . . . . . . 24
3.7.3 The approximate posterior $q(z|x)$ . . . . . . . . . . . . . . . . . 24
3.7.4 KL Divergence $KL(q(z|x)||p(z))$ . . . . . . . . . . . . . . . . . . 25
3.7.5 Expectancy over the logarithm of $p(x|z)$ . . . . . . . . . . . . . . 26
3.7.6 Putting the pieces together . . . . . . . . . . . . . . . . . . . . 26

**4 The Proposed Approach for Dimensionality Reduction in Health States Diagnosis 29**

**5 Proposed Architectures for the VAE model 32**
5.1 Ball Bearing Datasets . . . 32
5.1.1 Case Western Reserve University Bearing Data Center (CWR) . . . 32
5.1.2 Machinery Failure Prevention Technology Society (MFPT) . . . 33
5.2 Data Preprocessing and Data Types . . . 35
5.2.1 STFT and Spectrograms . . . 35
5.2.2 Manually Extracted Features . . . 35
5.2.3 Raw Vibration Signal . . . 36
5.3 Proposed Architectures for the VAE's Encoder and Decoder . . . 37

**6 Validation Procedure 42**

**7 Case Studies Results 45**
7.1 Spectrograms Images. . . . 45
7.1.1 Case Study #1: CWR . . . 45
7.1.2 Case Study #2: MFPT . . . 50
7.2 Manually Extracted Features. . . . 54
7.2.1 Case Study #1: CWR . . . 54
7.2.2 Case Study #2: MFPT . . . 58
7.3 Raw Vibration Signal. . . . 62
7.4 CNN Variational Auto Encoder . . . 63

**8 Concluding Remarks 67**

**9 Bibliography 69**

**A List of Manually Extracted Features 73**

**B Tables of Complete Results 75**

**C Raw Vibration Signal Dataset Plots 80**

# List of Tables

5.1 Classes for the CWR dataset. . . . . . . . . . . . . . . . . . . . . . . . . . 33
5.2 Classes for the MFPT dataset. . . . . . . . . . . . . . . . . . . . . . . . . . 34
5.3 Architectures tested for the VAE model. . . . . . . . . . . . . . . . . . . . 37
5.4 Final average accuracy for each architecture tested using spectrogram images. 38
5.5 Final average accuracy for each architecture tested using features extracted manually. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
5.6 Final average accuracy for each architecture tested using portions of the original vibration signal without any kind of preprocessing. . . . . . . . . . . . 39

6.1 Values for $k$, the number of dimensions of the latent space, and $\varepsilon\%$, the percentage of labeled data. . . . . . . . . . . . . . . . . . . . . . . . . . . 43
6.2 Architecture details for the three different classifiers tested. HL stands for Hidden Layer. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 43

7.1 Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the CWR datasets are preprocessing applying the STFT and then scaling down the resulting spectrograms to images of 96 by 96 pixels. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
7.2 Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the MFPT datasets are preprocessing applying the STFT and then scaling down the resulting spectrograms to images of 96 by 96 pixels. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50
7.3 Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the CWR dataset is preprocessing doing a manual feature extraction. . . . . . . . . . . . . . . . . . . . . . . . . . 54
7.4 Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the MFPT dataset is preprocessing doing a manual feature extraction. . . . . . . . . . . . . . . . . . . . . . . . . . 58
7.5 Best accuracy results and the conditions (classifier and dimension) under they were obtained when the raw data from the CWR and MFPT datasets is used without any kind of preprocessing. . . . . . . . . . . . . . . . . . . . . . 62
7.6 Training times per epoch and total for the VAE model shown in section 5.3 and the CNN-VAE model with both architectures tested. The training of all architectures is performed with a maximum of 500 epochs. . . . . . . . . . . 65
7.7 Better fault diagnosis accuracies and standard deviations based on the CNN-VAE architecture #2 for the CWR and MFPT datasets containing spectrograms. 66

A.1 List of hand-engineered features used as a preprocessing methodology for the generation of datasets. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 74

B.1 Average accuracies and standard deviations for CWR and MFPT case studies based on spectrograms. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 76
B.2 Average accuracies and standard deviations for CWR and MFPT case studies based on manually extracted features. . . . . . . . . . . . . . . . . . . . . . . . 77
B.3 Average accuracies and standard deviations for CWR and MFPT case studies based on chunks of the raw vibration signal. . . . . . . . . . . . . . . . . . . . 78
B.4 Average accuracies and standard deviations for CWR and MFPT case studies based spectrograms and the usage of the CNN-VAE model. . . . . . . . . . . 79

# List of Figures

3.1 Spectrogram of an undamaged ball bearing element signal. . . . . . . . . . . 10
3.2 Graphical representation of a neuron unit and a neural network. . . . . . . . 14
3.3 Activation functions for the intermediate layers of the ANN. . . . . . . . . . 16
3.4 Variational Auto-Encoder Models . . . . . . . . . . . . . . . . . . . . . . . 27
3.5 VAE model with the reparametrization trick applied. . . . . . . . . . . . . . 28

4.1 Proposed Approach for the VAE based dimensionality reduction. . . . . . . . 30

5.1 Amplitude-time signals for the baseline and the three fault locations present in the CWR dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
5.2 Amplitude-time signals from the three health states present in the MFPT dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35
5.3 Diagram showing the procedure of generating vector from the images of spectrograms. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36
5.4 VAE's Encoder Neural Network . . . . . . . . . . . . . . . . . . . . . . . . 40
5.5 VAE's Decoder Neural Network . . . . . . . . . . . . . . . . . . . . . . . . 40
5.6 Transformation model using only the encoder of the trained VAE. . . . . . . 41

7.1 Average accuracy versus latent space dimension for the CWR dataset and MLP1L classifier with spectrograms. . . . . . . . . . . . . . . . . . . . . . 47
7.2 Average accuracy versus latent space dimension for the CWR dataset and MLP1LDO classifier with spectrograms. . . . . . . . . . . . . . . . . . . . 48
7.3 Average accuracy versus latent space dimension for the CWR dataset and MLP3LDO classifier with spectrograms. . . . . . . . . . . . . . . . . . . . 49
7.4 Average accuracy versus latent space dimension for the MFPT dataset and MLP1L classifier with spectrograms. . . . . . . . . . . . . . . . . . . . . . 51
7.5 Average accuracy versus latent space dimension for the MFPT dataset and MLP1LDO classifier with spectrograms. . . . . . . . . . . . . . . . . . . . 52
7.6 Average accuracy versus latent space dimension for the MFPT dataset and MLP3LDO classifier with spectrograms. . . . . . . . . . . . . . . . . . . . 53
7.7 Average accuracy versus latent space dimension for the CWR dataset and MLP1L classifier with manually extracted features. . . . . . . . . . . . . . . 55
7.8 Average accuracy versus latent space dimension for the CWR dataset and MLP1LDO classifier with manually extracted features. . . . . . . . . . . . 56
7.9 Average accuracy versus latent space dimension for the CWR dataset and MLP3LDO classifier with manually extracted features. . . . . . . . . . . . 57

7.10 Average accuracy versus latent space dimension for the MFPT dataset and MLP1L classifier with manually extracted features. . . . . . . . . . . . . . . . 59
7.11 Average accuracy versus latent space dimension for the MFPT dataset and MLP1LDO classifier with manually extracted features. . . . . . . . . . . . . . 60
7.12 Average accuracy versus latent space dimension for the MFPT dataset and MLP3LDO classifier with manually extracted features. . . . . . . . . . . . . . 61
7.13 Reconstruction errors of the deep neural network VAE model discussed in Section 5.3 and the CNN-VAE model with both proposed architectures and for the (a) CWR dataset and (b) MFPT dataset. . . . . . . . . . . . . . . . . 64
7.14 CNN VAE's Encoder . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65
7.15 CNN VAE's Decoder . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 65

C.1 Average accuracy versus latent space dimension for the CWR dataset and MLP1L classifier with chunks of the original vibration signal. . . . . . . . . . 81
C.2 Average accuracy versus latent space dimension for the CWR dataset and MLP1LDO classifier with chunks of the original vibration signal. . . . . . . . 82
C.3 Average accuracy versus latent space dimension for the CWR dataset and MLP3LDO classifier with chunks of the original vibration signal. . . . . . . . 83
C.4 Average accuracy versus latent space dimension for the MFPT dataset and MLP1L classifier with chunks of the original vibration signal. . . . . . . . . . 84
C.5 Average accuracy versus latent space dimension for the MFPT dataset and MLP1LDO classifier with chunks of the original vibration signal. . . . . . . . 85
C.6 Average accuracy versus latent space dimension for the MFPT dataset and MLP3LDO classifier with chunks of the original vibration signal. . . . . . . . 86

# Chapter 1

# Introduction

One of the main challenges faced by many industrial sectors nowadays is the early detection of faults in machinery. Usually, fault related problems are treated in one of two ways. Either the component is replaced subjected to the manufacturer instructions following certain criteria about time of operation or number of cycles, or the fault presents itself in the form of a performance drop, or even worse, a sudden failure leading to the complete stop of the machine. The development of an early, nondestructive method to identify faults before they cause a stop in production but that also take advantage of all the possible operation time of the element in question is an active field of research in reliability [1] [2].

One approach to develop a methodology for the early detection of faults is to assume that the operational data have information about the health state of the system at the moment of its measure and, from there, use some technique to extract such information in a clear and clever way [3]. A technique that has been used extensively in the past to extract information from complex data is Machine Learning [4] [5].

In recent years, sensors have become cheaper and readily available for the mainstream industry. With this increasing accessibility to sensing technology, engineers can now take measures of a larger variety of variables, such as pressure, temperature, acoustic emissions and vibrations, with greater sampling frequency than before. This results in massive amounts of operational and maintenance data that can be gathered easily, and that have to be analyzed in order to be useful for diagnosis, prognosis and health management purposes [6]. Machine Learning, as stated before, is nowadays a popular strategy to perform the data analysis in an efficient way.

Nevertheless, one common problem when applying machine learning techniques to data acquired by sensors in the industry is the curse of dimensionality. By nature, operational as well as maintenance data is high dimensional and relatively noisy. Traditional machine learning methods might not perform well when the input data have a higher dimensional representation [7] [8]. Thus, a common approach to solve this problem is to perform a dimensionality reduction to the data before feeding it to a classification algorithm to make it more manageable, such as with Artificial Neural Networks (ANN) [9], Support Vector Machines [10] and even Decision Trees [11], to name a few.

Among the approaches for dimensionality reduction, Principal Component Analysis (PCA) [12] [13], and its variants [14] [15] [16], is one of the most popular ones in the fault diagnosis community [10] [11]. Moreover, other techniques have also been developed with the intention to generate a better reduced representation of the data. One such prominent technique encompasses Auto Encoders (AE) [17], which are unsupervised models based on neural networks. AEs reduce the original data into a lower dimensional representation formed by its smallest hidden layer. Traditional Auto Encoders and its variants have been used in recent years for fault diagnosis purposes. In particular, AEs were used in a deep neural network scheme to perform multimode fault classification [8] and also in the context of rolling bearing fault diagnosis [1]. Denoising Auto Encoders (DAE), which add noise to the original data in an attempt of helping the model to learn a better representation, were used in fault classification of rotating machinery [18]. While traditional and denoising Auto Encoders have relatively fewer neurons in their hidden layers, Sparse Auto Encoders (SAE) have a larger number of units in the network to compress the data, but parameters are applied to force the model to produce a near zero output in most of the neurons. SAEs were used for health monitoring and prognostics of machine bearings [2].

Another approach that share the concept of compressing the data by passing it through a hidden representation are Variational Auto Encoders (VAE) [19]. Nowadays, VAE are one of the most promising techniques for unsupervised learning and has seen many recent success cases in image processing [19] [20] [21], text generation [22] and speech recognition [23]. VAEs are a very auspicious approach because they combine Variational Inference (VI) with the use of neural networks as functions approximators such that the search for the approximate posterior distribution can be performed with stochastic gradient descent [24]. They differ from traditional AE, SAE and DAE by imposing a distribution over the latent variables and the data itself. Because of this, VAEs can generate new data by simply sampling from this distribution once the model has been trained. VAEs accomplish this by generating a latent representation of the data that can be chosen to be of a lower dimensionality that the data itself. This representation created by the VAE can be interpreted as a compressed characterization of the dataset.

This thesis proposes a novel approach for dimensionality reduction based on deep Variational Auto Encoders for fault diagnosis. Starting with a VAE with customized architecture for its encoder and decoder neural networks, the proposed VAE processes in a fully unsupervised way vibration sensor fault data in the form of raw signals, vector of extracted features (e.g., original, derivative and integral of the signal) and time-frequency image representations (e.g., spectrograms). Then, the encoder output's latent space is treated as a feature map, which is used to transform the dataset into a lower dimensional representation. In order to evaluate the robustness of this reduction, the reduced dataset is then fed to three different neural network based classifiers, to perform a diagnosis of the health states present in each original data sample. The results of these classification tasks are then compared to the ones produced by a second model that uses PCA instead of VAEs to perform the reduction in dimensionality and a third model that fed directly the original database (raw signal, extracted features or spectrograms) into the classifiers without performing a prior reduction in its dimensionality.

In particular, the use of spectrograms generated from vibration signals is explored because

VAEs have shown robust results when processing and generating images in many contexts such as handwritten digits [19] [20] , faces [21] and images from the CIFAR database [25]. From an engineering perspective, spectrograms provide the means of extracting time and frequency information from the original vibration signal instead of just time information, which is the case if one feeds the raw signal into the proposed approach. Moreover, two models for the encoder and decoder are investigated: deep feed forward neural networks and deep convolutional neural networks. To illustrate the flexibility and performance of the VAE based approach, two examples of applications involving fault diagnosis of ball bearings are presented, where different architectures of neural networks fault classifiers are fed with both the proposed VAE's and with PCA's data representation.

In the following, the motivation, general objective and specific objectives of the thesis are declared, as well as the scope of this work.

## 1.1 Motivation

The need for reliable models that can perform diagnosis of health states in mechanical elements under difficult conditions such as data contaminated with operational noise, or the lack of labeled examples to train the models is still an open problem among the reliability and maintainability community. Well performed diagnosis could potentially yield to significantly drops in costs related to the premature replacement of parts and equipment or the unexpected failure of machinery critical for production. Because of that, the principal motivation of this thesis is the development of a novel model for diagnosis of health states based on the dimensionality reduction performed by Variational Auto-Encoders, and the posterior evaluation of its advantages and disadvantages against existing methods, so that useful information for future work could be extracted from it and hopefully guide future research.

## 1.2 General Objective

Develop a model for the diagnosis of health states using a dimensionality reduction approach based on Variational Auto-Encoders and perform an evaluation of the situations under which it represents an advantage over two other tested models: one that uses PCA based reduction and a third one that do not perform reduction in the dimensionality of the databases.

## 1.3 Specific Objectives

1. Develop a model for the diagnosis of health states in mechanical elements using a dimensionality reduction approach based on Variational Auto-Encoders.

2. Measure the accuracy obtained in the classification of health states in ball-bearing

elements using the model developed for diagnosis and vibration data acquired by accelerometers.

3. Compare the results of the developed model with the the cases where PCA is used as method for dimensionality reduction and where no reduction is performed at all.

4. Draw conclusions about the applicability of the develop model and under which circumstances it presents advantages over the use of the other two methods.

## 1.4 Scope of This Work

This thesis consists in the development of a model for the diagnosis of health states in ball bearing elements, using the dimensionality reduction provided by the Variational Auto-Encoder models.

# Chapter 2

# Methodology

For this thesis, a methodology that allows the development of a model for dimensionality reduction based on the use of Variational Auto-Encoders and the posterior comparison of such model with the well-known method *Principal Component Analysis* and a third model where no reduction in dimensionality is perform to the datasets has to be follow. The proposed methodology is described below.

## 2.1 Data Acquisition

The model that will be developed in this thesis is based on a machine learning model named Variational Auto-Encoder. Since machine learning techniques are built on top of the concept of learning the underlying relationships between different examples that represent the phenomenon of interest, the availability of a database containing those examples is indispensable. Because of this, the first step of this thesis is the acquisition of operational databases of some mechanical element, to test and validate the proposed model. Ball-bearing elements, as one of the most used and important element in almost every rotating machine, counts with datasets available for free in on-line repositories.

For this thesis, two different databases will be used. The first one belongs to the Case Western Reserve University (CWR) Bearing Data Center. The second one belongs to the society for Machinery Failure Prevention Technology (MFPT). Both datasets consists in vibration data measure with accelerometers during the operation of the ball bearings.

## 2.2 Gathering Information

After the acquisition of suitable datasets is accomplished, the next step is to gather information about the different techniques, tools and models that already exists in the contexts of reliability, machine learning and diagnosis. In particular, this step was almost fully developed in the previous course (ME6908) during the Autumn of 2017. There are four clearly defined areas of knowledge that were researched:

- Signals and data processing, to being able to add noise to the databases if necessary and to extract time-frequency domain information and features from the vibration signals.
- Machine Learning in general, mostly to apply neural networks as classifiers of health states in the classification model and function approximators in the Variational Auto-Encoder model.
- Principal Component Analysis, because it is the most common approach for performing dimensionality reductions in the data science community and it will be used as a comparison method against the model proposed in this thesis.
- Variational Auto-Encoders, since they are the main focus of this thesis. In particular, the use of them as a dimensionality reduction method was studied extensively.

## 2.3 Development of the Models for the Diagnosis of Health States

Once enough information has been gathered, the development of the PCA based model, the VAE based model and the model that do not perform a reduction in dimensionality is executed. For this purpose, computational resources like Python and their scientific repositories, among machine learning libraries like TensorFlow are used.

## 2.4 Definition and Execution of Experiments

To evaluate the VAE based model developed in this thesis, experiments portraying specific situations of interest must be first defined and then performed. Because of this, the creation of such experiments is the next step. Once they are executed, the results of all of them are stored for the posterior analysis.

## 2.5 Analysis of the Results and Concluding Remarks.

Once all the experiments are concluded and the results are gathered, they will be analyzed with the purpose of evaluate in which situations the proposed model showed better performance and applicability. From this analysis, the concluding remarks of this thesis will be drawn.

## 2.6 Resources Used for this Thesis

As in any work that requires experiments, resources are needed. Below, a comprehensive list of the most important resources is shown.

- Databases for the training of the models.
- Computational hardware compatible with TensorFlow and CUDA. In particular, for this thesis a computer with Ubuntu 16.04, an i7-7700k processor, 32 Gb of RAM and a Titan X GPU (Nvidia) was used. This hardware was facilitated by the SRMI Lab.
- Python as a programming language, Scipy and Numpy as scientific libraries, TensorFlow 1.2.0 and CUDNN 8.0 as the machine learning framework.

# Chapter 3

# Theoretical Background

In this chapter, the theoretical background underlying this thesis is presented to the reader. At first, a description on the Short-Time Fourier Transform (STFT) and its graphical representation, the spectrogram, is discussed. Then, Principal Component Analysis (PCA), arguably the most known method for performing a dimensionality reduction, is explained. Afterwards, a brief introduction to Machine Learning, and in particular, the types of classification problems that exists within the area are presented to contextualize the reader into the topic of the diagnosis of health states. Next, the discussion is focus on Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), both algorithms of Machine Learning that have shown a considerable amount of success in classification tasks within the areas of computational vision, speech recognition, natural language processing and even reliability and condition based maintenance (CBM). Then, variational inference (VI), a powerful approach to approximate difficult-to-compute probability densities and the core component of Variational Auto-Encoders is introduced. Finally, this chapter present the Variational Auto-Encoder model to the reader.

## 3.1 Short-Time Fourier Transform (STFT) and the Spectrogram

STFT includes a time variable to the well-known Fourier spectrum. This allows to investigate the signal from a time-varying perspective. In this methodology, a short time window slice along the original signal, extracting only a part of it in each step. Since the length of the time window is very small, the segmented signal should not vary too much and hence, it can be treated as a stationary signal. Then, the Fourier transform is applied, extracting the local Fourier spectrum of each segment, revealing detailed frequency information of the original signal although its nature could be far from stationary [26].

The short time Fourier transform of a signal $x(t)$ is obtained as follows. First, a window function $w(t-\tau)$ is defined. This function is centered at time $\tau$ ($\tau$ is a time variable). Then, through this window, the signal function that is observed can be expressed as $x(t)w(t-\tau)$.

Then, the Fourier transform is applied to this segmented portion of the signal, leading to the expression shown in equation 3.1. Finally, the window slides a certain distance and the Fourier transform is applied again until the totality of the signal has been transformed.

$$STFT(\tau, f) = \int_{-\infty}^{+\infty} x(t)w(t-\tau)\exp(-j2\pi ft)\mathrm{d}t \tag{3.1}$$

In industrial sectors, usually the signals are captured with digital sensors which outputs a series of discrete measurements. When that is the case, the signal is now expressed as $x(n)$ and the STFT take the following expression:

$$STFT(m, k) = \sum_{-\infty}^{+\infty} x(n)w(n-m)\exp(-j2\pi kn/N) \tag{3.2}$$

Where $k$ is a band in the frequency domain. In the discrete case, usual elections for the window function include the rectangular function and the Hamming function. Both of them are shown in equations 3.3 and 3.4 respectively for windows of size N.

$$w(n) = \begin{cases} 1, & 0 \leq n \leq (N-1) \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

$$w(n) = \begin{cases} 0.5(1-\cos\frac{2\pi n}{N-1}), & 0 \leq n \leq (N-1) \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

The spectrogram is the visual representation of the STFT of a signal with respect to time and frequency, which are shown typically in the horizontal and vertical axis respectively. Every point of the plot is colored in accordance with the normalized value of the Fourier transform that correspond to that point in time and frequency. In figure 3.1 the spectrogram of a signal that was measured during the operation of a ball bearing element is shown. In order to define the spectrogram, there are two quantities of interest: the overlap and the size of the window $w$. First, the overlap represents the amount of points that two consecutives portions of the signal share when partitioned to compute the STFT. The use of overlap helps to avoid a loss of information due to the boundary between samples. The size of the window is the amount of points taken from the original signal in order to compute the STFT. This quantity defines the time and frequency resolution of the spectrogram. In simple words, if one wants higher resolutions in time, a narrow window should be used. For the oposite (higher resolutions in frequency), a wide window should be used. High resolution in both frequency and time can not be archieved at the same time. This is also called the "uncertainty principle" in signal processing.

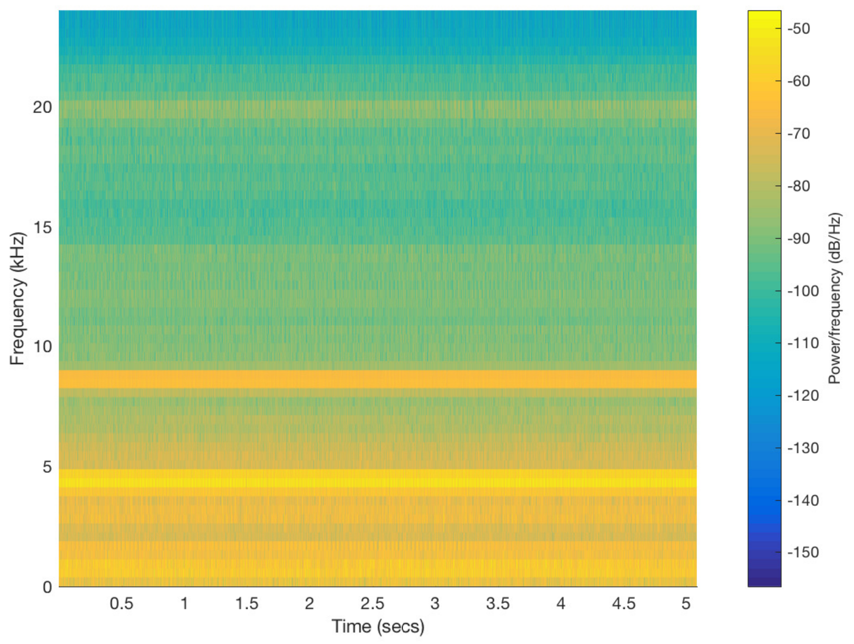


Figure 3.1: Spectrogram of an undamaged ball bearing element signal.

## 3.2 Principal Component Analisys

Principal Component Analisys (PCA) is a standard tool in modern data analysis, used for dimensionality reduction, information compression, feature extraction and data visualization. The main objective of the PCA algorithm is to reduce the dimensionality of a dataset, while conserving as much information as possible. This is archived by finding an orthogonal transformation **P** that transforms the original data into a new latent representation, in hope that this process will filter out the noise and reveal hidden relationships in the data. Below, some nomenclature is defined and then the general algorithm is explained.

First, the data will be represented as a matrix **X** that belong in the $\mathbb{R}^{N\times D}$ space. In **X**, the rows represent different data points and the columns represent the different characteristics of the data. The transformation **P** can be defined as another matrix, that belongs to a different space $\mathbb{R}^{D\times k}$. Then, the reduced data can be expressed as a matrix $\mathbf{Z} \in \mathbb{R}^{N\times k}$. The challenge is to find the correct coefficients of **P** that produces an optimal reduced representation. This optimal reduction is obtained when the variance of the projected data is maximized. It can be proved [27] that this happen when the transformation P is constructed in the following way:

1. The covariance matrix of **X**, $\mathbf{S_X}$, is computed.
2. The eigenvector and eigenvalues of $\mathbf{S_X}$ are determined.
3. For a reduced representation of dimension $k$, the greatest $k$ eigenvalues are selected, along with their corresponding eigenvectors.

4. The transformation is built by placing the selected eigenvectors as columns in a new matrix. That matrix corresponds to $\mathbf{P}$.

To actually perform the dimension reduction of the dataset, the following expression is computed.

$$\mathbf{Z} = \mathbf{X}\mathbf{P} \tag{3.5}$$

## 3.3 Machine Learning

Machine Learning can be defined as a set of methods that can automatically detect patterns and structure in data, and then use those learned characteristics to perform predictions or decision making tasks such as classification. In general, machine learning is a subset of the artificial intelligence field, that draw heavily from probability theory, statistics techniques and computer sciences. Models within machine learning attempts to learn from data, so that explicit programing is not needed for solving complex problems. The types of problem that can be solved with machine learning can be divided in, roughly, three different categories: supervised learning, unsupervised learning and reinforcement learning. For this thesis, models that belong to the first two are used, so here only they are explained. [28]

- **Supervised Learning:** In this type of problems, the main goal is to learn a mapping from inputs $x$ to outputs $y$ given a dataset containing a labeled set of N input-output pairs $D = \{(x_\mathrm{i}, y_\mathrm{i})\}_{\mathrm{i}=1}^{N}$. The existence of the label $y_\mathrm{i}$ for every input $x$ is fundamental in this type of approach, since all the training process is based on finding a relationship $f : x \to y$ that minimize the discrepancy between the desired outputs and the outputs that the model is producing.

  In general, each input $x_\mathrm{i}$ can be represented as a D-dimensional vector. If the input data is an image or a vibration signal, the process of expressing them as vectors is natural, but for more complex cases, as in message or written text recognition, some preprocessing may be necessary.

  The form of the outputs $y_\mathrm{i}$ depends mostly on the type of problem that is being solved. For classification problems, where the objective is to assign each input $x_\mathrm{i}$ to one or more classes from a set $C = \{c_1, c_2, ...c_k\}$, the form of the outputs is categorical. That means that every $y_\mathrm{i}$ is equal to the code of the class to which $x_\mathrm{i}$ belongs. For example, if $x_\mathrm{i}$ belong to the class $c_2$ from 4 possible classes, it is recommendable to work with an output written in the following way: $y_\mathrm{i} = [0, 1, 0, 0]^T$. This representation is often call *one-hot encode representation* in the literature.

  In the context of reliability, an example of a classification problem could be the diagnosis of failure modes in mechanical elements or the detection of the location for a certain fault. In section 3.4 a discussion about the different types of classification problems is presented to the reader.

- **Unsupervised Learning:** For these type of problems, the database only consists in a set of inputs $D = \{x_\mathrm{i}\}_{\mathrm{i}=1}^{N}$, and the objective is not to predict or classify anything, but to find patters or structures within the data. For that reason, this subset of machine learning is also called *knowledge discovery.* These kind of problems are much less well-defined and much more difficult in general, because nothing tells the models what kind of patters it have to look for, nor if there exists an error metric to use (in supervised learning, the comparison between the prediction and the desired output is what guide the algorithm towards good solutions).

The most common examples for these type of problems are the algorithms for *clustering*, which main objective is to group the different examples inside the database into classes, without having prior knowledge about their real categories, or if they exist at all

For this thesis, inside the context of supervised learning, neural networks will be used extensively as models capable of doing classification tasks. In the context of unsupervised learning, Variational Auto-Encoders and PCA are the most important models used for this work, where they are used for their dimensionality reduction capabilities.

## 3.4 Classification Problems

Inside classification problems, there are mainly four categories that depends on the number and structure of the classes present in the problem [29].

1. **Binary Classification:** In these types of problems, the input is to be classified into one of two non-overlapping classes: $C_1$ or $C_2$. Examples of this problem include the classification of the failure / no failure state in mechanisms, or the decision making process of choosing whether a maintenance should be performed or not.

2. **Multi-class Classification:** The objective of this kind of classification is to classify every input as one, and only one class from a set of $k$ non-overlapping classes. Examples of this type of problems include face recognition or voice recognition in the field of biometrics or, in the field of reliability, the diagnosis of failure modes in a mechanical element from a list of more than two possible faults that do not present themselves at the same time. For this thesis, this is the type of problem that the model for classification will aim to solve inside the context of ball-bearing elements.

3. **Multi-labeled Classification:** In multi-labeled problems, the objective is to classify each input into several of $K$ non-overlapped classes. These type of problems are more complex that multi-class ones since inputs can belong to more than one class, therefore the number of possibilities for each example increased by a great amount. In industrial sectors, this kind of problems are important for elements that can fail by more than one reason at the same time.

4. **Hierarchical Classification:** The input for these kind of problems is to be classify into one, and only one class, which could be divided into subclasses or grouped into different superclasses. One characteristic of these problems is that the hierarchy is assumed stable during the classification task. One example of this type of classification problem is the general failure of a gear box, where the first category could be binary

(fails or not fails), then if the first results is that the gear box failed the classification task goes into a second step to determine which element failed (ball-bearings, gears, bolts) and then into a third step to identify the failure mode that occurred in that specific element.

For all the types of classification problems discussed above, a high variety of metrics can be defined to measure the performance of a certain classifier. In this thesis, the metric used to compare the implemented models between them is the *accuracy of classification*, which can be understand as number of correct predictions divided by the total number of predictions. If $y_\mathrm{i}$ is the desired categorical output for the input $x_\mathrm{i}$ and $\hat{y}_\mathrm{i}$ is the output obtained from the classifier, then the accuracy for the dataset $D = \{(x_\mathrm{i}, y_\mathrm{i})\}_{\mathrm{i}=1}^{N}$ can be defined as:

$$Accuracy(y, \hat{y}) = \frac{1}{N} \sum_{\mathrm{i}=1}^{N} \mathbb{1}(y_\mathrm{i} = \hat{y}_\mathrm{i}) \tag{3.6}$$

Where $\mathbb{1}$ is equal to 1 if the condition in the argument is true, and 0 otherwise. $N$ represent the total number of samples that were classified, either correctly or incorrectly.

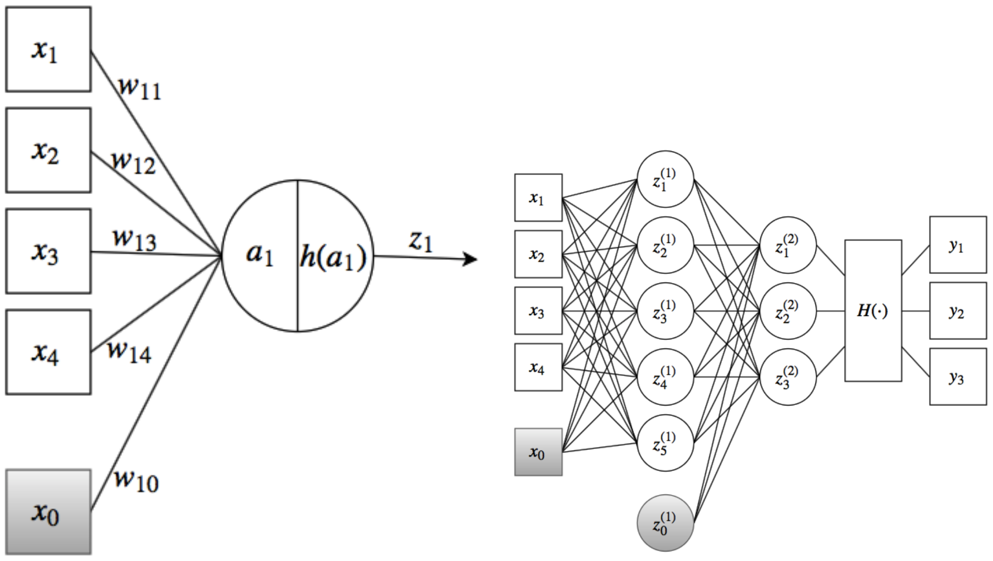
## 3.5 Artificial Neural Networks

Artificial Neural Networks (ANNs) are a type of model inside the field of machine learning which main objective is to approximate complex to express functions. One of the most popular applications for ANNs includes the resolution of supervised learning problems, in particular, classification tasks. For this, ANNs combine a basis of non-linear functions, where each function that belongs to that basis is in itself another non-linear function of a linear combination of the inputs, where the parameters that controls that combination are adaptive parameters that can be optimized towards an objective [27].

A vector of inputs for the ANN is defined as a vector $\vec{x} = \{x_1, ..., x_D\}$. The ANN itself consists of $C$ different layers stacked one on top of each other, where each layer consists of $c_k$ units or neurons. In the first layer of the ANN, the input vector $\vec{x}$ is combined linearly with a first set of parameters as expressed in equation 3.7:

$$a_j = \sum_{\mathrm{i}=1}^{D} w_{j\mathrm{i}}^{(1)} x_\mathrm{i} + w_{j0}^{(1)} \tag{3.7}$$

where $j = 1, ..., c_1$ is a subscript that indicates the respective neuron of the first layer and the superscript (1) indicates that the corresponding parameter belongs to the first layer of the ANN. The parameters $w_{j\mathrm{i}}$ are known as the *weights* of the network and $w_{j0}$ as *biases*. The quantity $a_j$ is known as the *activation* of the neuron $j$. With the purpose of making the writing of the equations more clear and concise, usually in literature the bias parameter



(a) Neuron unit. (b) Neural Network.

Figure 3.2: Graphical representation of (a) a neuron unit with input dimension $D = 4$ and (b) neural network with input dimension $D = 4$, 3 outputs units and 2 layers. The number of units in the first and second layers are $c_1 = 5$ and $c_2 = 3$ respectively.

is included as an extra weight. For this, it is necessary to define a new component of the input vector, $x_0$, which value is always equal to 1. With this new input, equation 3.7 can be rewritten as:

$$a_j = \sum_{\mathrm{i}=0}^{D} w_{j\mathrm{i}}^{(1)} x_\mathrm{i} \tag{3.8}$$

Where the bias parameter is still $w_{j0}^{(1)}$ and the component $x_0$ is equal to 1. Then, each of the activations $a_j$ is transformed by a differentiable, non-linear function $h(\cdot)$, as follows:

$$z_j = h(a_j) \tag{3.9}$$

As an example, figure 3.2a shows a graphical representation of a neuron unit that have as an input a vector $\vec{x}$ with dimensionality $D = 4$.

The quantities $z_j$ are known as the *outputs* of the neurons. In a second step, the process is repeated again but with the next layer of neurons and using as input the outputs of the previous layer, $\{z_1, ..., z_j, ..., z_{c_1}\}$, which have a dimensionality equal to the number of neurons of the first layer, $c_1$. Again, it is defined a new component of the current input vector, $z_0$, as 1 and the bias parameter is included into the weights for the new layer. In this case, the activations of the second layers can be written as:

$$a_l = \sum_{j=0}^{c_1} w_{lj}^{(2)} z_j \tag{3.10}$$

Where now, the subscript $l = 1, ..., c_2$ indicates the neuron unit of the second layer. It is important to notice that the weights and biases of the ANN belongs to a specific layer within the network, that is why now the superscript is (2). Again, these activations are transformed with the function $h(\cdot)$, in the form:

$$z_l = h(a_l) \tag{3.11}$$

At this point, if the ANN has more layers, the process is repeated again until the outputs of the last layer are computed. Once this happens, a last activation function $H(\cdot)$ is applied to them to generate the global output of the ANN, the vector $\vec{y}$.

$$\vec{y} = H(\vec{z}) \tag{3.12}$$

The mathematical expressions for the outputs of all the layers of the network can be combined to formulate a general expression. For the case where the ANN only consist of two layers, that expression can be written as:

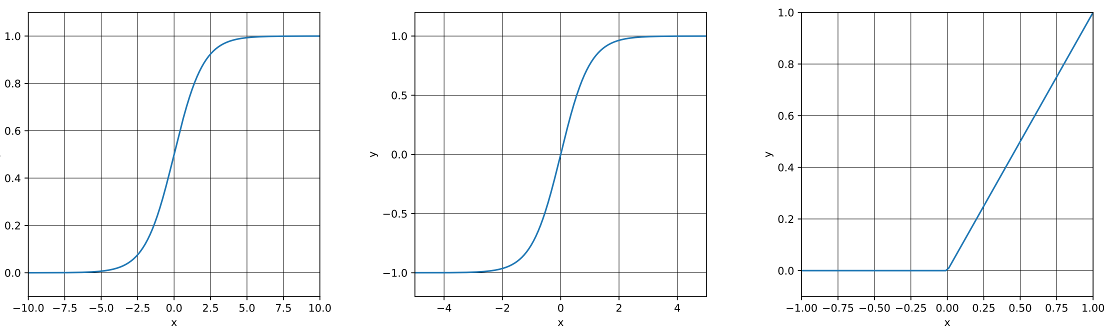
$$y_l(\vec{x}, \vec{w}) = \sum_{j=1}^{c_1} w_{lj}^{(2)} h\left( \sum_{\text{i}=1}^{D} w_{j\text{i}}^{(1)} x_\text{i} + w_{j0}^{(1)} \right) + w_{l0}^{(2)} \tag{3.13}$$

In equation 3.13, $\vec{w}$ is a vector that contains all the weights and biases of the ANN. The overall process of evaluating the previous expression can be interpreted as the *forward pass* of the input vectors through the network. Equation 3.13 also shows that the model of ANN is just a non-linear function that is controlled by a set of adaptive parameters represented by the vector $\vec{w}$. In figure 3.2b a graphical representation of a neural network is shown.

The explicit form that function $H(\cdot)$ will take depends mainly on the kind of problem that the ANN is design to solve. As in this thesis the use of ANNs is restricted to multi-class classification problems for the diagnosis of health states of mechanical elements, in section 3.5.2 is explained the most frequently used form for $H(\cdot)$ in those kinds of problems. Below, the activation functions used for the intermediate layers of the network are presented.

### 3.5.1 Activation Functions for the Intermediate Layers

In the intermediate layers of the network, the usage of functions that are non-linear and diferential in almost every point of its domain is what allows ANN to aproximate difficult to express or compute functions. A more detailed discussion about how a sum of non-linear functions can be used to approximate almost every function can be found in [30] and [31].



(a) Sigmoid function (b) Hyperbolic tangent Function (c) ReLU function

Figure 3.3: Activation functions for the intermediate layers of the ANN.

In the context of ANN, there are three very popular activation functions: the sigmoid function, hyperbolic tangent function and the ReLU function. In Figure 3.3 plots for all of them are portrayed.

The expression and plot for the sigmoid function are shown in equation 3.14 and figure 3.3a respectively. As one can see from the plot, the sigmoid function transform the input $x$ from the initial domain $\mathbb{R}$ to the space contained in the interval (0,1). This activation function is a popular choice for problems of binary classification, where it is needed to express the outputs of the neuron units as probabilities of the input belonging to one of two classes.

$$\sigma(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{3.14}$$

The expression for the hyperbolic tangent function is shown in equation 3.15. As one can see from Figure 3.3b, this function exhibit a similar behavior to the sigmoid function, where the only difference relies on the the space to which the inputs are transformed. In this case, that space is the interval $(-1, 1)$ instead of $(0, 1)$.

$$\tanh x = \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}} \tag{3.15}$$

Finally, the expression for the ReLU function is presented in equation 3.16. This function is nowadays the typical election as activation function in most neural networks. The plot for the ReLU function is shown in figure 3.3c.

$$ReLU(x) = \begin{cases} x \quad si \quad x \geq 0 \\ \\ 0 \quad si \quad x < 0 \end{cases} \tag{3.16}$$

### 3.5.2 Activation Function for the Output Layer

The election for the activation function used in the output layer will depend mostly on the type of problem for which the ANN is designed to solve. As it was mentioned before, the function $H(\cdot)$ is applied to the outputs of the last layer's neurons to generate the global output of the neural network.

For multi-class classification neural networks, the general idea is to output a vector $\vec{y}$ that represents in its $k$-component the probability that the input vector belongs to the class $k$. To do this, the most used activation function for the output layer is the *Softmax* function.

For a problem with $K$ different classes, the expression for the Softmax function applied on the last layer output vector $\vec{z}$ is shown in equation 3.17. As it can be seen from that expression, the output layer's vector is *squashed* into the interval $(0, 1)$ in a way that the sum of the transformed vector adds 1. Therefore, they can be interpreted as probabilities.

$$\vec{y} = H(\vec{z}) = \left\{ \frac{\mathrm{e}^{z_k}}{\sum_{\mathrm{i}=1}^{K} \mathrm{e}^{z_k}} \right\}_{k=1,\dots,K} \tag{3.17}$$

The following section continues the discussion on neural networks, specifically on the subject of their training.

### 3.5.3 Training of Artificial Neural Networks

As it was mentioned in section 3.5, an ANN is a non-linear function controlled by a set of adaptive parameters denoted by the vector $\vec{w}$ (which contains both the weights and the biases of the model). The main objective of every neural network is to approximate a function which is very difficult to express or compute. That function models some situation of interest. For example, it could be modeling the existing relationship between the acoustic emissions of a certain mechanical component and the presence of a certain fault in it. As neural networks are models that relies on a database to learn, the general idea behind its training is to minimize some notion of error or discrepancy between the response of the neural network and the desired output for a certain input $\vec{x}$. This process of minimization is perform varying the adaptive parameters contained in $\vec{w}$ until a local minimum is reached.

The error notion will be a function of both the parameters and the input data, $E(\vec{x}, \vec{w})$. Its explicit form will depend on the type of problem to be solve. The methodology that neural networks use to adapt their parameters to minimize function $E$ is based on a numeric technique of optimization called *Gradient Descent*. In what follows, gradient descent is explained.

**Gradient Descend**

In equation 3.18 it is shown the optimization problem that the training of neural networks present.

$$\vec{w}^* = \arg\min_{\vec{w}} E(\vec{w}, \vec{x}) \tag{3.18}$$

In many cases, in which the neural networks are included, it is impractical to solve an optimization practical by founding an analytical solution. For those situations, it is necessary to resort on an iterative method. The most common of these methods is gradient descent [32]. If $E(\vec{w})$ is a differentiable function that is desired to be minimize with respect to a vector of parameters $\vec{w} \in \mathbb{R}^D$, as in 3.18, it is always possible to compute a vector that contains all the partial derivatives of the function $E$ with respect to every parameter, as shown in equation 3.19:

$$\frac{\partial E}{\partial \vec{w}} = \left[ \frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_1}, ..., \frac{\partial E}{\partial w_D} \right]^T \tag{3.19}$$

In gradient descent, the iteration process starts with random initial values for $\vec{w}$. Then, in each iteration, the gradient vector is computed, and then every parameter is updated in the opposite direction of the gradient [1]:

$$w_{\mathrm{d}} \to w_{\mathrm{d}} - \eta \frac{\partial E}{\partial w_{\mathrm{d}}}, \quad \forall \mathrm{d} \in \{1, ..., D\} \tag{3.20}$$

Where $\eta$ is an hyperparameter called *learning rate* and determines how much to move in that direction in every step. With this process, $E(\vec{x}, \vec{w})$ will be minimized until a global or local minimum is reached, where the gradient is zero and therefore, the process is stopped. In practice, the process is stopped when the gradient reaches a value that is near zero with some kind of tolerance defined previously by the programmer. For neural networks, as the number of parameters increase heavily with the number of neuron and layers, a method called Back-Propagation (BP) was developed for performing gradient descent when the dimensionality of $\vec{w}$ results in a capacity problem. In a nutshell, the BP algorithm make use of the chain rule of multiplication to propagate the error from the layer $k$ to the layer $k-1$, thus making the process to run quicker and more efficiently. More details on this topic can be found in [27]. In what follows, convolutional neural networks (CNN) are introduced, as they represent an interesting variety of neural networks that have gained popularity in the recent years for their state of the art results within the computer vision field.

### 3.5.4 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are another model of Deep Learning, very similar to traditional ANN (described in Section 3.5). In fact, CNNs also have weights and biases that can be optimized via stochastic gradient descent and the back propagation algorithm in

[1] If it is desired to maximize the function $E$, then the parameter is updated in the positive direction of the gradient.

order to solve a classification or regression problem, they are also formed by neuron units, where every one of them receives an input, performs a dot product and then follows it with a non-linearity and they even has a score function that represent how well the model is learning the relationship between the inputs and the desirable outputs.

The main difference between CNN and traditional NN is in the topological treatment of the input [33]. CNN make the explicit assumption that the inputs has spatial dependencies between its different components (e.g images or time series), so the model can take advantage of the spatial relationship between the characteristics of the data. In simple words, ANNs convert the inputs to vectors prior its passing through the model, loosing all types of rich information related to the spatial location of each component with respect to the others. CNNs do not make that initial transformation of the input, so they tend to work better with data like images, for example.

A traditional CNN model is formed by three types of layers stacked that work together to both extract useful information from the data and to perform the classification or regression task. These layers are briefly described below:

1. **Convolutional Layer:** this type of layer performs a mathematical operation called *convolution* between the input and a set of learnable parameters organized to form a structure called *filter*. This filter slide through the input performing convolution, extracting feature maps in the process.

2. **Pooling Layer:** this type of layer performs a down sampling of its input, reducing the dimension of the data that flows through the network, to both making it more manageable and, hopefully, help in the process of extracting useful information from the initial input. The most common pooling layer is the *max pooling layer*, which slides a window of size $p \times p$ though the input and reports to the next layer only the maximum value of such window.

3. **Feed Forward Network:** in general, a CNN will have $n$ convolutional layers interspersed with $m$ pooling layers. This process is expected to work as a feature extraction process, obtaining in the end a vector of useful characteristics from the input. The last stage of the CNN is always a feed forward network, which is the one that performs the classification of regression task using as input those characteristics extracted by the combination of convolutional and pooling layers.

## 3.6 Variational Inference

One of the main challenges in probabilistic inference in general, and in fault diagnosis and prognosis in particular, is the approximation of difficult to compute probabilities densities. In Bayesian statistics, this problem arises as a fundamental one, since the posterior plays an important part in the inference of the quantities of interest. Currently, a powerful and flexible approach to approximate these probabilities densities is variational inference (VI).

Variational inference treat the problem of finding an approximate distribution as an optimization problem. To see this in a clearer way, it is necessary to first set the problem that variational inference try to solve. Let's consider a set of observations, $x = \{x_1, ..., x_D\}$, where $x_\mathrm{i}$ is a vector of dimensionality $D$. This set of observations is assumed to be controlled by a set of latent variables $z = \{z_1, ..., z_K\}$, where each one of them is a vector of dimensionality $K$. The joint distribution that make explicit the relationship between $x$ and $z$ is shown in equation 3.21:

$$p(x, z) = p(z)p(x|z) \tag{3.21}$$

In a Bayesian framework, latent variables are sampled from a prior distribution $p(z)$, which is common to be defined as a known probability distribution, and then related to the observations through the likelihood $p(x|z)$, which is the model that it is assume explain the data. The main objective is to make inference over the posterior distribution $p(z|x)$. This serves two main purposes. First, the values of $z_\mathrm{i}$ that generate each observation $x_\mathrm{i}$ can be of interest by itself. One example is when it is of interest to model some physical phenomenon and $z$ represent coefficients that have real world interpretations. For instance, in the diagnosis of cracks in steel mechanical parts, the data that can be measure from the crack itself may be its form, its size or the distance between the initial point of the crack and the border of the element, to name a few. One might also be interested in knowing some quantities that control those observations, but that are not easily measurable. For example, the velocity of the propagation of the crack or the amount of elastic energy that is being released by it. In the context of Bayesian models, Those quantities can be interpret as "hidden" or "latent" variables that controls what can be actually seen. The second purpose is that if it is possible to compute $p(z|x)$, then it can be used for computing $p(x)$ from the Bayes theorem. This is of interest in the case of generative models where the main objective is to generate new, unseen data that is alike to the existing data.

For complex models, computing $p(z|x)$ can be very difficult, in some cases even impossible. In situations like these, one possibility is to approximate the posterior $p(z|x)$ by some distribution $q(z|x)$ that belongs to a family $Q$ of probability distributions that can be parametrized by some set of parameters $\theta$. If $Q$ is a complex enough family to being able to find a good approximation to the posterior, but not so complex that the search for $q(z|x)$ in that family is infeasible, then variational inference may solve the problem of finding a good approximation for the posterior. Based on this approach, the objective is to solve the optimization problem of finding the closest approximate distribution $q(z|x)$ to the true posterior $p(z|x)$ when $q(z|x)$ is restricted to belong to $Q$. The key element of the problem is the notion of distance, or similitude between probability distributions. That is, the problem needs an objective function that can reflect how close two distributions are. This objective function is called *Kullback-Leibler divergence* [27]. For example, if $p$ and $q$ are two probability distributions defined over a continuous random variable $X$, then the Kullback-Leibler Divergence between those two probability densities is defined by equation 3.22:

$$KL(p||q) = \int_{-\infty}^{+\infty} p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x \tag{3.22}$$

If the properties of the logarithm and the definition of the expectancy over a random variable are used, the Kullback-Leibler divergence can be rewritten as it is shown in equation 3.23:

$$KL(p||q) = \mathbb{E}_{p(x)}[\log p(x)] - \mathbb{E}_{p(x)}[\log q(x)] \tag{3.23}$$

Using the form of equation 3.23 on the distributions $p(z|x)$ and $q(z|x)$, it results:

$$KL(q(z|x)||p(z|x)) = \mathbb{E}_{q(z|x)\sim Q}[\log q(z|x)] - \mathbb{E}_{q(z|x)\sim Q}[\log p(z|x)] \tag{3.24}$$

So, the variational inference approach to finding the approximate posterior can be understood as trying to solve the following optimization problem described in 3.25:

$$q^*(z|x) = \underset{q(z|x)\sim Q}{\arg\min}\, KL(q(z|x)||p(z|x)) \tag{3.25}$$

But the main problem that variational inference faces is that the objective function described in equation 3.25 is not estimable because it needs the computation of the logarithm of $p(x)$ which in turn is assumed to be unknown and that is also of interest when dealing with generative models. To see this, the fact that $p(z|x)p(x) = p(z, x)$ can be used to rewrite equation 3.24 as:

$$KL(q(z|x)||p(z|x)) = \mathbb{E}_{q(z|x)\sim Q}[\log q(z|x)] - \mathbb{E}_{q(z|x)\sim Q}[\log \frac{p(z,x)}{p(x)}] \tag{3.26}$$

Then, equation 3.26 can be written as:

$$KL(q(z|x)||p(z|x)) = \mathbb{E}_{q(z|x)\sim Q}[\log q(z|x)] - \mathbb{E}_{q(z|x)\sim Q}[\log p(z,x)] + \mathbb{E}_{q(z|x)\sim Q}[\log p(x)] \tag{3.27}$$

And because $\log p(x)$ is a constant with respect to the expectation on $q(z|x)$, it results:

$$KL(q(z|x)||p(z|x)) = \mathbb{E}_{q(z|x)\sim Q}[\log q(z|x)] - \mathbb{E}_{q(z|x)\sim Q}[\log p(z,x)] + \log p(x) \tag{3.28}$$

It is clear from equation 3.28 that for the computation of the objective function in equation 3.25 it is necessary to know $p(x)$ and, therefore, it is an intractable problem. To overcome this hurdle, one possibility is to try to solve a slightly different optimization problem, but with the same outcome as the previous one: an approximate distribution $q(z|x)$ close enough to $p(z|x)$. To do this, a new objective function is proposed, known as the evidence lower bound (ELBO). It is called the evidence lower bound because, as it can be seen in equation

3.29, it correspond to a lower bound of the logarithm of the evidence, or $p(x)$, given the the property of non-negativity of the KL divergence. Thus:

$$ELBO(q(z|x)) = -KL(q(z|x)||p(z|x)) + \log p(x) \tag{3.29}$$

As it can be seen from equation 3.29, ELBO function is the negative KL divergence plus the logarithm of $p(x)$, thus maximizing equation 3.29 is equivalent to minimizing equation 3.25 as $\log p(x)$ is constant under $q(z|x)$. Now, to show that ELBO is tractable, the expectancy definition of the KL divergence shown in equation 3.23 and the fact that $\mathbb{E}_{q(z|x)\sim Q}[\log p(x)] = \log p(x)$ are used, so that equation 3.29 becomes:

$$ELBO(q(z|x)) = -\mathbb{E}_{q(z|x)\sim Q}[\log q(z|x)] + \mathbb{E}_{q(z|x)\sim Q}[\log p(z|x)] + \mathbb{E}_{q(z|x)\sim Q}[\log p(x)] \tag{3.30}$$

Then, applying the Bayes theorem on $p(z|x)$ to rewrite equation 3.30 results in:

$$ELBO(q(z|x)) = -\mathbb{E}[\log q(z|x)] + \mathbb{E}[\log p(x|z)] + \mathbb{E}[\log p(z)] - \mathbb{E}[\log p(x)] + \mathbb{E}[\log p(x)] \tag{3.31}$$

Where the dependency of the expectancies over $q(z|x)$ have been omitted for simplicity. Reducing the previous expression and rearranging with the KL divergence definition, it results a computable form of the ELBO function as follows:

$$ELBO(q(z|x)) = \mathbb{E}_{q(z|x)\sim Q}[\log p(x|z)] - KL(q(z|x)||p(z)) \tag{3.32}$$

Therefore, the optimization problem to find an approximation $q(z|x)$ for the true posterior $p(z|x)$ can be defined as:

$$q^*(z|x) = \underset{q(z|x)\sim Q}{\arg\max}\, \mathbb{E}_{q(z|x)\sim Q}[\log p(x|z)] - KL(q(z|x)||p(z)) \tag{3.33}$$

In what follows, it will be shown how VI can be applied to a machine learning model called Variational Auto-Encoder in order to either generate new data or reduce the dimensionality of a certain database.

## 3.7 Variational Auto-Encoders

Variational Auto-Encoders (VAEs), originally proposed by [19] are generative models that combine neural networks, variational inference and unsupervised learning to address the problem of finding an approximation to a posterior probability distribution $p(z|x)$. They are called “Auto-Encoders” because they are composed of two main structures: the first one encodes the input data into a latent representation, whereas the second one decodes this latent representation onto an approximation or reconstruction of the original data. Thus, both parts are called encoder and decoder, respectively.

The main difference between VAEs and any other model that uses variational inference is that the former assumes well-known distributions for $p(z)$, $q(z|x)$ and $p(x|z)$. More precisely, when working with VAEs, it is assume that those distributions belong to a family of parametric distributions controlled by a set of parameters, which are treated as the desired outputs of the neural networks that conforms the encoder and the decoder of the VAE. Therefore, back propagation can be used to train those neural networks and then found good approximations for such parameters. These two features, i.e., the assumption of known forms for the distributions of the model and the use of neural network to approximate the parameters that controls such distributions, represent one of the main advantages of VAEs in comparison to other variational inference based approaches. Indeed, assuming a parametric form for the distributions of interest allows for the computation of the objective function in a much more efficient way (see Section 3.7.4). Also, the use of neural networks to obtain the parameters for such distributions permit the use of efficient optimization techniques such as Stochastic Gradient Descent. Next, a discussion about the parametric distributions forms for $p(z)$, $q(z|x)$ and $p(x|z)$ in the context of Variational Auto-Encoders is presented.

### 3.7.1 Prior over the latent variables $p(z)$

One of the main challenges in Bayesian models is defining the latent variables. Questions like the nature of the latent variables (should them be positive numbers, integer numbers) or the relationship between them are difficult to answer. VAEs take an approach to defining latent variables that make these questions to go away and only let the person in charge with the responsibility to choose the number of latent variables that the model will have. Following [19], it is assume that the latent variables are normally distributed with mean equal to 0 and covariance matrix equal to the identity:

$$p(z) \sim N(z|0, I) \tag{3.34}$$

This assumes no prior knowledge over the relationship between the variables nor the sign of them. The only apparent restriction is that they need to be real numbers centered around 0. But the following question arises: how such a simple election for the latent variables will be able to control complex data, such as vibration signals? The key is to note that any distribution in $k$ dimensions can be produced by a set of $k$ normally distributed random variables that are mapped through a complex enough function, as it is mention in [24]. Next, it is discussed what means to be a “complex enough” function.

### 3.7.2 The original data conditioned by the latent variables $p(x|z)$

The distribution $p(x|z)$ represent the probability that certain data point $x$ will be generated under the latent variable $z$. The nature of this distribution should be decided by the nature of the data itself. As shown in [19], if the original data are real numbers that can take either positive or negative values, a good election for this distribution is a normal distribution:

$$p(x|z) \sim N(x|f(z, \theta)) \tag{3.35}$$

But if they are binary valued, a good election is a Bernoulli distribution:

$$p(x|z) \sim Be(x|f(z, \theta)) \tag{3.36}$$

Above, in both equation 3.35 and 3.36, function $f$ is a neural network that takes a set of latent variables $z$ and a set of weights and biases $\theta$ and then outputs a vector of parameters for the correspondent distribution. For the Bernoulli distribution $f$ outputs a vector $\vec{\rho}$, which contains the probability of each output component to be either 1 or 0. For example, if the original data are black and white images, $\vec{\rho}$ will represent the probability of each pixel to be either black or white. But for gray scale images, each component of $\vec{\rho}$ can be interpreted as the intensity of gray of each pixel in the output image. For the case where $p(x|z)$ is a normal distribution, the function $f$ outputs a vector of means and a vector of variances that controls the sampling of each output dimension.

It can be seen that it does not matter if the prior distribution for the latent variables $z$, $p(z)$, is a standard normal distribution because if the neural network $f$ is complex enough, then the first layers of $f$ will do the job of finding a good approximation of the true but unknown distribution that controls the latent variables. Then, the rest of the layers are responsible for doing the rest of the transformation to obtain the vector of parameters.

This probability distribution, $p(x|z)$, and the neural network that outputs its parameters are called the decoder of the VAE since they take the latent representation $z$ and outputs a reconstruction, $x^*$, of the original data $x$. Thus, "reconstructing" or "decoding" the data from its latent representation.

### 3.7.3 The approximate posterior $q(z|x)$

The main objective of a VAE is to find a good approximation for the true posterior. The most common decision regarding the family $Q$ (from which the approximate distribution is search) corresponds to the family of multivariate isotropic normal distributions [24]. This serves a double purpose. First, the election of this family allows each latent variable to have its own mean and variance, so it is a relatively flexible model. Second, an isotropic Normal distribution, in conjunction with the selection of the prior for $p(z)$, makes the KL divergence in the objective function of equation 3.33 to have a close and easy to compute form (see details in section 3.7.4). Thus, the following is true for the VAE model:

$$q(z|x) \sim N(z|\mu(x, \phi_1), \sigma(x, \phi_2)I) \tag{3.37}$$

Here, $\mu$ is a neural network that takes as input the original data $x$ as well as weights and biases denoted by $\phi_1$ and then outputs a vector of means for the latent variables; $\sigma$ works in a similar way, i.e., it is another neural network that takes the original data as input, it has a set of weights and biases denoted by $\phi_2$, and outputs a vector of variances. Since the distribution $q(z|x)$ is forced to be isotropic, that vector of variances is multiplied by the identity matrix to form the covariance matrix. The distribution $q(z|x)$ and the two neural networks that parametrize $q(z|x)$ are called the encoder of the VAE since its task is to make a codification of the original data into a latent representation $z$.

Next, it is discussed how this model is optimized to solve equation 3.33 and find a good approximation to the true posterior. Since VAEs use the variational inference approach, the function ELBO can be used as the objective function (See equation 3.32) since it has an easy to compute form. Furthermore, as stated before, one of the advantages of VAEs is that, since the probability distributions are parametrized by neural networks, stochastic gradient descent can be used to find the maximum of equation 3.33. But first, it is needed to be more precise about the explicit form that the ELBO function takes given the decisions made about the distributions $p(z)$, $q(z|x)$ and $p(x|z)$. Indeed, the ELBO function is made of two parts: the KL divergence between the prior $p(z)$ and the approximate posterior $q(z|x)$; and the expectancy over the logarithm of $p(x|z)$. In what follows, these two parts are tackled separately to show how they are estimated.

### 3.7.4 KL Divergence $KL(q(z|x)||p(z))$

Since $q(z|x)$ and $p(z)$ are chosen to be normal distributions, the KL divergence between them has closed form. Indeed, equation 3.38 shows that the KL divergence between two normal distributions when one is an isotropic normal distribution and the other is a standarized one is [24]:

$$KL(N(\mu, \sigma I)||N(0, I)) = \frac{1}{2}(tr(\sigma I) + \mu^T \mu - k \log \det(\sigma I)) \tag{3.38}$$

where $k$ is the dimensionality of the distribution. Since both distributions outputs vectors or parameters related to the latent variables, $k$ is the dimensionality of the latent variables, i.e., the number of latent variables that controls the model under study. In equation 3.38, $\mu$ and $\sigma$ represent the outputs of the neural network of the encoder.

### 3.7.5 Expectancy over the logarithm of $p(x|z)$

For a feasible training process of the VAE model, an efficient way of computing $\mathbb{E}_{q(z|x)\sim Q}[\log p(x|z)]$ is needed. One approach would be to sample from $z$ to obtain a good estimation of the expectancy. But this is very expensive because it requires a significant number of passes through $f$ (which, in this case, is a complex neural network) in order to estimate $p(x|z)$. As stated in [24], this problem can be addressed by noting that when searching for the approximate distribution $q(z|x)$, Stochastic Gradient Descent (SGD) is being used. Thus, SGD can also be used on the sampling of $z$, such that a sample of $z$ is taken, then from that sample $\log p(x|z)$ is estimated and then treat that result as an estimation of $\mathbb{E}_{q(z|x)\sim Q}[\log p(x|z)]$. This process is repeated many times until it converges to a good approximation of the true posterior.

### 3.7.6 Putting the pieces together

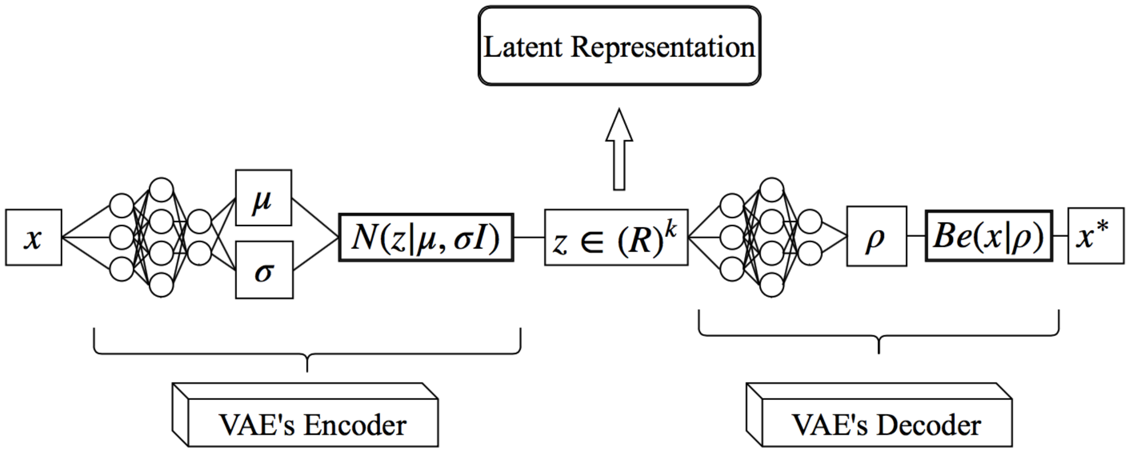
Figure 3.4a show the VAE model comprised of the various elements discussed in the previous sections for the case when the nature of the data requires that the probability distribution of the decoder, $p(x|z)$ belongs to the Bernoulli family. In a similar way, Figure 3.4b represent the model of the VAE for the case when $p(x|z)$ is a normal distribution. Note that in both cases, the encoder and the decoder work together to first transform the original data into a latent representation and then to decode that latent representation into a reconstruction of the original data.

However, there is still one problem with this model. Since it uses neural networks to estimate the parameters of the distributions in the VAE model, the ideal situation would be to use backpropagation to optimize this model along with SGD. But backpropagation cannot work with stochastic units inside the neural network, since they are non-continuous operations and, therefore, the gradient is not defined at these points. Nevertheless, what backpropagation can do is handle stochastic inputs to the network, if they are sampled outside of it and, for the neural network, they work as any other deterministic input. To solve this problem, one can refer to a trick called "the reparametrization trick" as proposed by [19], which works using the following property:

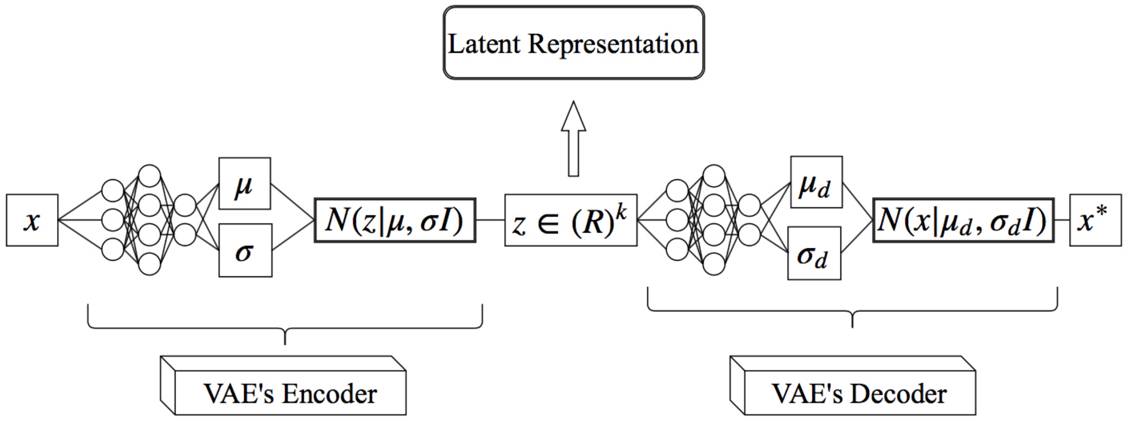
$$N(\mu, \Sigma) = \mu + \sqrt{(\Sigma)} * \varepsilon \tag{3.39}$$

which states that a normal distribution with mean $\mu$ and covariance matrix $\Sigma$ is equivalent (i.e., has the same distribution) as $\mu + \sqrt{(\Sigma)} * \varepsilon$ when $\varepsilon$ is a random variable following a standard normal distribution. Since in this VAE model $q(z|x)$ is parametrized by an isometric normal distribution, the matrix of covariances can be fully represented by its diagonal vector, in this case, $\vec{\sigma}$. Based on equation 3.39, the model can be optimized with backpropagation as illustrated in Figures 3.5a and 3.5b.

Thus, once the Variational Auto-Encoder model has been trained and optimized, it can be used to generate new data by feeding values of $z$ sampled from the prior distribution $p(z) \sim N(0, I)$ into the decoder, obtain the vector of parameters that controls $p(x|z)$ and

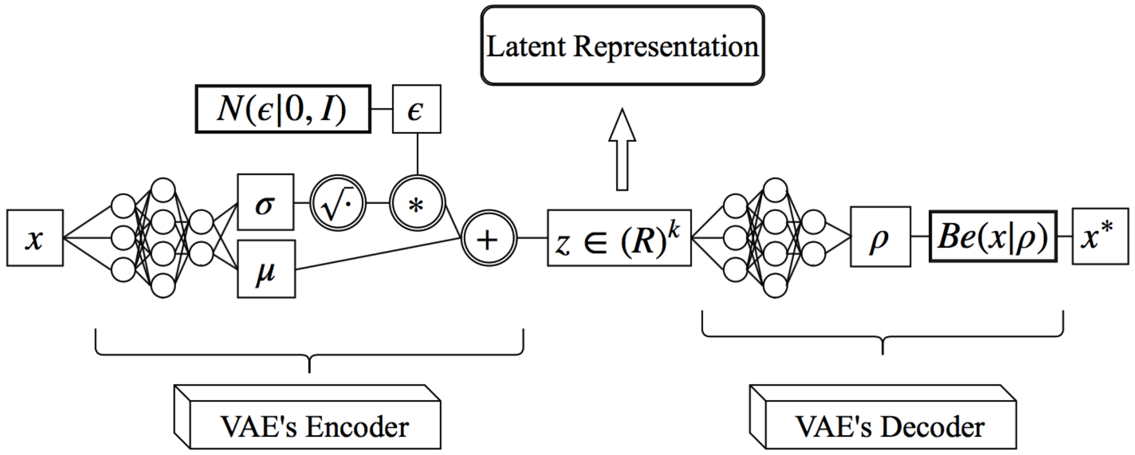


(a) VAE model with a decoder containing a Bernoulli probability distribution, suited for binary data or real valued data restrain to the $(0, 1)$ interval, for example, images.

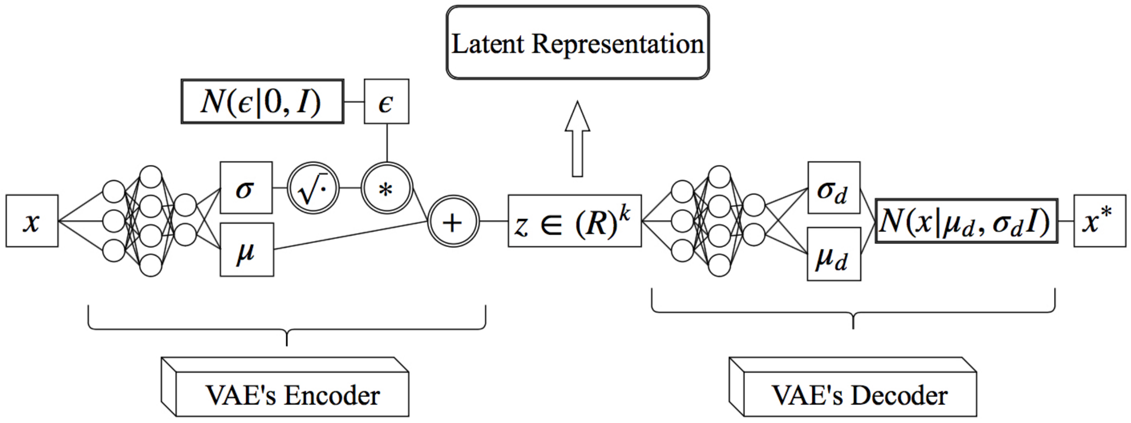


(b) VAE model with a decoder containing a Normal probability distribution, suited for real valued data, for example, acceleration signals.

Figure 3.4: Variational Auto-Encoder Models with both the Encoder and Decoder networks. The latent representation is denoted as $z$, and can be interpreted as a compressed characterization of the data. Both (a) and (b) represents the cases where the probability distributions of the decoder belong to the Bernoulli and normal family, respectively.



(a) VAE model with reparametrization trick applied for the case where a decoder probability distribution is a Bernoulli distribution.



(b) VAE model with reparametrization trick applied for the case where a decoder probability distribution is a normal distribution.

Figure 3.5: VAE model with the reparametrization trick applied. Here, $\varepsilon$ is a vector of dimensionality equal to that of the latent space, sampled from a standardized multivariate normal. The vector of variance $\sigma$ is first passed through a square root operation (denoted by $\sqrt{(\cdot)}$), then multiplied element-wise (operation $*$) with the vector $\varepsilon$. The result of these operations is added to the vector of means $\mu$ (represented by $+$). Both (a) and (b) represents the cases where the probability distributions of the decoder belong to the Bernoulli and normal family, respectively.

posteriorly, sample from it to obtain a synthetic, newly generated data point. More important in the context of the proposed dimensionality reduction method, one can interpret and make use of the obtained latent space from this trained VAE model (the output of its encoder) as a set of features, i.e., a feature map, providing a high-level representation (abstraction) of the vibration signals. Note also that is the user who defines the dimension of such latent space, thus being able to specify the desired level of compression or dimensionality reduction.

These features represented by the latent variables can then be used in a segmentation task accomplish by a clustering technique for unsupervised fault identification or, as it shall be done in this thesis, supervised fault identification via a classifier in terms of a neural network. The latter shall be discussed in detail by means of examples of applications in Chapter 7.

# Chapter 4

# The Proposed Approach for Dimensionality Reduction in Health States Diagnosis

In this section, the integration between the prior VAE model and the proposed dimensionality reduction is discussed. The process starts with a dataset where each data point has dimension $\Psi$. If it is wanted to reduce this dimensionality, it is necessary to find a way in which every data point of the database gets a smaller representation in some latent space. That transformation from the original space to the latent space can be linear as in PCA, or it can be more complex, generating a series of latent variables for each original data point. The proposed VAE based approach makes use of the latter alternative.

In a nutshell, the VAE based approach for dimensionality reduction comprises the following four steps (see Figure 4.1):

1. Acquire vibration signals from the system under analysis.
2. From the original vibration signals, generate the dataset $X$ that will be used to train the VAE. $X$ contains $N$ points, each of dimension $\Psi$. The following three options are considered (see Section 5.2).
   (a) Spectrogram images
   (b) Vectors of manually extracted features from the original, derivative and integral of the signal as well as frequency domain
   (c) Raw Vibration Signals
3. Perform unsupervised training of the VAE model with the vectors obtained by the transformation of the original vibration signals explained in the previous step and a chosen dimension for the latent space.
4. Use the encoder of the trained VAE to transform the vectors of the original dataset,

which have dimensionality equal to $\Psi$ to the latent space of dimension $k$.

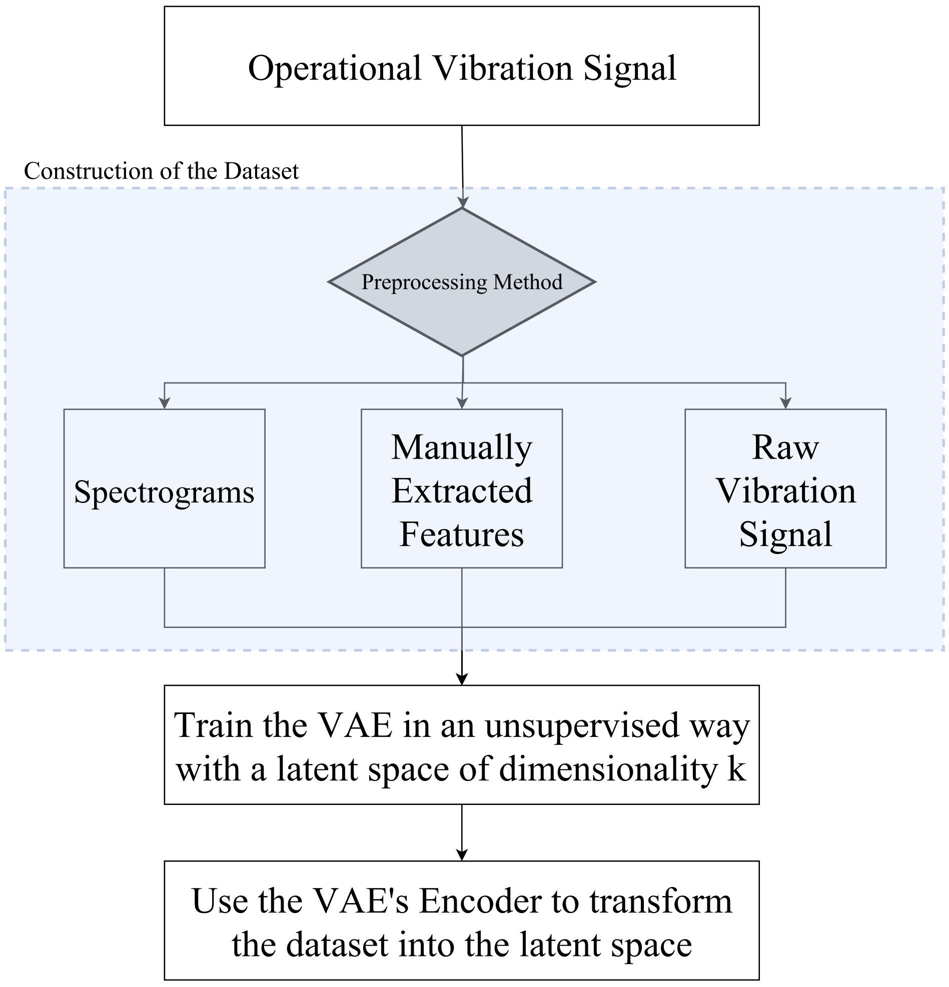

Operational Vibration Signal
Construction of the Dataset
Preprocessing Method
Spectrograms
Manually Extracted Features
Raw Vibration Signal
Train the VAE in an unsupervised way with a latent space of dimensionality k
Use the VAE's Encoder to transform the dataset into the latent space

Figure 4.1: Proposed Approach for the VAE based dimensionality reduction.

Step 1 refers to the process of acquiring vibrations signals from the system of interest. In this case, those signals correspond to acceleration data measured during the operation of ball bearing elements. In step 2, from the vibration signals a dataset is generated following one of three preprocessing methods. All of them consider splitting the original vibration signal into chunks of length $L = 1024$ points with an overlap of 50% between adjacent samples. As explained in more details in Section 5.2, from the chunks of the original vibration signal, the first method computes spectrogram images, the second method manually extracts 100 traditional features and the third method feeds the chunks directly to the VAE model, i.e., no preprocessing is performed on the vibration signals.

As the VAE model does not require labels to learn the latent representation of the data, unsupervised training can be performed. In step 3, the VAE model is trained in an unsupervised way reconstructing the input dataset (e.g., spectrogram images, features or raw signal). The dimensionality of the latent space $k$ is chosen equal to the desire dimensionality of the data once the reduction is performed. In step 4, to reduce a data set $x$ to its latent representation $z$, with a trained VAE model, it is necessary to feed that dataset through the encoder's neural network to obtain values for the variance vector $\vec{\sigma}$ and the means vector $\vec{\mu}$ and then use the reparametrization trick (as explained in Section 3.7.6) to obtain $z$. Note, however, that this process induces some variability into the latent representation since the latent variables are sampled from a probability distribution. It is usually wanted for the latent representation of the data to be deterministic for certain specific input since that would allows to obtain consistent results that do not depend on the variance of each latent variable. Thus, instead of sampling from $q(z|x)$ to obtain $z$, the vector of means $\vec{\mu}$ is taken as the latent representation in the form $\vec{\mu} = z$. As the neural network that produces the mean vector $\vec{\mu}$ is a deterministic function, for a certain input data point, the encoder produces the same vector of means, thus delivering the same specific representation in the latent space for that data point.

Note that steps 3 and 4 not only depend on the choices made for the $p(z)$, $p(x|z)$ and $q(z|x)$ distributions, but also on the architectures of the neural networks for the VAE's encoder and decoder. If the architectures are not complex enough, the process might result in a poor approximate posterior distribution $q(z|x)$. On the other hand, if the training is not well performed, we might not take full advantage of the approximation power of the VAE. In either case, the VAE's encoder will be poorly optimized and then produce a far from ideal high-level representation of the input data thus leading to a poor dimensionality reduction.

In what follows, the design of proper architectures for both the encoder and decoder is discussed. Since this process depends on the nature and structure of the data upon which the VAE operates, the following chapter discusses the types of data and the corresponding datasets that we use for validating and exemplifying the proposed approach. Then, in Section 5.3 the architectures used to train the different VAE models depending on the type of dataset are shown.

# Chapter 5

# Proposed Architectures for the VAE model

For the experiments performed in this thesis in order to test and evaluate the proposed approach, operational data of ball bearing elements from two different on-line repositories were used. The first one is from the Case Western Reserve (CWR) University Bearing Data Center and the second one is from the Machinery Failure Prevention Technology (MFPT) Society. Then, to the data of both repositories, three different methodologies for the preprocessing of the data were tested: 1) compute spectrograms, 2) compute traditional features manually and 3) the case where no preprocessing of the vibration signal is performed, i.e., when the raw data is directly to the VAE. This chapter first introduce the details concerning both data repositories and then explain the different methodologies of preprocessing used, to end with a discussion about the selection of the internal architectures for the encoder and decoder of the VAE.

## 5.1 Ball Bearing Datasets

### 5.1.1 Case Western Reserve University Bearing Data Center (CWR)

The first data repository is from the Case Western Reserve (CWR) University Bearing Data Center [34]. A Reliance electric motor with two horsepower was used with ball bearings in experiments for the acquisition of vibration data on both the drive end and fan end bearings, although for this thesis, only the data corresponding to the drive end bearing was used. This bearing correspond to an SKF deep-groove ball bearing, model 6205-2RS JEM. The signal is generated from sensors located in the housing of the drive end bearing. Single point artificial faults ranging in diameter from 0.18 to 0.71 mm were seeded in the bearing with an electro-discharge machining, where the fault was either located in the balls, the outer ring or the inner ring. Also, zero to three horsepower motor loads were used in the experiments.

For the purpose of this thesis, location of the faults and the fault sizes themselves are included as individual classes, thus resulting in 12 classes present in this data repository. Every class name and the number of examples that belong to each class can be seen in Table 5.1. To better represent monitoring conditions typically found in field applications, and to make the problem more challenging, 5% of white Gaussian noise was added to the vibration signals prior their preprocessing. As discussed before, the vibration signals corresponding to each class were split into chunks of length $L = 1024$ points each with an overlap of 50% between adjacent samples, then this repository consists in 13617 samples in total. Figures 5.1(a),(b),(c) and (d) show an example of the raw signals prior to the addition of the noise for the baseline, inner race fault, outer race fault and ball fault, respectively. Note that these figures only represent the differences between signals produced by different fault locations, and does not show the actual classes that were used, which are further divided using the fault size too.

Table 5.1: Classes for the CWR dataset.

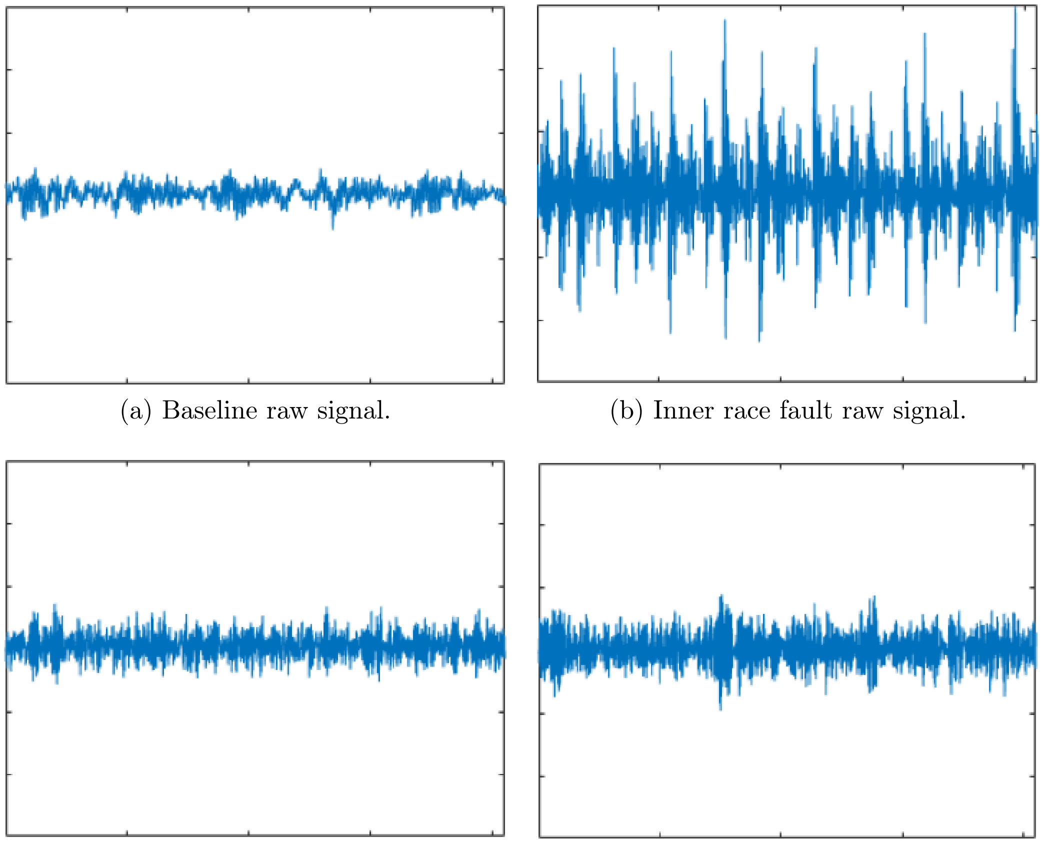
| **ID of the Class** | **Size of the fault [mm]** | **Location of the fault** | **Number of samples** |
|---|---|---|---|
| **18BF** | 0.18 | Balls | 938 |
| **36BF** | 0.36 | Balls | 939 |
| **53BF** | 0.53 | Balls | 939 |
| **71BF** | 0.71 | Balls | 931 |
| **18IR** | 0.18 | Inner Race Ring | 940 |
| **36IR** | 0.36 | Inner Race Ring | 936 |
| **53IR** | 0.53 | Inner Race Ring | 938 |
| **71IR** | 0.71 | Inner Race Ring | 932 |
| **18OR** | 0.18 | Outer Race Ring | 941 |
| **36OR** | 0.36 | Outer Race Ring | 938 |
| **53OR** | 0.53 | Outer Race Ring | 941 |
| **Baseline** | Undamaged | Undamaged | 3304 |

### 5.1.2 Machinery Failure Prevention Technology Society (MFPT)

The second repository used in this thesis was provided by the Machinery Failure Prevention Technology (MFPT) Society [35]. An experimental test rig with a NICE bearing[1] gathered accelerometer data for three conditions. First, a baseline condition was measured at 270lbs of load and a sampling rate of 97,656 Hz. Second, ten total outer-raceway faults were tracked. Three outer race faults were loaded with 270lbs with a sampling rate of 97,656 Hz, and seven outer race faults were assessed at varying loads: 25, 50, 100, 150, 200, 250 and 300 lbs. The sampling rate for the outer race faults was 48,828 Hz. Third, seven inner race faults were analyzed with varying loads of 0, 50, 100, 150, 200, 250 and 300 lbs. The sampling rate for the inner race faults was 48,848 Hz.

Following the same procedure as it was done with the CWR repository, the original vibration signals were split into chunks of length $L = 1024$ points each, with an overlap of 50% between adjacent samples. The classes present in this repository are: normal baseline (N), inner race fault (IR), and outer race fault (OR). The total number of samples used for

[1]Further information regarding the type of ball bearing used can be found in [36]



(a) Baseline raw signal. (b) Inner race fault raw signal.

(c) Outer race fault raw signal. (d) Ball fault raw signal.

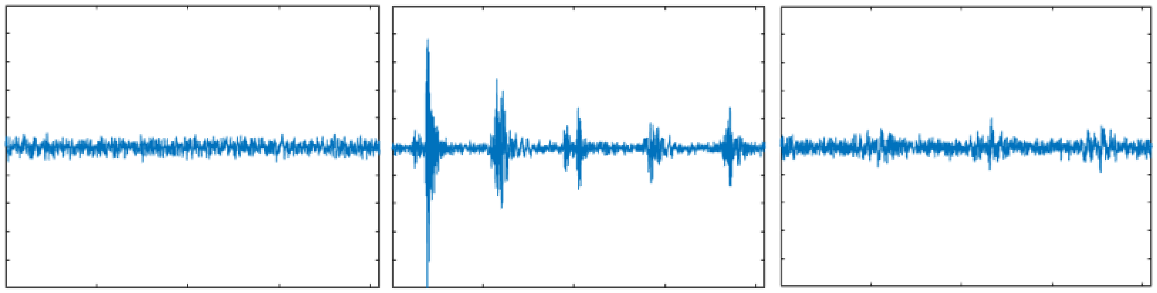
Figure 5.1: Amplitude-time signals for the baseline and the three fault locations present in the CWR dataset.

each class are shown in Table 5.2, with a total of 10,808 samples. Figures 5.2(a),(b) and (c) show an example of the raw signals for the baseline, inner race fault and outer race fault, respectively. From these figures, there are a few areas of notice. The noise level within the baseline and outer race data appears to be higher than the inner race. The baseline and outer race faults are similar in look, hence the potential difficulty in the conducting fault diagnosis on this data set.

Table 5.2: Classes for the MFPT dataset.

| ID of the Class | Location of the fault | Number of data points |
|---|---|---|
| **IR** | Inner Race Ring | 1981 |
| **OR** | Outer Race Ring | 5404 |
| **Baseline** | No Failure | 3423 |

In the next section, the methodologies used for the preprocessing of the data are discussed. Note that the data has previously been divided into chunks of length $L = 1024$ points as stated before.


(a) Baseline raw signal. (b) Inner race fault raw signal. (c) Outer race fault raw signal.
Figure 5.2: Amplitude-time signals from the three health states present in the MFPT dataset.

## 5.2 Data Preprocessing and Data Types

### 5.2.1 STFT and Spectrograms

The first methodology used for the generation of the dataset is to compute the spectrogram of the chunks of original vibration signals using the STFT (explained in Section 3.1) and then scale down the spectrograms into images of $p$ by $p$ pixels using a bilinear interpolation [37]. Note that the selection for the parameter $p$ which controls the image size, is not set in the approach as it depends on various factors. For example, if one is dealing with reduced amount of vibration data, the number of spectrograms that will be generated might be less than ideal. In this case, it may be a good idea to use images with higher resolutions so to compensate for the reduced dataset size. Another important factor that affects the size of the images is the available computational capacity. If that is an issue, then smaller images will result in a VAE model with fewer parameters to be optimized, thus allowing for a faster optimization. In both cases dealing with ball bearing elements discussed in Section 7.1, for this thesis it was chosen a size of 96 by 96 pixels for the images, and one color channel (gray scale), since that size represents a good compromise between speed and resolution of the data given the available hardware (a Nvidia Titan XP GPU). But since the VAE model is designed to work with inputs in the form of one dimensional vectors, not matrixes as if it would be if the inputs are images, a process of converting the $p$ by $p$ images into vectors with dimensionality equal to $p^2$ is performed to the scale down spectrograms. This process is shown graphically in Figure 5.3 but basically consist in stacking the rows of each image horizontally to form the corresponding vector. As one can see, this dataset will consist on vectors that store the information contain originally in gray scale images, thus, the type of data in this dataset is real-valued, restrain to be in the interval $(0, 1)$. Because of this, for this dataset, the probability distribution used for the decoder of the VAE is chosen to be from the Bernoulli family, as it was discussed previously in Section 3.7.2.

### 5.2.2 Manually Extracted Features

This type of dataset was generated with the same number of classes and the same number of data points per class as the spectrograms for each data repository. Also, each sample

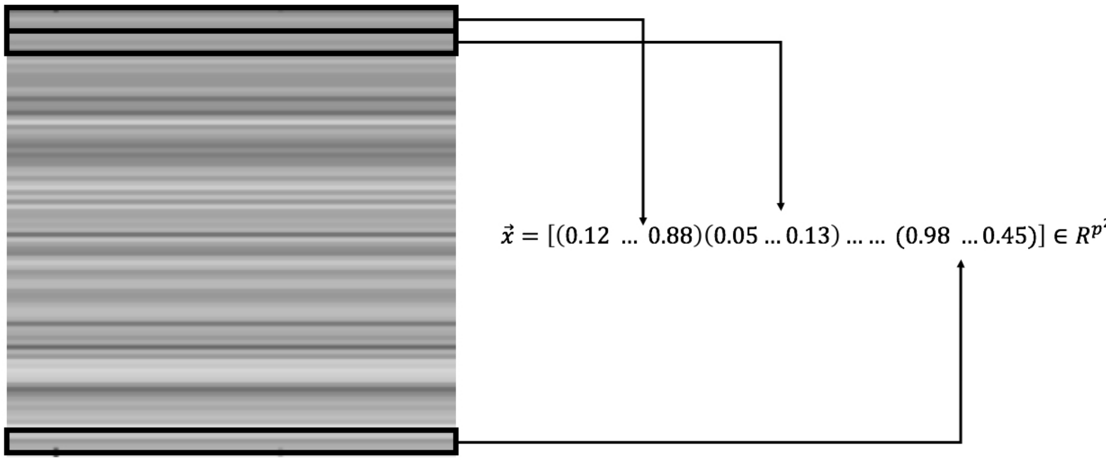

$\vec{x} = [(0.12 \ldots 0.88)(0.05 \ldots 0.13) \ldots\ldots (0.98 \ldots 0.45)] \in R^{p^2}$

Figure 5.3: Diagram showing the procedure of generating vector from the images of spectrograms.

was generated from the same portion of the original vibration signal as the correspondent spectrogram. This dataset consists of a series of 100 hand-engineered features from the original, derivative and integral form of the vibration signal as well as the frequency domain (see Table A.1 in the Appendix for the full list of features extracted). Since now the type of data has changed from spectrograms to manually extracted features, the probability density of the decoder, $p(x|z)$ (see Section 3.7.2) cannot be of the Bernoulli family anymore because the data is real valued and not necessarily restricted to the $(0, 1)$ interval, so $p(x|z)$ takes now the form of an isotropic normal distribution of dimension equal to 100, as the VAE tries to reconstruct the input from its latent representation.

### 5.2.3 Raw Vibration Signal

For comparison purposes, it is in this work's best interests to test the case where no preprocessing is performed to the original vibration signal. This will show if the VAE model can work with raw data directly fed from the sensors. For this purpose, a third type of dataset is created using directly the chunks (with length $L = 1024$ points) of data obtained from splitting the original signal as samples. Each one of these chunks were the ones used to generate the correspondent spectrogram image or vector of features in the previously explained datasets. Note that since this dataset will also contain real valued data not necessarily restricted to the $(0, 1)$ interval, then the probability distribution of the decoder, $p(x|z)$, will have the same form as in the dataset containing features extracted manually, i.e, a normal isotropic distribution.

Now, once the data repositories used in this thesis and the types of preprocessing have been described, next section will discussed the election of the architectures for the neural networks of the encoder and decoder inside the VAE model.

## 5.3 Proposed Architectures for the VAE's Encoder and Decoder

The design of the proposed architecture for the encoder and decoder neural networks within the VAE model was accomplished based on the comparison between the results of a series of classification tasks using as input the reduced dataset generated by the VAE model under those architectures. In total, nine different architectures were tested, where both the encoder and the decoder are fully connected neural networks with one, two or three hidden layers and ReLUs or hyperbolic tangents activation functions in every one of them, depending on the type of data used to train the VAE. If the data is binary or real valued but restricted to the $(0, 1)$ interval (i.e. the dataset containing spectrogram images), then the activation function will be the ReLU function. On the other hand, if the data is real valued and not restricted to the $(0, 1)$ interval, the activation function used was the hyperbolic tangent. This was done because the ReLU activation function in both the VAE's encoder and decoder deep neural networks tend to be numerically unstable due to the input data characteristics (real valued and restricted to the $(0, 1)$ interval, as mentioned before), thus resulting in deficient training and data dimensionality reduction. To address this issue, it was found that the use of a hyperbolic tangent activation function for those two neural networks solve the numerical instabilities.

Indeed, let the number of hidden units for the encoder be $(E_1, E_2, E_3)$ and for the decoder $(D_1, D_2, D_3)$ for the first, second and third hidden layers, respectively. Table 5.3 shows the tested architectures (i.e., number of hidden layers and corresponding number of units per layer).

Table 5.3: Architectures tested for the VAE model. A zero means that the correspond hidden layer was not part of the architecture.

| **ID** | $E_1$ | $E_2$ | $E_3$ | $D_1$ | $D_2$ | $D_3$ |
|---|---|---|---|---|---|---|
| **Architectures with one hidden layer** | | | | | | |
| **1** | 200 | 0 | 0 | 200 | 0 | 0 |
| **2** | 500 | 0 | 0 | 500 | 0 | 0 |
| **3** | 1000 | 0 | 0 | 1000 | 0 | 0 |
| **Architectures with two hidden layers** | | | | | | |
| **4** | 200 | 100 | 0 | 100 | 200 | 0 |
| **5** | 500 | 250 | 0 | 250 | 500 | 0 |
| **6** | 1000 | 500 | 0 | 500 | 1000 | 0 |
| **Architectures with three hidden layers** | | | | | | |
| **7** | 400 | 200 | 100 | 100 | 200 | 400 |
| **8** | 1000 | 500 | 250 | 250 | 500 | 1000 |
| **9** | 2000 | 1000 | 500 | 500 | 1000 | 2000 |

The fault diagnosis tasks used for the design of the VAE's architecture are the same employed in the validation process of the proposed approach, discussed in Chapter 6. In particular, the same values for $k$ (presented in Table 6.1) and the same classifiers (presented in Table 6.2) are used to obtain values for the classification accuracies under different topologies of each architecture. When the classification tasks were completed, the final accuracies of the different fault diagnosis experiments based on the dimensionality reduction produced for each architecture were average to obtain a representative accuracy value of a given architecture.

Then, for each type of dataset generated (spectrograms, extracted features or the use of raw vibration signals) it is proposed to chose the architecture that, based on that average, delivers good performance in both CWR and MFPT data repositories and use such architecture for both of these study cases.

First, the spectrograms based results and the choice of architecture for this type of dataset is discussed. The average accuracies obtained from the experiments using the spectrograms images dataset as the input for the VAE model are shown in Table 5.4.

Table 5.4: Final average accuracy for each architecture tested using spectrogram images.

| **Architecture ID** | **CWR-Spectrograms** | **MFPT-Spectrograms** |
|---|---|---|
| **1** | 87.45% | 87.24% |
| **2** | 92.55% | 90.35% |
| **3** | 93.19% | 89.95% |
| **4** | 92.80% | 90.54% |
| **5** | 96.46% | 90.84% |
| **6** | 95.96% | 90.75% |
| **7** | 96.07% | 90.98% |
| **8** | 96.28% | 90.26% |
| **9** | 95.77% | 88.61% |

Therefore, it can be seen from Table 5.4 for the CWR and the MFPT data repositories, architectures #5, #6, #7 and #8 show similar performance. For this type of dataset, the chosen architecture is architecture #7, which correspond to the smaller three-hidden layer network because it allows for a deeper architecture on both the encoder and decoder without significantly increasing the number of required parameters to optimize. Thus, the selected architecture comprises three hidden layers with 400, 200 and 100 units, respectively, in the encoder and 100, 200 and 400 units in the decoder. Also note that the value of the average accuracies does not present great variation among different architectures, being the simpler one (only one layer with 200 units) the only significant outlier. This shows other characteristic of the VAE model: its stability to different internal neural network topologies in both the encoder and decoder.

The selection of the architecture of the VAE for the case where manually extracted features are used as the dataset is performed in a similar manner. The average accuracies obtained for this dataset are shown in Table 5.5 for both the MFPT and CWR study cases.

As shown in Table 5.5, architecture #8 is the one that shows overall good performance when it is used for both the CWR and MFPT cases. Then, for when features extracted manually are used to generate the dataset, the chosen architecture is architecture #8 for the VAE model. The selected architecture comprises three hidden layers with 1000, 500 and 250 units for the encoder NN, respectively, and 250, 500 and 1000 units for the decoder NN.

The average accuracies for when the dataset is constructed with non-preprocessed portions of the original vibration signal can be seen in Table 5.6:

Table 5.5: Final average accuracy for each architecture tested using features extracted manually.

| **Architecture ID** | **CWR-Features** | **MFPT-Features** |
|---|---|---|
| **1** | 59.05% | 98.44% |
| **2** | 57.50% | 98.74% |
| **3** | 56.52% | 98.16% |
| **4** | 59.40% | 98.95% |
| **5** | 62.32% | 98.61% |
| **6** | 64.16% | 98.99% |
| **7** | 62.12% | 99.01% |
| **8** | 65.01% | 99.30% |
| **9** | 64.09% | 98.38% |

Table 5.6: Final average accuracy for each architecture tested using portions of the original vibration signal without any kind of preprocessing.

| **Architecture ID** | **CWR-Raw Vibration Signal** | **MFPT-Raw Vibration Signal** |
|---|---|---|
| **1** | 50.21% | 56.76% |
| **2** | 51.74% | 58.27% |
| **3** | 51.47% | 59.34% |
| **4** | 49.16% | 56.32% |
| **5** | 52.86% | 59.66% |
| **6** | 54.80% | 61.37% |
| **7** | 51.11% | 57.95% |
| **8** | 53.29% | 61.47% |
| **9** | 48.46% | 65.01% |

As shown in Table 5.6, there is not a single architecture that delivers good performance for both the CWR and MFPT data repositories when the dataset consists in the raw vibration signal (i.e. when no preprocessing is performed). For this, the best solution is to choose architecture #8 and #9 for the CWR and MFPT case study respectively when this type of dataset is used. Even though for the CWR data repository there is a better performing architecture, the author thinks that it is best to limit the analysis to architectures with three hidden layers for simplicity.

Therefore, as shown in Figure 5.4, for every type of dataset used (spectrogram images, features extracted manually and the raw vibration signals) the proposed architecture for the deep neural network of the encoder has a total of five layers: three hidden layers and two different output layers with their own weights and biases. The first output layer is for estimating the vector of means $\vec{\mu}$ and the second one is for estimating the vector of variances $\vec{\sigma}$. The number of units of these output layers is equal to the dimension of the latent space, $k$. However, the biases and weights of the hidden layers connected to these two output layers are the same. Besides, the input layer of the encoder has $\Psi$ units, where $\Psi$ is equal to $p^2$ in the case of the spectrogram images, 100 for the features manually extracted and 1024 when the raw vibration signal without any kind of preprocessing is used.

On the other hand, the decoder's deep neural network architecture will have three hidden layers and one output layer for the case where the decoder's probability distribution belongs

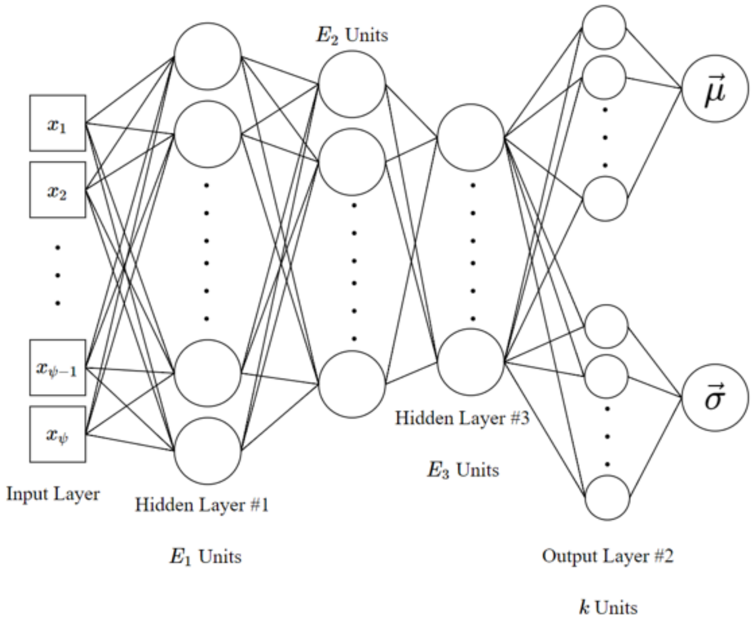
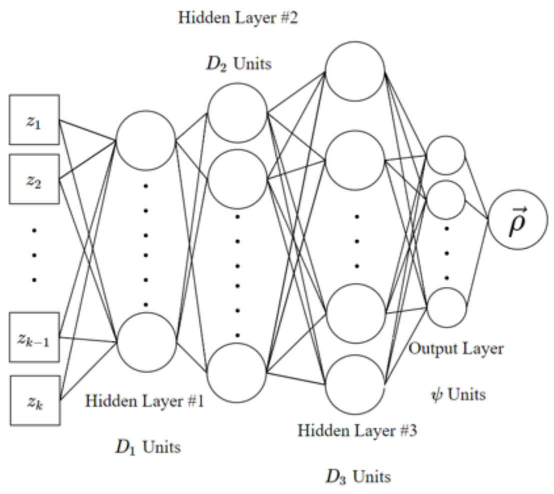
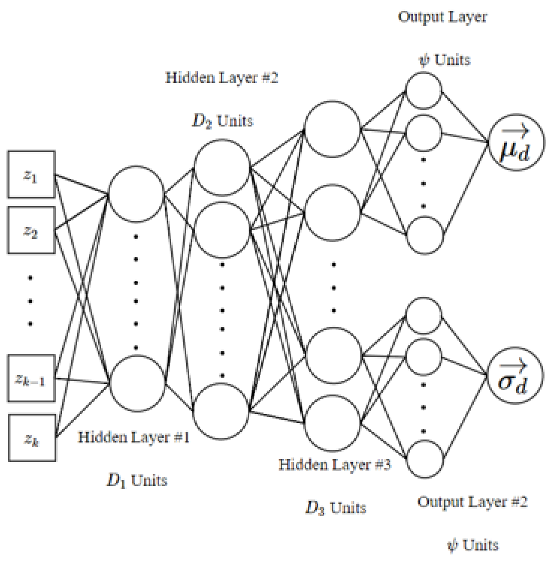


Figure 5.4: The encoder is composed by a total of five layers: three hidden layers with $E_1$, $E_2$ and $E_3$ units, respectively, and two outputs layers, one for the variances vector $\vec{\sigma}$ and the other for the median vector $\vec{\mu}$. The number of output units in both output layers is equal to the dimension of the latent space.





(a) Decoder's neural network for the case where the type of data used requires that $p(x|z)$ takes the form of a Bernoulli distribution.

(b) Decoder's neural network for the case where the type of data used requires that $p(x|z)$ takes the form of an isotropic normal distribution.

Figure 5.5: The decoder is composed of three hidden layers with $D_1$, $D_2$ and $D_3$ units, respectively, and one or two outputs layers depending on the type of data used for the training of the VAE.

to the Bernoulli family, in which case that output layer will produce a vector of Bernoulli parameters to reconstruct the original vector (see Figure 5.5a).For the case where the decoder's probability distribution is an isotropic normal, then the decoder's neural network will have two output layers, in a similar way as the encoder, one for estimating a vector of variances and other for estimating a vector of means (see Figure 5.5b).

Based on the proposed architectures for the encoder and decoder, the latent representation provided by the encoder of the trained VAE model is used to reduce the input data dimensionality from $\Psi$ to $k$ (the latent space of dimension), as illustrated in Figure 5.6.

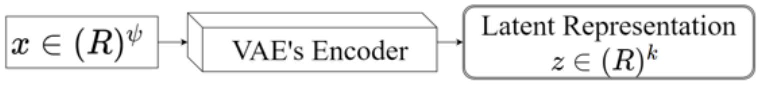


Figure 5.6: Transformation model using only the encoder of the trained VAE.

# Chapter 6

# Validation Procedure

To validate the proposed approach, a series of experiments are performed where the VAE based latent representation is used to train a neural network classifier for fault diagnosis, in which the true system's health states are known. Also, for the sake of comparison, the fault diagnosis results obtained based on the VAE's latent representation of the spectrograms images are compared to the ones obtained when performing fault diagnosis with both no dimensionality reduction (i.e., the baseline) and based on dimensionality reduction via PCA. Notice, however, that the latent representations are obtained in a fully unsupervised manner via the proposed VAE architecture and from PCA. The latter has been chosen as it is straightforward to use and one of the most popular dimensionality reduction techniques in the fault diagnosis community.

It is also the interest of this thesis to investigate two problems. The first one is when the amount of labeled data is low, but the number of non-labeled examples is high. This is of importance in the industry because it is often cheaper and easier to acquire massive amount of unlabeled data for which one might not know the underlying health state for the streaming data other than the baseline (undamaged) condition. Thus, identifying specific health conditions (i.e., labeling the data into classes) is usually expensive and requires a significant number of man-hours of highly qualified personnel. In this context, one objective is to explore whether the latent representation of the data created by the VAE can identify the different faulty conditions in an easy-to-learn manner, so the fault classifier can be successfully trained and operate with a reduced number of labeled examples. This is accomplished by taking advantage of the fully unsupervised training of both the VAE and the PCA on the totality of the available data. Then, using the obtained latent representation, the dimension of the dataset is reduced and only the portion of the data for which the system's health is known is used to train and test the classifier responsible for the fault diagnosis.

The second problem of interest is the selection of the number of dimensions of the latent space, $k$. Since this is a decision that ultimately relies on the analyst responsible for the fault diagnosis, it is advantageous to have a method of dimensionality reduction able to compress the original data into different latent spaces without a significant loss of information and therefore impact on the quality of that compression. Thus, reductions to different number of dimensions are performed and a discussion about how the fault diagnosis performance metrics are affected by different values of $k$ is held afterwards.

The experiments were performed for the two data repositories considered (CWR and MFPT) and the three different methodologies to generate datasets described in Section 5.2. The results corresponding to the usage of spectrogram images as inputs for the model are show in Section 7.1. Results regarding the usage of features extracted manually or the raw vibration signals can be found in Section 7.2 and Section 7.3 respectively.

Table 6.1 shows the values of the percentage of labeled data, $\varepsilon$, and the dimension of the latent space, $k$, used in the experiments discussed in Sections 7.1, 7.2 and 7.3.

Table 6.1: Values for $k$, the number of dimensions of the latent space, and $\varepsilon\%$, the percentage of labeled data.

| $k$ | $\varepsilon$ |
|---|---|
| 2, 4, 8, 16, 64, 128 | 1%, 5%, 25%, 100% |

Note that when features are extracted from the original vibration signals, the number of features extracted is 100. For this reason, the considered values for the dimension to which the reduction is performed, for this case are the same as in Table 6.1 except for $k = 128$, since it represents an expansion in dimensionality and not a reduction.

The fault diagnosis is accomplished by neural network classifiers. To analyze the impact that different neural network architectures have on the classifier operating on representations provided by both the VAE and PCA, three different fully connected NN based classifiers are used (see Table 6.2): MLP3LDO, MLP1LDO and MLP1L. All of them use softmax activation for the output layer, cross-entropy cost function and were trained with a learning rate of $10^{-4}$ and a maximum of 15000 epochs. ReLU activation is used for the hidden layers. The number of hidden layers and neurons for each of these three fault classifiers are also shown in Table 6.2.

With respect of the initialization of parameters, for the VAE model the weights were initialized with the Xavier initialization [38] and the biases with arrays full of zeros of the correspondent dimensionality. For the NN classifiers, both the weights and biases were initialized as samples of a normal distribution with mean equal to zero and variances equal to the identity matrix.

Table 6.2: Architecture details for the three different classifiers tested. HL stands for Hidden Layer.

| **Classifier** | **HL** | **# of Units in HL** | **Regularization** |
|---|---|---|---|
| **MLP3LDO** | 3 | 500-400-300 | Dropout with 50% prob. in all HL. |
| **MLP1LDO** | 1 | 100 | Dropout with 50% prob. in the HL. |
| **MLP1L** | 1 | 100 | None |

Therefore, the methodology for the validation of the proposed dimensionality reduction for fault diagnosis comprehends the following steps:

1. Choose the values for $\varepsilon$ and $k$ from Table 6.1 and a fault classifier architecture from Table 6.2 and divide the dataset into a training and testing sets in the proportion 3:1.

2. Perform unsupervised training of both VAE and PCA using the totality of the training set. In the case of the VAE, the training was performed with batches of 100 samples, 500 epochs and a learning rate of $10^{-4}$.

3. Perform dimensionality reduction over the training and test datasets with VAE and PCA.

4. Extract an $\varepsilon\%$ of the transformed training dataset (labeled data).

5. Train the chosen NN fault classifier with only the portion of labeled data extracted from the transformed dataset.

6. Use the totality of the testing dataset to evaluate the NN fault classifier performance. For each classifier, value for $k$ and $\varepsilon$ the correspondent classification task is repeated 10 times in order to obtain an average accuracy and standard deviation.

For the baseline case, in which no reduction in dimensionality is performed, the same steps as before apply, but skipping steps 2 and 3.

All the results shown in this thesis were obtained using a computer running and i7 7700k processor, 32GB of DDR4 RAM, a Nvidia Titan X graphical processing unit with Tensorflow 1.2.0. and CUDNN 8.0.

# Chapter 7

# Case Studies Results

In this chapter, results regarding the comparison between the proposed VAE model for dimensionality reduction, the model that uses PCA and the model that do not perform any kind of reduction in the dimensionality of the data are presented. For this, section 7.1 discusses the case where spectrograms are used as dataset, while section 7.2 and section 7.3 adresses the cases where manually extracted features and the raw vibration signal compounds the datasets, respectively. Finally, in section 7.4 a new architecture for the VAE is introduced, where the internal neural networks of the encoder and decoder are replaced by convolutional neural networks. Note that the format of the results in all the tables of this chapter is the following: average accuracy $\pm$ standard deviation.

## 7.1 Spectrograms Images.

As mentioned above, first this thesis will explore the proposed VAE based dimensionality reduction for the case where the datasets are composed of spectrogram images for both data repositories, CWR in Section 7.1.1 and MFPT in Section 7.1.2.

### 7.1.1 Case Study #1: CWR

The results from the CWR data repository using 96x96 pixels spectrogram images are shown in Figures 7.1, 7.2 and 7.3. The plots shown the average accuracy in fault diagnosis based on different MLP architectures and as a function of the dimension of the latent space. Table 7.1 shows the top average accuracies and standard deviations achieved in the fault classification tasks based on the VAE and PCA data representations as well as when no dimensionality reduction is done. The complete set of results are given in Table B.1 in the Appendix. Each MLP is trained on a given percentage $\varepsilon$ of the transformed training dataset. It can be seen that the fault diagnosis tasks achieve higher accuracies for lower dimensions, i.e., for $k = \{2, 4\}$, when operating on the VAE based latent representation and irrespective of the classifier architecture, i.e., a more robust data compression is achieved. Also, note

Table 7.1: Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the CWR datasets are preprocessing applying the STFT and then scaling down the resulting spectrograms to images of 96 by 96 pixels.

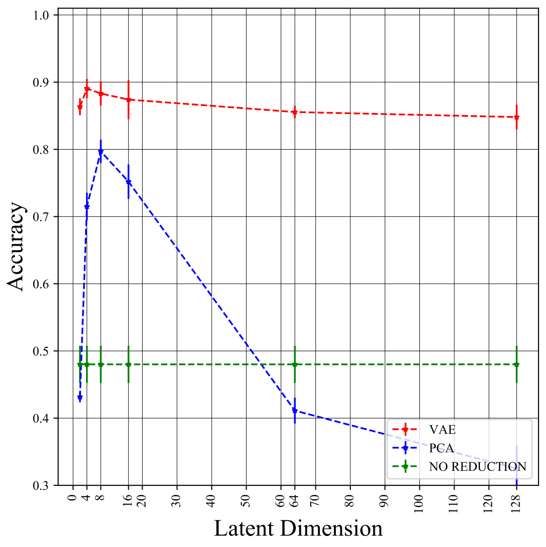
| | **CWR - Spectrogram Images** | | | | | |
|---|---|---|---|---|---|---|
| | **VAE** | **PCA** | **Baseline** | **VAE** | **PCA** | **Baseline** |
| | $\varepsilon = 100\%$ | | | $\varepsilon = 25\%$ | | |
| Classifier | MLP1L | MLP1LDO | MLP1L | MLP1LDO | MLP3LDO | MLP1L |
| $k$ | 128 | 128 | - | 128 | 128 | - |
| Accuracy | 98.55% ± 0.17% | 99.08% ± 0.14% | 98.35% ± 0.18% | 98.12% ± 0.16% | 98.28% ± 0.24% | 93.52% ± 0.72% |
| | $\varepsilon = 5\%$ | | | $\varepsilon = 1\%$ | | |
| Classifier | MLP1LDO | MLP3LDO | MLP1L | MLP3LDO | MLP3LDO | MLP1L |
| $k$ | 128 | 16 | - | 128 | 16 | - |
| Accuracy | 96.99% ± 0.43% | 96.74% ± 0.24% | 77.17% ± 2.08% | 90.22% ± 1.41% | 91.22% ± 1.08% | 48.00% ± 2.77% |

that in general, superior accuracy results are obtained with the VAE's representation when smaller labeled datasets (i.e., for $\varepsilon = \{1\%, 5\%\}$ ) are used in the training of the MLP classifier. This is of interest in situations where identifying the health states of an equipment is labor intensive and therefore expensive, such as in the big machinery data context, because reasonable fault diagnosis results are achieved with a relatively small investment in labeling data by a domain expert. Moreover, for cases where significant labeled data is available for training (i.e., $\varepsilon = \{25\%, 100\%\}$) both Variational Auto-Encoders and PCA show strong latent representations and the resulting fault diagnosis accuracies are comparable.

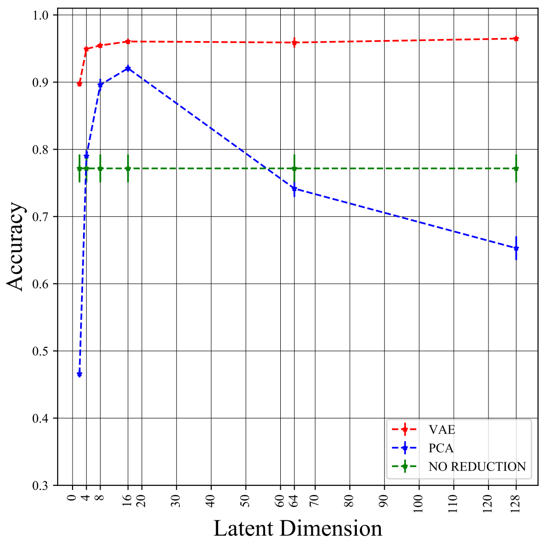
However, from Figure 7.3 and Table 7.1, note that for $\varepsilon = \{1\%, 5\%, 25\%\}$ the best fault diagnosis accuracies based on the PCA representation are obtained from the most complex MLP, i.e., with three hidden layers and dropout regularization. This might be an indication that is the MLP doing the heavy lifting to compensate for the not as rich and robust representation provided by the PCA. On the other hand, fault diagnosis based on the VAE representation results in reasonable results even when a simple one hidden layer MLP classifier is used. Indeed, the fault diagnosis accuracies for 1% and 5% of labeled data based on the single hidden layer MLP with dropout (MLP1LDO) and with a latent space of dimension of 2 are 85.27% and 89.16% for the VAE based representation, whereas these results are equal to 42.71% and 47.36% for the PCA based representation, respectively. When the latent space has a dimensionality of 4, the results are 88.93% and 95.33% for the VAE based representation whereas they are 75.90% and 81.61% for the PCA based representation, respectively, as shown in Table B.1 in the Appendix.

From Figure 7.1, Figure 7.2 and Figure 7.3 note also that the fault diagnosis accuracies based on both the VAE and PCA reductions are superior to the ones when no reduction is performed (baseline). This is particular significant for the fault diagnosis results from the three-hidden layers with dropout MLP with 25% and 100% of labeled data where the accuracies are 23.47% and 24.10%, as shown in 7.3(c) and 7.3(d) (where the line does not even show in range) and Table B.1 in the Appendix.

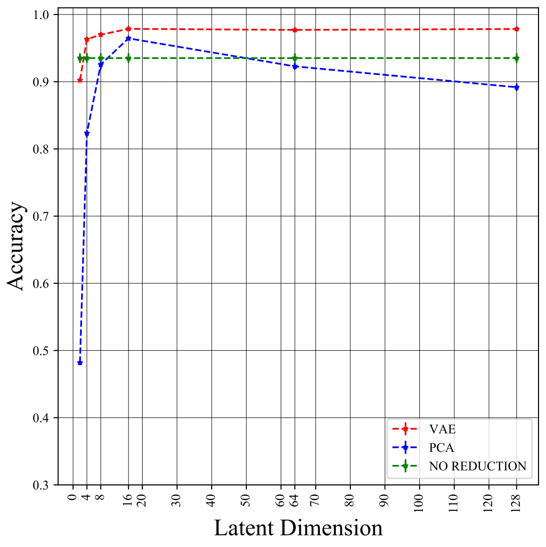
Overall, one can argue that the proposed Variational Auto-Encoder architecture delivers a better latent representation for fault diagnosis, especially when dealing with low latent dimension and reduced amount of labeled data. In cases where abundant labeled data is available and a not so drastic dimensionality reduction is required, both VAE and PCA are comparable based on the fault diagnosis accuracies.



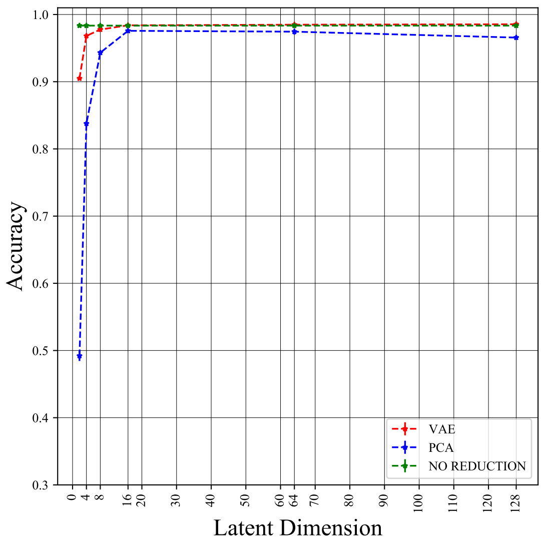
(a) CWR Spectrograms - MLP1L - $\varepsilon = 1\%$



(b) CWR Spectrograms - MLP1L - $\varepsilon = 5\%$

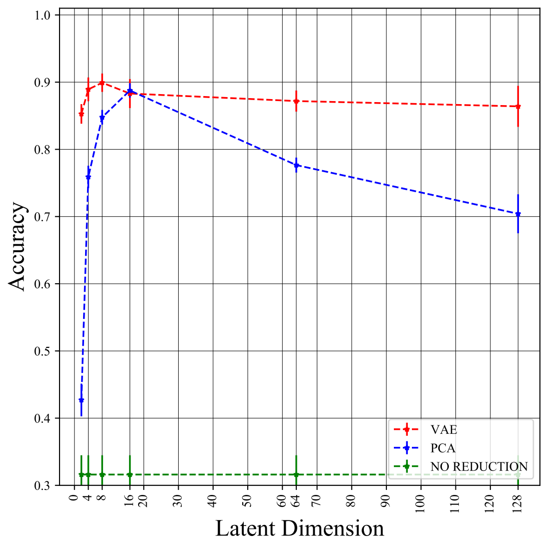


(c) CWR Spectrograms - MLP1L - $\varepsilon = 25\%$

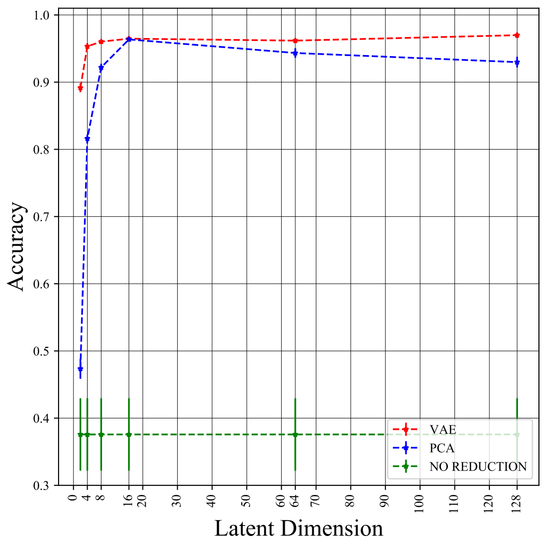


(d) CWR Spectrograms - MLP1L - $\varepsilon = 100\%$

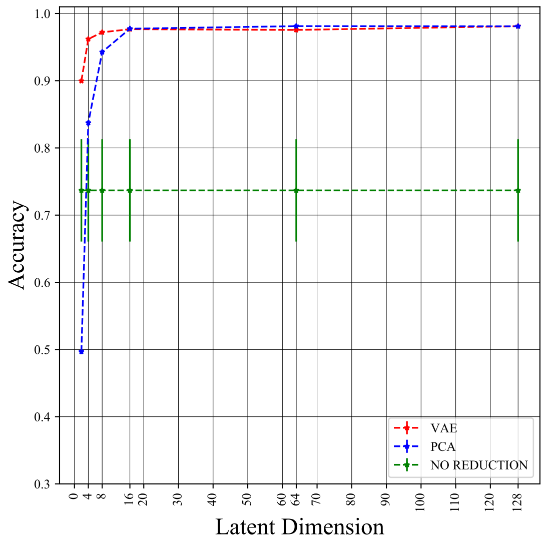
Figure 7.1: Average accuracy versus latent space dimension for the CWR dataset and MLP1L classifier with spectrograms.



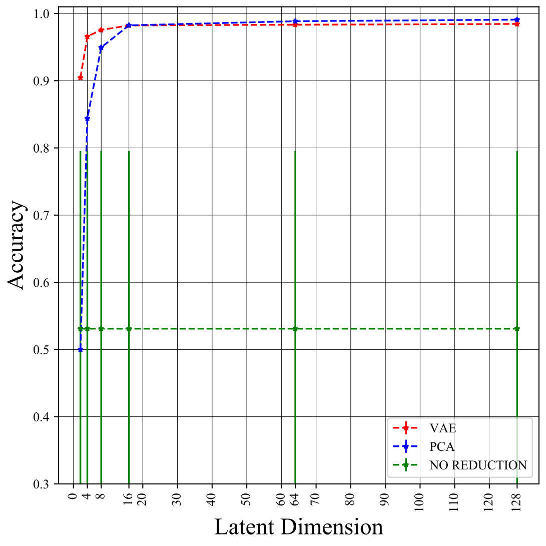
(a) CWR Spectrograms - MLP1LDO - $\varepsilon = 1\%$



(b) CWR Spectrograms - MLP1LDO - $\varepsilon = 5\%$

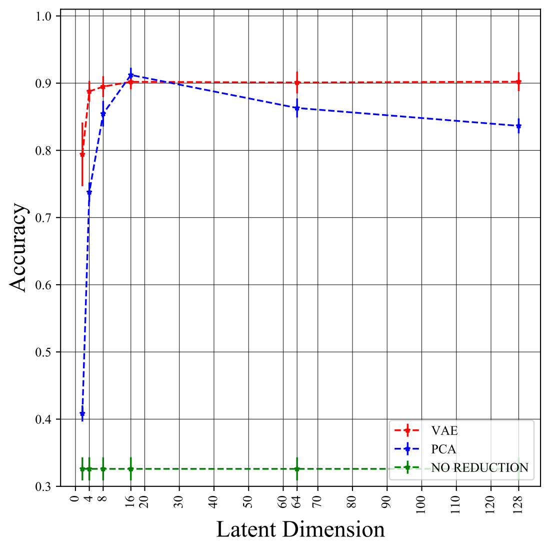


(c) CWR Spectrograms - MLP1LDO - $\varepsilon = 25\%$

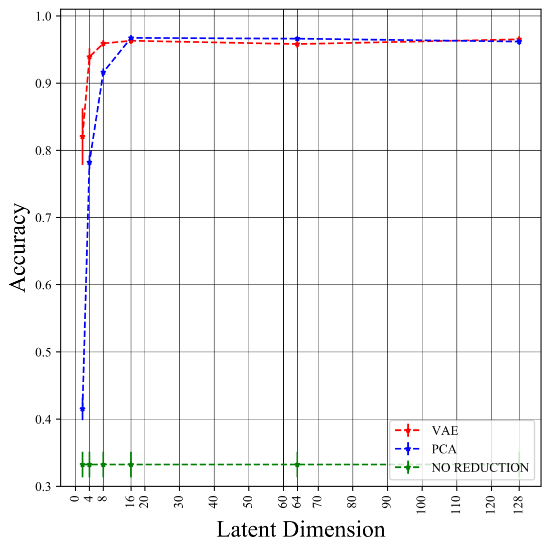


(d) CWR Spectrograms - MLP1LDO - $\varepsilon = 100\%$

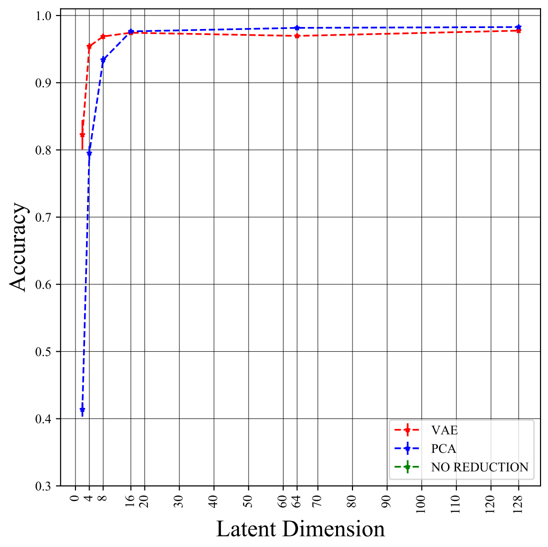
Figure 7.2: Average accuracy versus latent space dimension for the CWR dataset and MLP1LDO classifier with spectrograms.



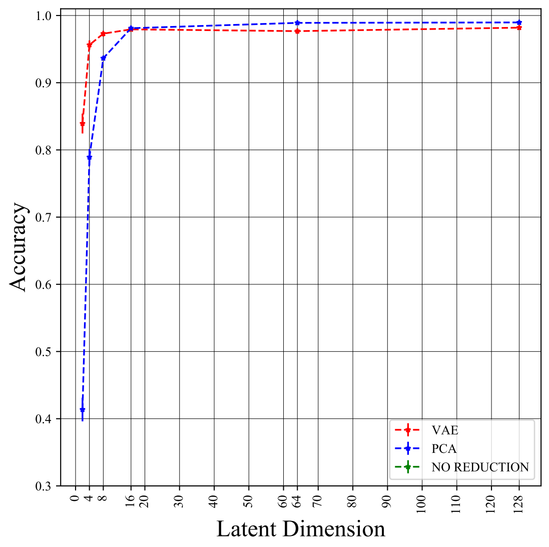
(a) CWR Spectrograms - MLP3LDO - $\varepsilon = 1\%$



(b) CWR Spectrograms - MLP3LDO - $\varepsilon = 5\%$



(c) CWR Spectrograms - MLP3LDO - $\varepsilon = 25\%$



(d) CWR Spectrograms - MLP3LDO - $\varepsilon = 100\%$

Figure 7.3: Average accuracy versus latent space dimension for the CWR dataset and MLP3LDO classifier with spectrograms.

### 7.1.2 Case Study #2: MFPT

The fault diagnostics average accuracies and corresponding standard deviations are shown in Figure 7.4, Figure 7.5 and Figure 7.6 for the single layer with no dropout (MLP1L) and with dropout architecture (MLP1LDO) as well as the three-layer with dropout (MLP3L), respectively, and also in Table Table B.1 of the Appendix. The best results are shown in 7.2. Even though this dataset has a considerable lower dimensionality than the CWR case, the partial overlap between the baseline and the outer race fault makes the CWR case study more challenging, as it was shown in Figure 5.2. Indeed, the fault diagnosis accuracies are significantly lower than the results obtained in the previous section for the CWR study case. However, fault diagnosis based on the proposed VAE architecture outperforms the classification results from the PCA based representation for low dimensions of the latent space, i.e.,$k = \{2, 4\}$ , as well as for situations where there is scarce labeled data corresponding to $\varepsilon = \{1\%, 5\%\}$. For example, for $\varepsilon = \{5\%\}$, the best accuracy of 92.15% based on the VAE representation is obtained with the single layer with dropout architecture (MLP1LDO) for dimension of $k = 4$ whereas the best accuracy of 90.61% for the PCA based representation is for dimension of $k = 8$ and MLP1LDO architecture.

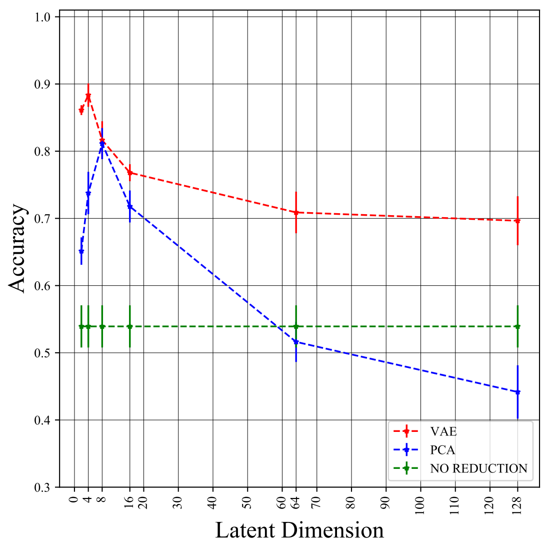
Table 7.2: Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the MFPT datasets are preprocessing applying the STFT and then scaling down the resulting spectrograms to images of 96 by 96 pixels.

| | **MFPT - Spectrogram Images** | | | | | |
|---|---|---|---|---|---|---|
| | **VAE** | **PCA** | **Baseline** | **VAE** | **PCA** | **Baseline** |
| | $\varepsilon = 100\%$ | | | $\varepsilon = 25\%$ | | |
| Classifier | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP1L |
| $k$ | 128 | 128 | - | 128 | 8 | - |
| Accuracy | 93.91% ± 0.30% | 93.44% ± 0.68% | 90.92% ± 0.27% | 92.82% ± 0.31% | 92.41% ± 0.32% | 87.89% ± 0.47% |
| | $\varepsilon = 5\%$ | | | $\varepsilon = 1\%$ | | |
| Classifier | MLP1LDO | MLP1LDO | MLP1L | MLP1LDO | MLP1LDO | MLP3LDO |
| $k$ | 4 | 8 | - | 4 | 8 | - |
| Accuracy | 92.15% ± 0.41% | 90.61% ± 0.76% | 79.25% ± 1.16% | 89.09% ± 1.39% | 87.10% ± 1.47% | 54.51% ± 1.41% |

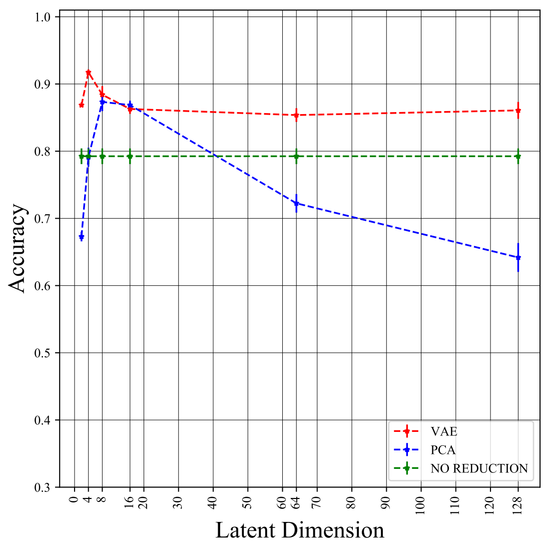
Also note that, differently from the CWR case study, now the fault diagnosis accuracy based on the PCA based dimensionality reduction achieves the best scores with the simpler one hidden layer with dropout architecture (MLP1LDO) for all levels of labeled data ($\varepsilon = \{1\%, 5\% 25\%, 100\%\}$). These results seem to indicate that the low dimension (health states) of the original problem tends to facilitate the task of the PCA in providing a satisfactory data representation for fault diagnosis. The same is observed for the accuracies based on the VAE dimensionality reduction.

However, note that the representation provided by the proposed VAE architecture seems to be less sensitive to the increased complexity (and level of help) provided by the architecture of the fault classifier. For instance, the accuracy is 89.09% with the MLP1LDO and 88.63% with the MLP3L, both results for $\varepsilon = \{1\%\}$ and $k = 4$ as shown in Table B.1 of the Appendix. Note also in 7.2 that the best fault classification accuracy based on the VAE representation is achieved for a lower dimensionality ($k = 4$) than in the case of the PCA ($k = 8$) when the amount of labeled data is low $\varepsilon = \{1\%, 5\%\}$. In other words, the results show that VAE representations of the data are more stable across different dimensions of the latent space but the fault diagnosis accuracies based on VAE and PCA are comparable in terms of classifier complexity when the dimensionality is $k > 8$.

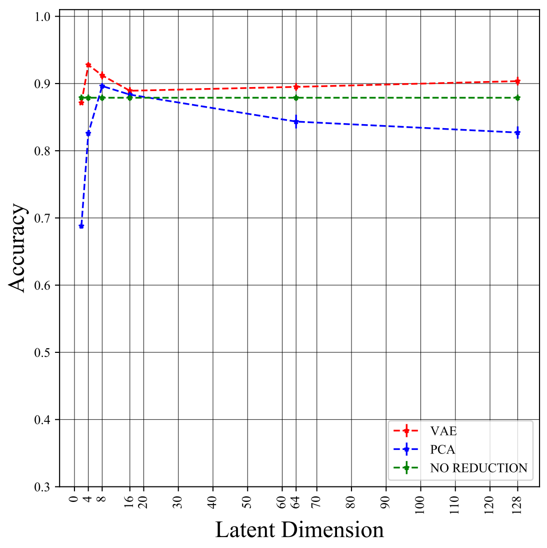
In summary, based on the results for the MFPT dataset, the VAE and PCA based data representations provide comparable results in terms of the accuracies for the fault diagnosis task, even though the VAE based representation leads to improved fault diagnosis accuracies when dealing with reduced size labeled data ($\varepsilon = \{1\%, 5\%\}$) and for lower dimensions of the latent space ($k = \{2, 4\}$ ).



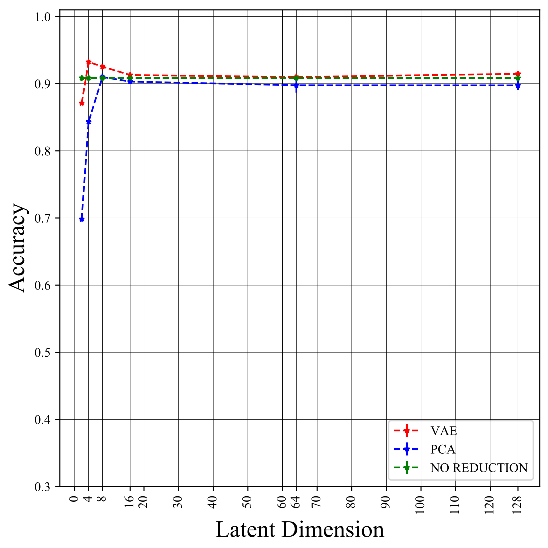
(a) MFPT Spectrograms - MLP1L - $\varepsilon = 1\%$



(b) MFPT Spectrograms - MLP1L - $\varepsilon = 5\%$

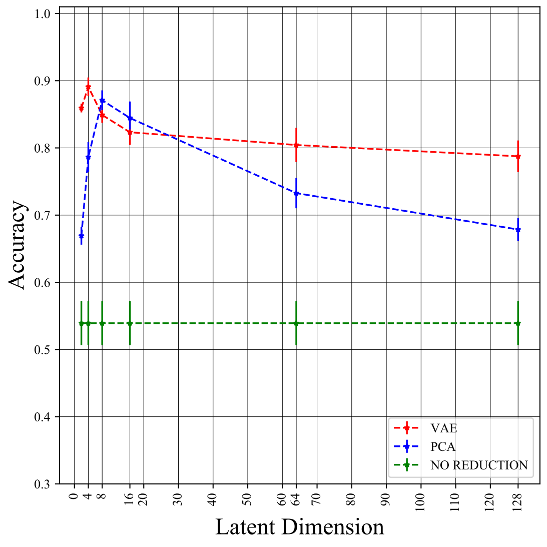


(c) MFPT Spectrograms - MLP1L - $\varepsilon = 25\%$

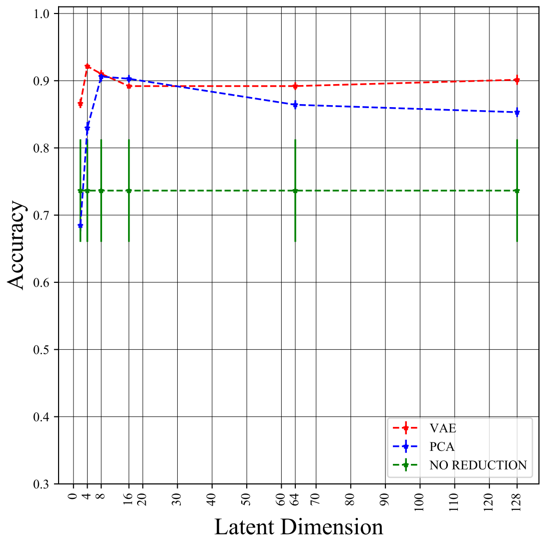


(d) MFPT Spectrograms - MLP1L - $\varepsilon = 100\%$

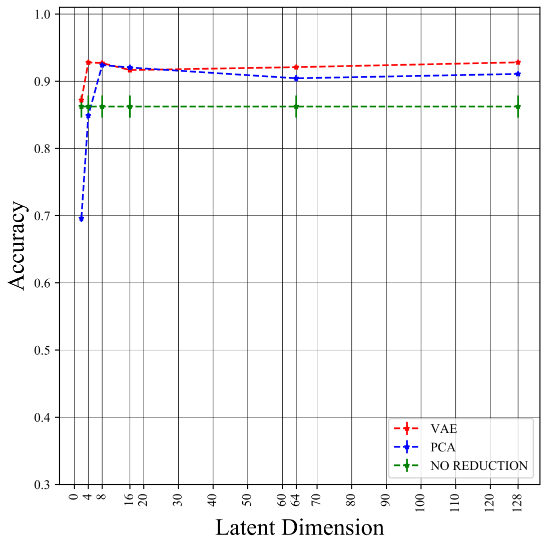
Figure 7.4: Average accuracy versus latent space dimension for the MFPT dataset and MLP1L classifier with spectrograms.



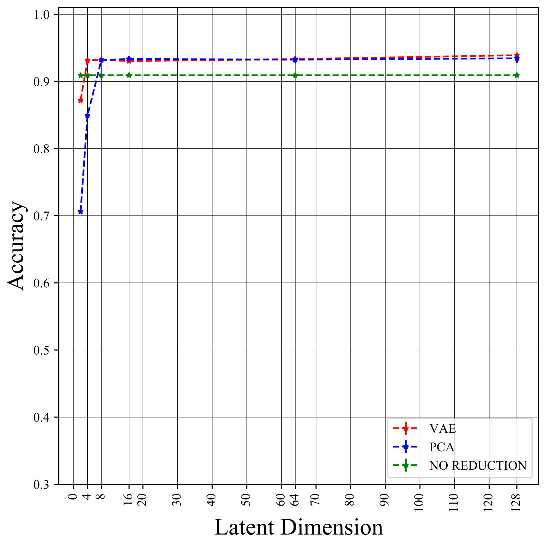
(a) MFPT Spectrograms - MLP1LDO - $\varepsilon = 1\%$



(b) MFPT Spectrograms - MLP1LDO - $\varepsilon = 5\%$

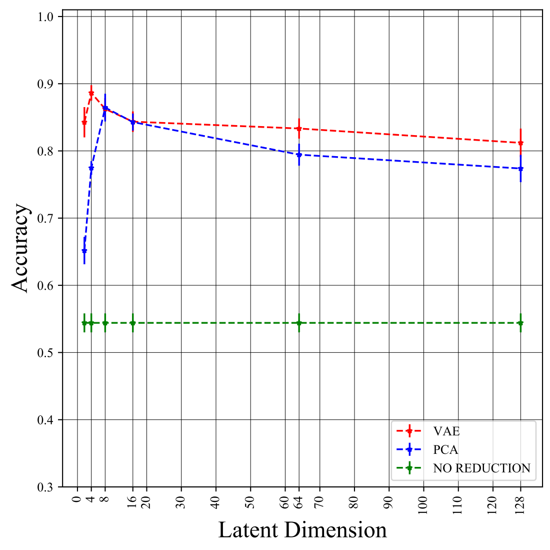


(c) MFPT Spectrograms - MLP1LDO - $\varepsilon = 25\%$

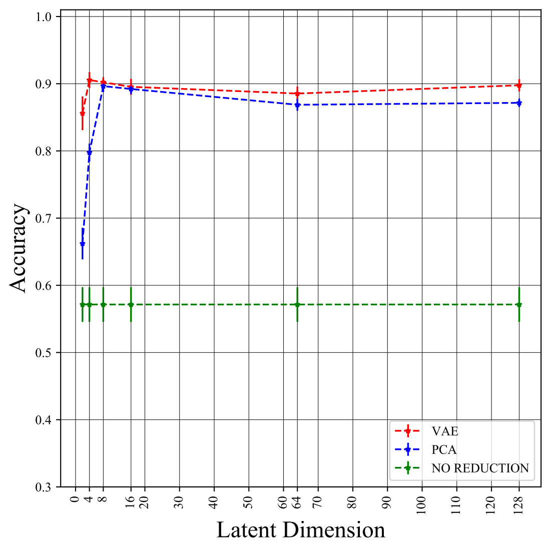


(d) MFPT Spectrograms - MLP1LDO - $\varepsilon = 100\%$

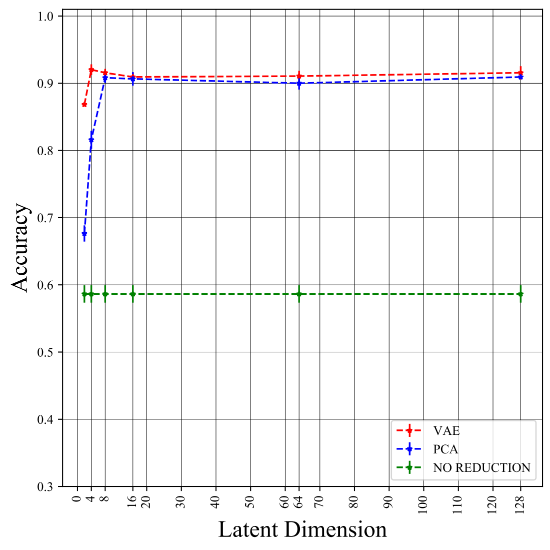
Figure 7.5: Average accuracy versus latent space dimension for the MFPT dataset and MLP1LDO classifier with spectrograms.



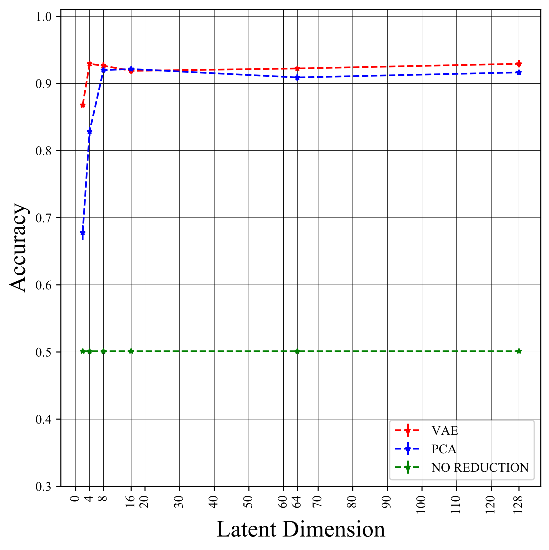
(a) MFPT Spectrograms - MLP3LDO - $\varepsilon = 1\%$



(b) MFPT Spectrograms - MLP3LDO - $\varepsilon = 5\%$



(c) MFPT Spectrograms - MLP3LDO - $\varepsilon = 25\%$



(d) MFPT Spectrograms - MLP3LDO - $\varepsilon = 100\%$

Figure 7.6: Average accuracy versus latent space dimension for the MFPT dataset and MLP3LDO classifier with spectrograms.

## 7.2 Manually Extracted Features.

Now, this thesis explore the proposed VAE based dimensionality reduction when dealing with a typical situation from a real-world application perspective: manually extracted features from vibration signals. Based on the same fault diagnosis classifiers discussed in Section 6, both CWR and MFPT case studies are also analyzed and contrasted to the diagnosis results obtained based on spectrograms presented in Sections 7.1.1 and 7.1.2.

### 7.2.1 Case Study #1: CWR

Table 7.3 shows the best fault diagnosis accuracies results for the CWR case study using manually extracted features as dataset. Note that when the entire dataset is used, $\varepsilon = \{100\%\}$, the accuracy results are very similar for the VAE based and PCA based dimensionality reductions as well as when no reduction is performed. This can be explained from the fact that the input data corresponds to manually extracted features, so performing a reduction might not significantly improve the fault classification as useful information is already extracted from the original datasets.

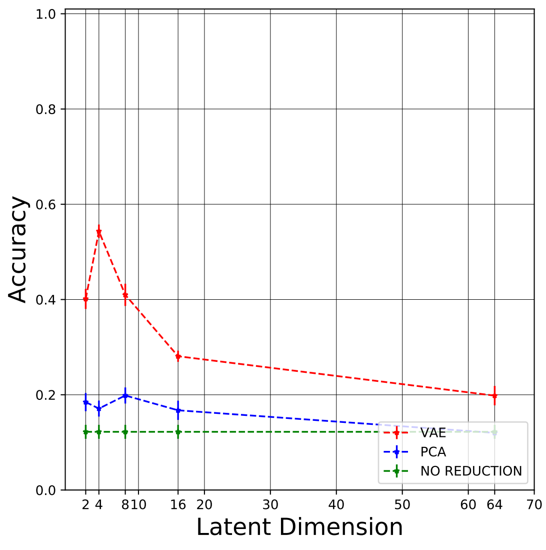
Table 7.3: Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the CWR dataset is preprocessing doing a manual feature extraction.

| | **CWR - Features Extracted Manually** | | | | | |
|---|---|---|---|---|---|---|
| | **VAE** | **PCA** | **Baseline** | **VAE** | **PCA** | **Baseline** |
| | $\varepsilon = 100\%$ | | | $\varepsilon = 25\%$ | | |
| Classifier | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO |
| $k$ | 64 | 64 | - | 64 | 64 | - |
| Accuracy | 95.25% ± 0.25% | 95.64% ± 0.53% | 95.59% ± 0.25% | 89.49% ± 0.94% | 81.66% ± 0.87% | 79.85% ± 1.25% |
| | $\varepsilon = 5\%$ | | | $\varepsilon = 1\%$ | | |
| Classifier | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP3LDO | MLP3LDO |
| $k$ | 8 | 16 | - | 4 | 16,00% | - |
| Accuracy | 67.13% ± 1.56% | 43.45% ± 1.61% | 39.11% ± 0.79% | 55.28% ± 2.65% | 25.37% ± 2.09% | 26.09% ± 2.12% |

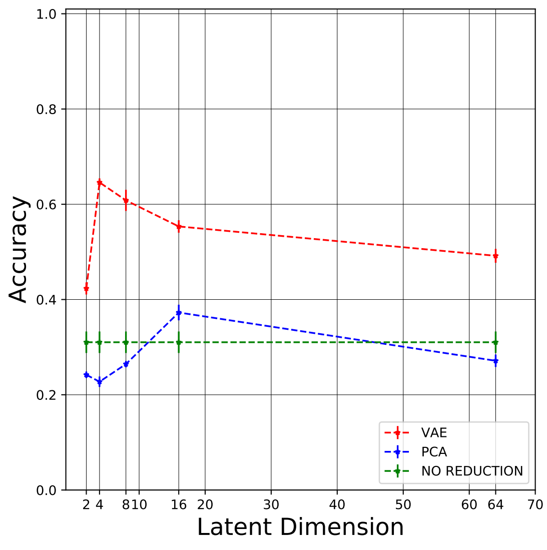
Now, for $\varepsilon = \{1\%, 5\%\}$, accuracy significantly drops across the board, but the decline is more significant for the PCA based data representation and for the baseline case. However, for the this repository and the mentioned values for $\varepsilon$, none of the fault classifiers deliver satisfactory results as the best one, based on the proposed VAE architecture, is barely superior to 65%. Maybe the the most extreme case is for $\varepsilon = \{1\%\}$, where the best results is barely superior to 50%, as for when spectrogram images are used, the top result for that value of $\varepsilon$ is above 90% as shown in Table 7.1.

Even for the cases where more labeled data is available ($\varepsilon = \{25\%, 100\%\}$), when compared to the spectrogram based fault diagnosis (see Section 7.1.1), the use of hand-engineered features in the CWR case study yields worse results. For instance, for $\varepsilon = \{100\%\}$, the best accuracy is equal to 95.64% while for the former case (spectrogram images) the maximum accuracy is above 98%. When $\varepsilon = \{25\%\}$, hand-engineered features perform significantly worse in accuracy compared to spectrograms images for which the best fault identification accuracies are above 98% for both VAE and PCA based dimensionality reduction, as shown in Table 7.1, while for features, the accuracies do not reach the 90% value for the accuracy.

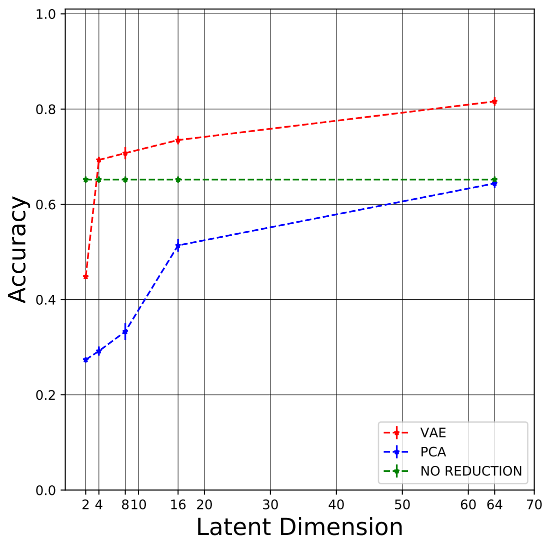
As a final comment, it is worth observing that, based on Figures 7.7, 7.8 and 7.9, the VAE based data representation consistently leads to better fault diagnosis accuracies across classifiers in situations where the amount of data with identified health states (labeled data) is small, i.e. $\varepsilon = \{1\%, 5\%, 25\%\}$, even though the results themselves are not ideal. For the case where the full dataset is used for the training of the classifier, which is the case that yields aceptable accuracies results, VAE based dimensionality reduction also show better or similar results across dimensions than PCA based reduction, but in the MLP1L and MLP1LDO classifiers, it is comparable with the case where no reduction is performed if the dimension is chosen correctly, which is obviously a disadvantage since it's require the optimization of the VAE model's latent space. The complete results for every choice of $k$, $\varepsilon$ and classifier architecture are reported in Table B.2 in the Appendix.



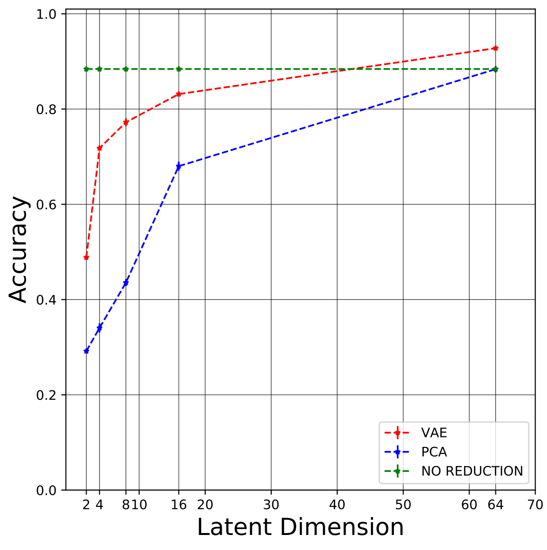
(a) CWR Features - MLP1L - $\varepsilon = 1\%$



(b) CWR Features - MLP1L - $\varepsilon = 5\%$

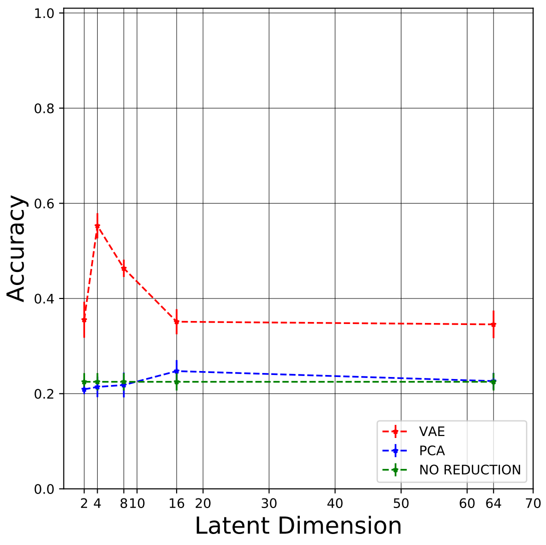


(c) CWR Features - MLP1L - $\varepsilon = 25\%$

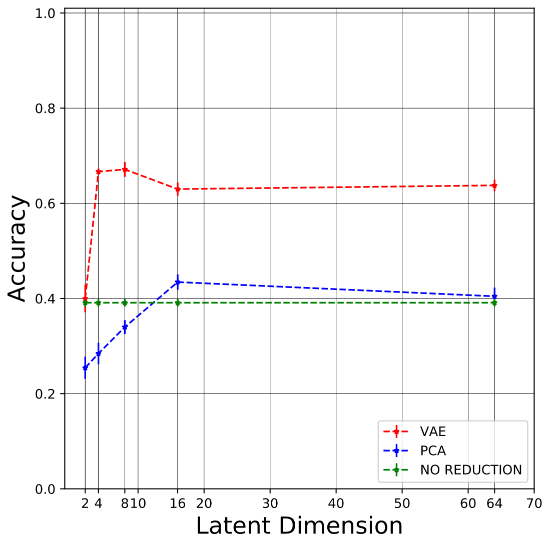


(d) CWR Features - MLP1L - $\varepsilon = 100\%$

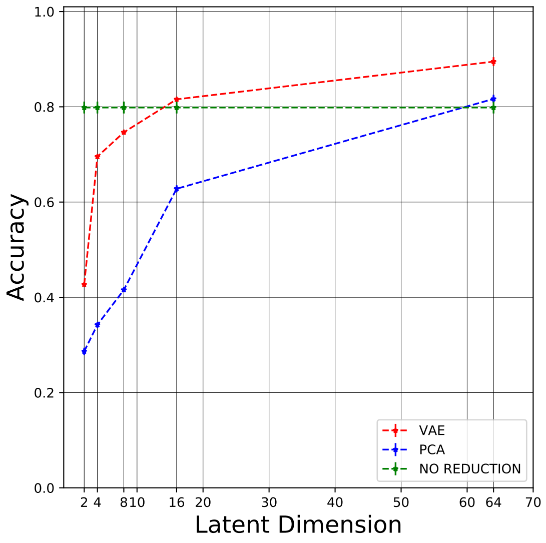
Figure 7.7: Average accuracy versus latent space dimension for the CWR dataset and MLP1L classifier with manually extracted features.



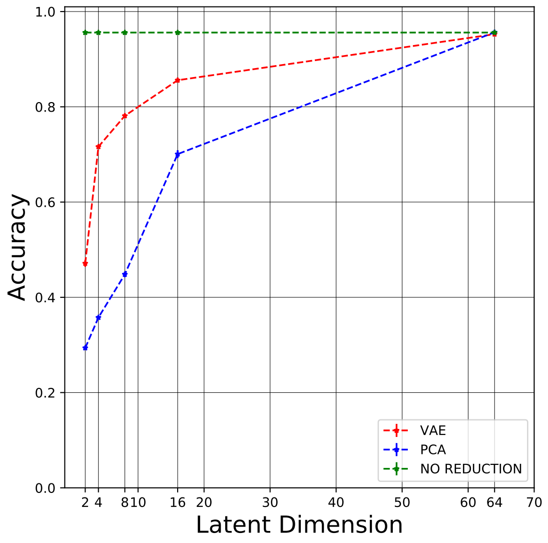
(a) CWR Features - MLP1LDO - $\varepsilon = 1\%$



(b) CWR Features - MLP1LDO - $\varepsilon = 5\%$

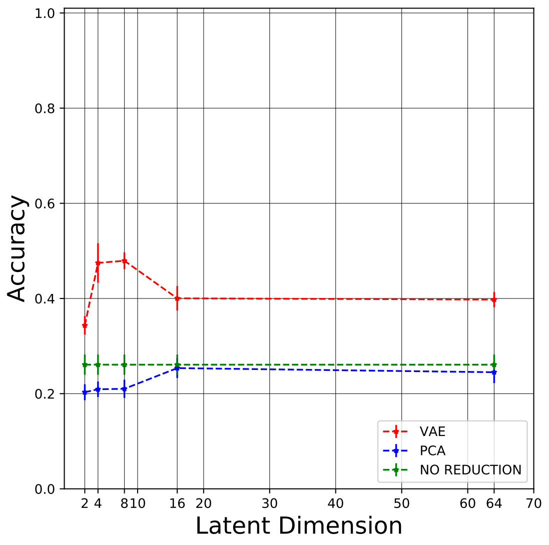


(c) CWR Features - MLP1LDO - $\varepsilon = 25\%$

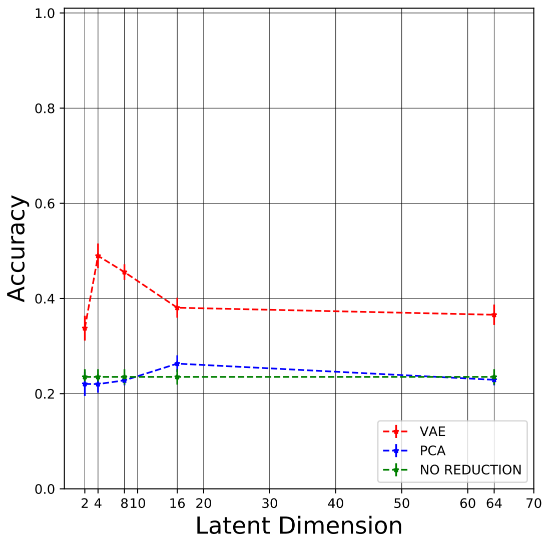


(d) CWR Features - MLP1LDO - $\varepsilon = 100\%$

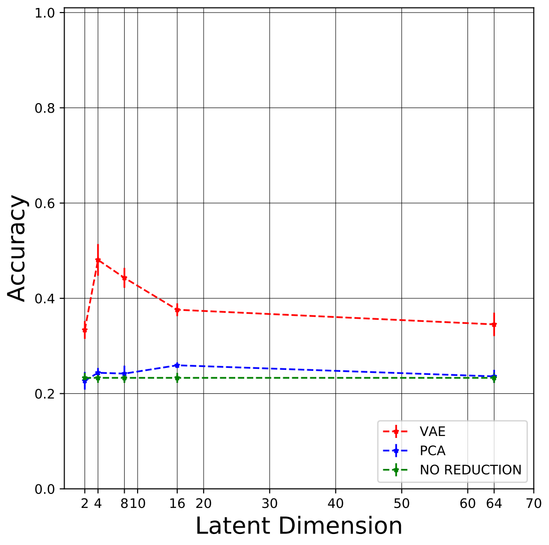
Figure 7.8: Average accuracy versus latent space dimension for the CWR dataset and MLP1LDO classifier with manually extracted features.



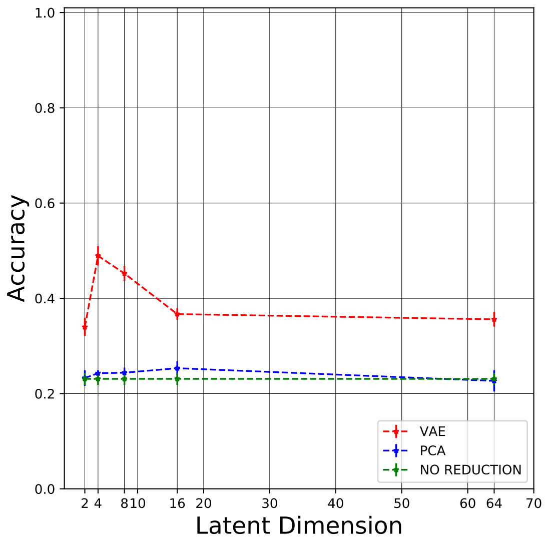
(a) CWR Features - MLP3LDO - $\varepsilon = 1\%$



(b) CWR Features - MLP3LDO - $\varepsilon = 5\%$



(c) CWR Features - MLP3LDO - $\varepsilon = 25\%$



(d) CWR Features - MLP3LDO - $\varepsilon = 100\%$

Figure 7.9: Average accuracy versus latent space dimension for the CWR dataset and MLP3LDO classifier with manually extracted features.

### 7.2.2 Case Study #2: MFPT

In the context of the MFPT case study, Table 7.3 shows the best fault diagnosis accuracies results when manually extracted features are used as dataset. In a similar way than with the CWR case study, note that when the entire dataset is used, $\varepsilon = \{100\%\}$, accuracies results for the VAE approach, the PCA approach and the case where no dimensionality is performed, are extremely similar. Again, this can be explained from the fact that the input data are features extracted manually, so they already represent useful information contained in the vibration signals.

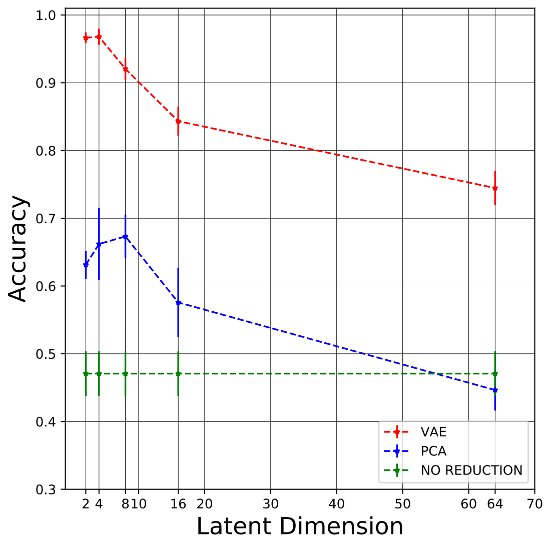
Table 7.4: Best accuracy results and the conditions (classifier and dimension) under they were obtained when the data from the MFPT dataset is preprocessing doing a manual feature extraction.

| | **MFPT - Features Extracted Manually** | | | | | |
|---|---|---|---|---|---|---|
| | **VAE** | **PCA** | **Baseline** | **VAE** | **PCA** | **Baseline** |
| | $\varepsilon = 100\%$ | | | $\varepsilon = 25\%$ | | |
| Classifier | MLP1LDO | MLP1LDO | MLP1LDO | MLP3LDO | MLP1LDO | MLP1LDO |
| $k$ | 16 | 64 | - | 64 | 64 | - |
| Accuracy | 99.57% ± 0.17% | 99.62% ± 0.14% | 99.62% ± 0.19% | 99.58% ± 0.15% | 98.07% ± 0.39% | 98.05% ± 0.17% |
| | $\varepsilon = 5\%$ | | | $\varepsilon = 1\%$ | | |
| Classifier | MLP3LDO | MLP3LDO | MLP3LDO | MLP1L | MLP3LDO | MLP3LDO |
| $k$ | 64 | 64 | - | 4 | 16 | - |
| Accuracy | 98.66% ± 0.45% | 91.78% ± 0.86% | 93.04% ± 0.89% | 96.79% ± 1.19% | 75.69% ± 2.67% | 69.39% ± 2.60% |

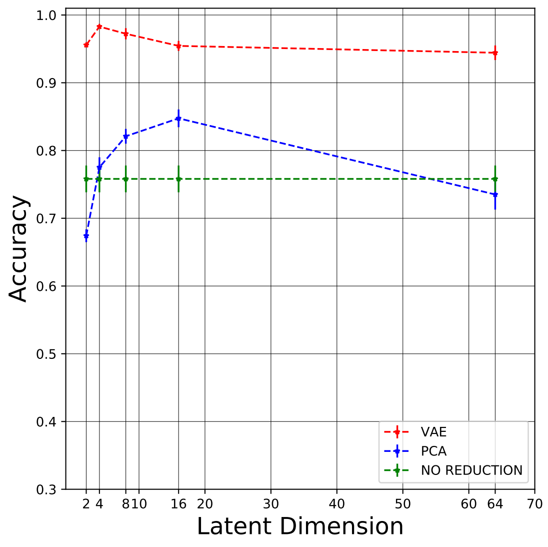
For this case study, contrary with what happen in the CWR data repository, features extracted manually delivers in general better results than ones obtained when spectrogram images are used. This can be seen in the fact that the best VAE and PCA accuracy results for this preprocessing method are 99.57% and 99.62% respectively (as shown in Table 7.4), while for spectrogram images are 93.91% and 93.44% respectively (See Table 7.2 for details). This increment in nearly 6% in the accuracy results shows clearly that for the MFPT dataset, features extracted manually represents better the health conditions present in the ball bearing.

Note also from Table 7.4 that the accuracy results for PCA and the case where no reduction is performed drops significantly when the amount of labeled data is low ($\varepsilon = \{1\%, 5\%\}$), while for the VAE approach the drop in performance is much more subtle. This corroborates the results obtained in Section 7.1, that for cases where the amount of labeled data is reduced, VAEs achieve better results than the other two approaches.

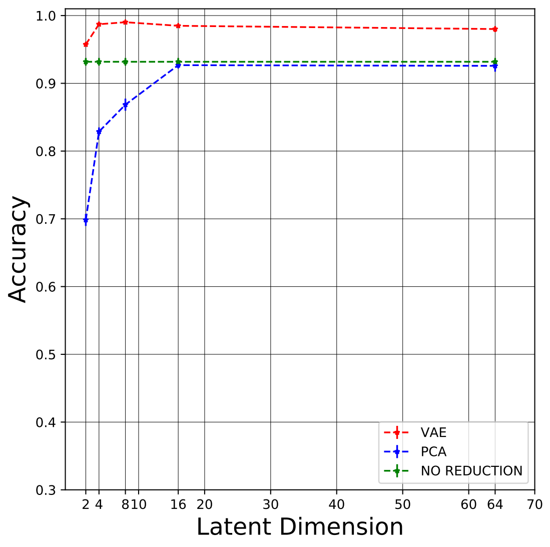
From Figures 7.10, 7.11 and 7.12, one can notice two main aspects. First, the general stability of the VAE approach to the change in the value of $k$, the latent space dimensionality, is greater than for the PCA approach, across sizes of the training dataset and classifiers. Second, in the same manner as before with the spectrograms dataset, the VAE model produces a stronger representation than PCA when the level of compression is drastic, for example, $k = \{2, 4\}$. This can be seen from the fact that VAE shows accuracies that are in general between 30% and 15% better than PCA for $k = 2$ and $k = 4$ respectively. As a final comment, it can be seen from 7.4 and the figures below, that the VAE approach shows for this case study and preprocessing method better performance than the other two approaches, specially when dealing with low latent dimension or reduced amounts of labeled data.



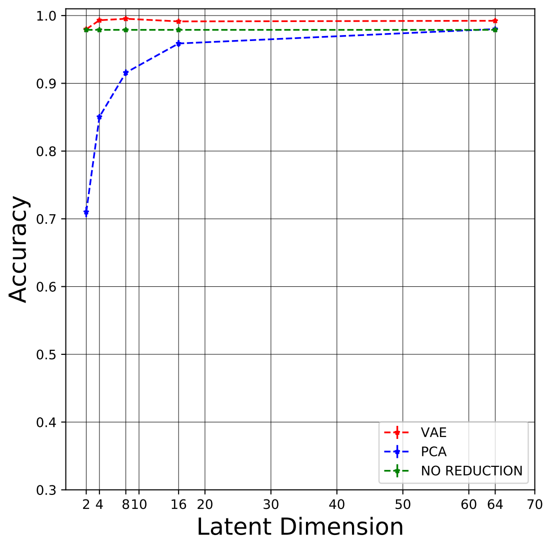
(a) MFPT Features - MLP1L - $\varepsilon = 1\%$



(b) MFPT Features - MLP1L - $\varepsilon = 5\%$

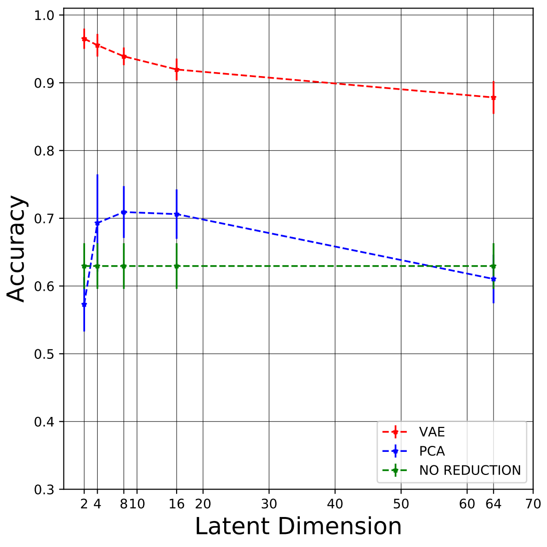


(c) MFPT Features - MLP1L - $\varepsilon = 25\%$

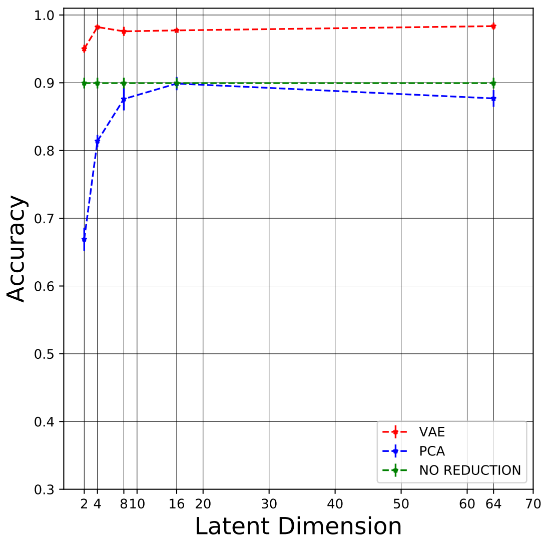


(d) MFPT Features - MLP1L - $\varepsilon = 100\%$

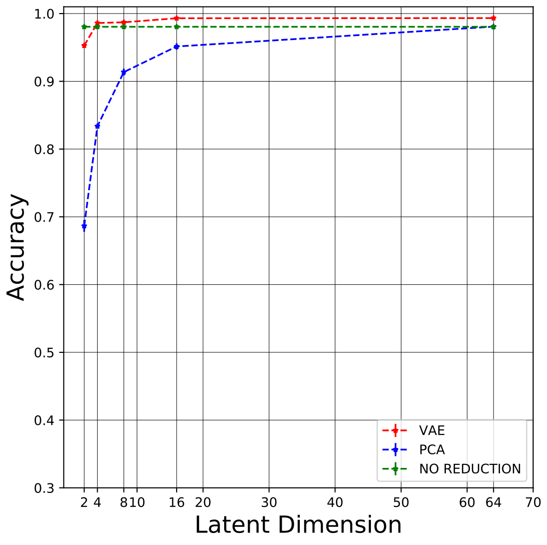
Figure 7.10: Average accuracy versus latent space dimension for the MFPT dataset and MLP1L classifier with manually extracted features.



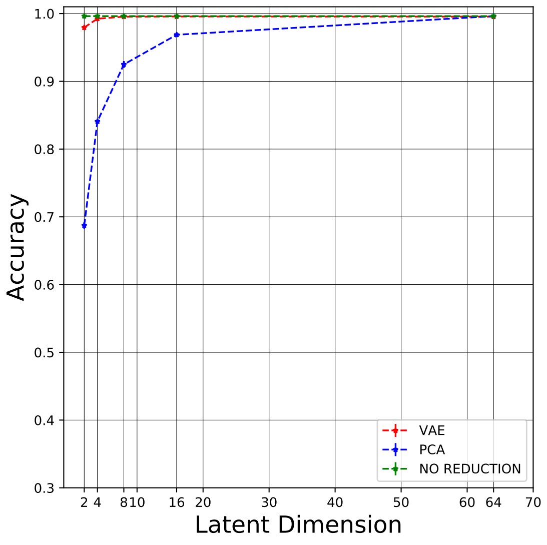
(a) MFPT Features - MLP1LDO - $\varepsilon = 1\%$



(b) MFPT Features - MLP1LDO - $\varepsilon = 5\%$

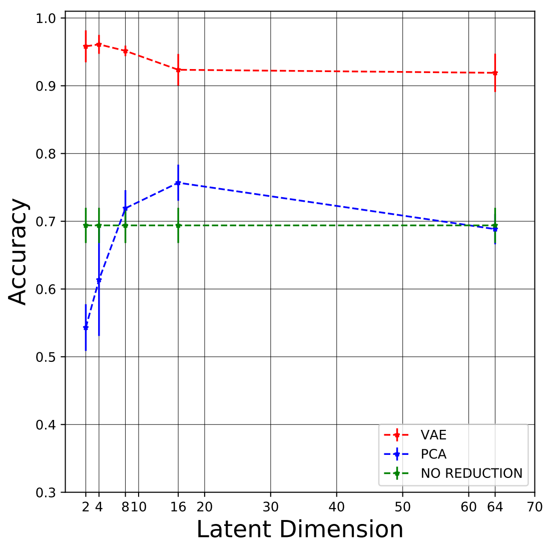


(c) MFPT Features - MLP1LDO - $\varepsilon = 25\%$

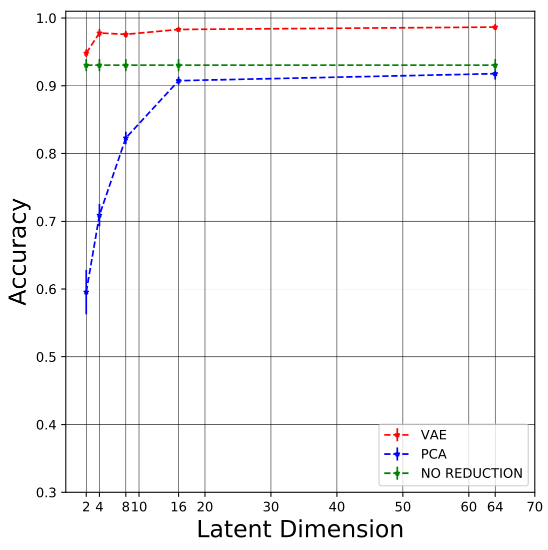


(d) MFPT Features - MLP1LDO - $\varepsilon = 100\%$

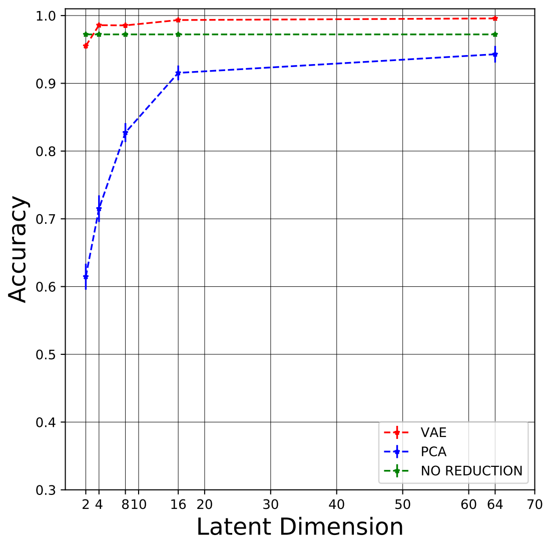
Figure 7.11: Average accuracy versus latent space dimension for the MFPT dataset and MLP1LDO classifier with manually extracted features.



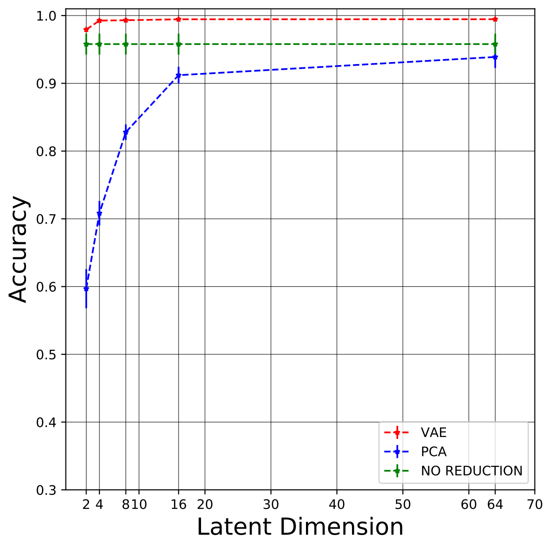
(a) MFPT Features - MLP3LDO - $\varepsilon = 1\%$



(b) MFPT Features - MLP3LDO - $\varepsilon = 5\%$



(c) MFPT Features - MLP3LDO - $\varepsilon = 25\%$



(d) MFPT Features - MLP3LDO - $\varepsilon = 100\%$

Figure 7.12: Average accuracy versus latent space dimension for the MFPT dataset and MLP3LDO classifier with manually extracted features.

## 7.3 Raw Vibration Signal.

This section presents the results based on the use of raw vibration data to train both dimensionality reduction methods. Recall from Section 5.3 that the architectures of the encoder and decoder neural networks vary for the CWR and MFPT cases: architecture #8 for CWR and architecture #9 for the MFPT (see Table 5.3 for details). The best fault identification accuracies under each dimensionality reduction approach and also for the baseline are shown in 7.5. For the sake of brevity, and also because the results obtained with the use of the raw vibration signals are far worse than those obtained with any of the other two preprocessing methods for both case studies (CWR and MFPT), this section focus the discussion on the extreme cases where the percentage of available labeled data is $\varepsilon = \{1\%, 100\%\}$.Nevertheless, the complete set of results for every choice of $k$, $\varepsilon$ and classifier are shown in Table B.3 of the Appendix.

Table 7.5: Best accuracy results and the conditions (classifier and dimension) under they were obtained when the raw data from the CWR and MFPT datasets is used without any kind of preprocessing.

| | **CWR - Raw Data** | | | **MFPT - Raw Data** | | |
|---|---|---|---|---|---|---|
| | **VAE** | **PCA** | **No Reduction** | **VAE** | **PCA** | **No Reduction** |
| | $\varepsilon = 100\%$ | | | | | |
| Classifier | MLPLDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO | MLP1LDO |
| $k$ | 64 | 128 | - | 128 | 128 | - |
| Accuracy | 64.09% ± 1.01% | 84.49% ± 0.56% | 81.91% ± 0.53% | 74.62% ± 0.49% | 89.92% ± 0.33% | 68.36% ± 0.36% |
| | $\varepsilon = 1\%$ | | | | | |
| Classifier | MLP1L | MLP1L | MLP3LDO | MLP1LDO | MLP1LDO | MLP3LDO |
| $k$ | 2 | 4 | - | 8 | 2 | - |
| Accuracy | 53.28% ± 0.54% | 40.37% ± 1.54% | 30.93% ± 4.34% | 64.94% ± 0.91% | 47.52% ± 1.87% | 46.42% ± 1.26% |

The accuracy results are significantly worse than those obtained with spectrograms and with manually extracted features. Nevertheless, notice that the best accuracy is achieved by the single hidden layer with dropout architecture based on the PCA data representation when dealing with the full dataset for both the CWR and MFPT case studies, reaching an accuracy of 89.92% in the classification of health states for the MFPT and 84.49% accuracy in the CWR dataset. However, when $\varepsilon = \{1\%\}$ opposite results are obtained: 64.94% accuracy in the MFPT and 53.28% in the CWR, both results based on the VAE data representation. Although these results are arguably of no usability for fault diagnostics purposes and are far worse than those obtained in Section 7.1 and 7.2, it is observed again that the proposed VAE architecture provides a consistently better dimensionality reduction for fault diagnosis when processing datasets with a reduced amount of labeled data. A similar set of plots as the ones shown in the previous section for the use of spectrogram images and features extracted manually can be found in the Appendix for the case where the raw vibration signal is used to conform the dataset in both CWR (Figures C.1, C.2 and C.3) and MFPT (Figures C.4, C.5 and C.6) case studies.

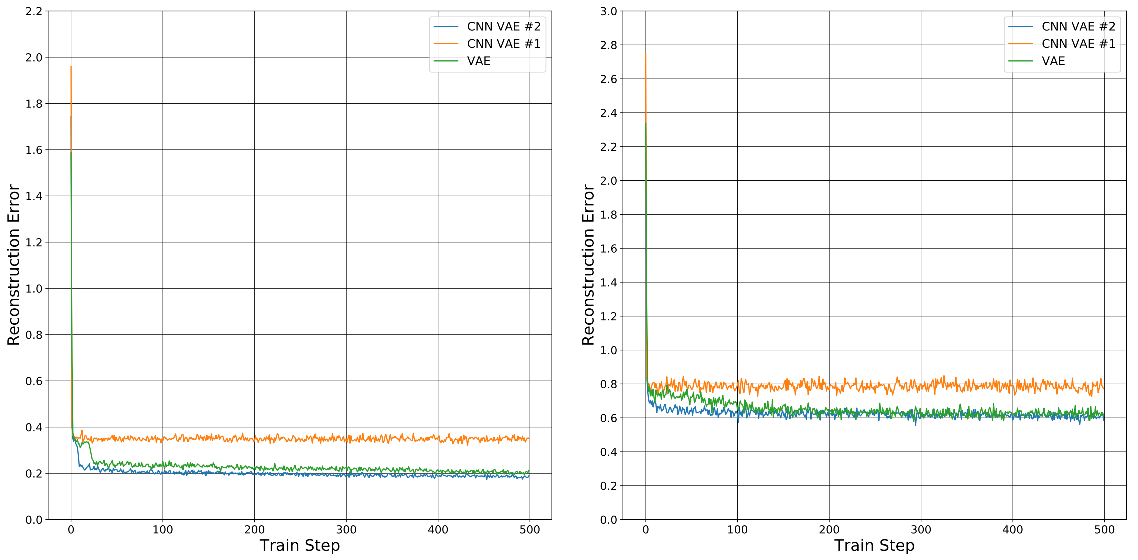
## 7.4 CNN Variational Auto Encoder

Convolutional neural networks (CNNs) have experienced significant popularity in recent years mainly for their capacity for automatically extracting abstract and hierarchical high-level features from images. In fact, to handle the complexity of image classification, CNNs are the dominant method [39] [40] [41] [42] [43] [44] surpassing their rival human accuracies for the same tasks [45]. Therefore, in this section it is proposed the use of deep convolutional neural networks in both the encoder and decoder of the VAE in lieu of the deep fully connected neural networks discussed in Section 5.3. The objective is to explore the potential of this CNN-VAE approach using the same cases discussed in Section 7.1, i.e., CWR and MFPT datasets with spectrograms images.

Two different architectures are explored for the CNN-VAE approach. The first architecture is inspired by Masci et al (2011) [46], where CNNs are used in the context of traditional (i.e., non variational) auto-encoders (AEs) and denoising auto-encoders (DAEs, which are also non-variational) to perform classification tasks with the MNIST and CIFAR-10 datasets. These AEs use a neural network as encoder to transform the data to a latent representation and then uses the same neural network, but with the matrices of weights transposed (called decoder) to reconstruct the original data from this latent representation. Denoising auto-encoders use the same principle but the input is contaminated with noise prior to being fed to the auto-encoder. Then, the DAE is trained to reconstruct a clean input from a partially noisy one. All the transformations that AE and DAE perform on the data are deterministic, i.e., no stochasticity is involved other than the noise added to the DAE.

VAEs, on the other hand, execute the same process of compressing and decompressing the data from a latent representation, but searching for a probability distribution that best explains the nature of the latent variables controlling the problem (see Section 3.7). Therefore, its not possible to use the same architecture used in [46] in the proposed CNN-VAE approach. The main change to the architecture is that the weights of the decoder's convolutional neural network are not the transposed weights of the encoder, since the process of decompressing the data is not the direct inverse of the compression process.

Indeed, the first architecture (architecture #1) is as follows. The encoder consists of six hidden layers: 1) the input is feed to a convolutional layer with 100 filters of size 5x5; 2) a max pooling layer is applied with size 2x2 and stride 2x2 to process non-overlapping regions; 3) a convolutional layer with 150 filters of size 5x5; 4) another max pooling layer with the same characteristics as before; 5) a convolutional layer with 200 filters of size 3x3; and 6) a fully connected layer with 300 units whose output is fed to the two outputs layers of the encoder of the architecture discussed in Section 5.3 and shown in in Figure 5.4. Now, the decoder consists of five hidden layers: 1) the latent variables are fed to a fully connected layer with 300 units; 2) the outputs of that first hidden layer are fed to another fully connected layer with 200x24x24 units, which are then reshaped to form 200 features maps of size 24x24; 3) a deconvolutional layer of 150 filters with size 3x3; 4) a deconvolutional layer of 100 filters with size 5x5; and 5) a final deconvolutional layer of 1 filter of size 5x5. The outputs of this last deconvolutional layer are fed to an output layer to produce the vector of Bernoulli parameters $\vec{\rho}$ (see Section 3.7.2). The weights of the convolutional, deconvolutional and fully connected layers are initialized with samples of a truncated Normal distribution of mean equals to zero and standard deviation of 0.1. The biases are initialized as constants with a value equal to 0.1.



(a) Reconstruction error for the CWR dataset. (b) Reconstruction error for the MFPT dataset.

Figure 7.13: Reconstruction errors of the deep neural network VAE model discussed in Section 5.3 and the CNN-VAE model with both proposed architectures and for the (a) CWR dataset and (b) MFPT dataset.

With this architecture, the reconstruction error (taken as the mean square error between the original input and the reconstructed image) of the CNN-VAE for both the CWR and MFPT datasets did not show any improvement over the VAE model discussed in Section 5.3, as it can be seen in Figure 19 (a) and (b). As a result, the obtained dimensionality reduction was very deficient, which resulted in classifiers with poor health state classification accuracies: best result below 50%.

To address these deficiencies in the context of fault diagnosis, this thesis proposes another architecture based on the previous one, but with reduced number of filters per convolutional (and therefore, deconvolutional) layer. This second architecture (architecture #2) is shown in Figure 7.14 and Figure 7.15 for the CNN VAE's encoder and decoder, respectively. Due to its lower number of filters, this architecture has less learnable parameters, thus being cheaper to train in terms of computational resources. Parameter initialization is performed in the same way as the previous architecture. As it can be seen in Table 7.6, the training of the CNN-VAE with architecture #2 takes 2.94 seconds per epoch for the CWR dataset and 2.31 seconds per epoch for the MFPT dataset compared to 22.06 seconds and 17.38 seconds per epoch, respectively, for the first architecture. Moreover, architecture #2 shows a reduction in the reconstruction error as the training progresses, resulting in error values similar to ones produced by the VAE discussed Section 5.3 in the case of the MFPT dataset, and better in the case of the CWR dataset, as shown in Figure 7.13a and 7.13b.

Table 7.7 shows the best fault diagnosis accuracy results for both the CWR and MFPT datasets with $\varepsilon = \{1\%, 5\%, 25\%, 100\%\}$ and based on the data representation from the CNN-VAE architecture #2. The complete set of results are reported in Table B.4 in the Appendix.

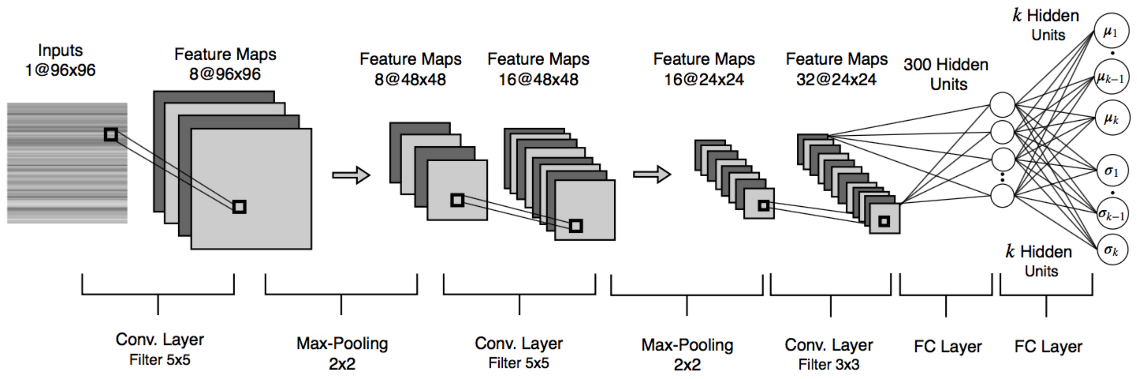


Figure 7.14: Encoder of the CNN-VAE comprised of three convolutional layers, two max-pooling layers, one fully connected layer and two outputs layers (one for the vector of means and other for the vector of variances).

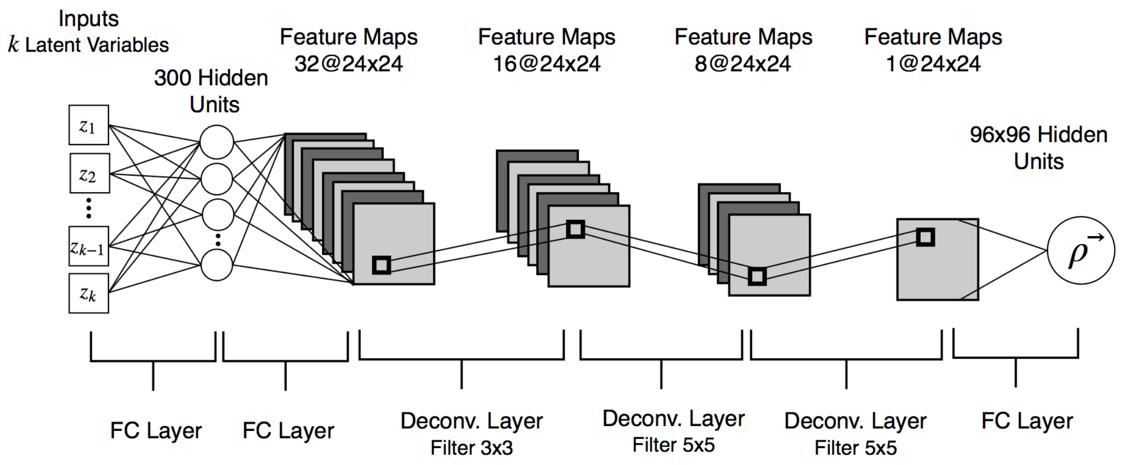


Figure 7.15: Encoder of the CNN-VAE comprised of three convolutional layers, two max-pooling layers, one fully connected layer and two outputs layers (one for the vector of means and other for the vector of variances).

Table 7.6: Training times per epoch and total for the VAE model shown in section 5.3 and the CNN-VAE model with both architectures tested. The training of all architectures is performed with a maximum of 500 epochs.

| | **Traditional VAE** | | **CNN-VAE Arch. #1** | | **CNN-VAE Arch. #2** | |
|---|---|---|---|---|---|---|
| | **CWR** | **MFPT** | **CWR** | **MFPT** | **CWR** | **MFPT** |
| **Time per epoch** | 0.66 [s] | 0.51 [s] | 22.06 [s] | 17.38 [s] | 2.94 [s] | 2.31 [s] |
| **Total Training time** | 330.5 [s] | 254.0 [s] | 11029.5 [s] | 8691.0 [s] | 1467.5 [s] | 1152.5 [s] |

Table 7.7: Better fault diagnosis accuracies and standard deviations based on the CNN-VAE architecture #2 for the CWR and MFPT datasets containing spectrograms.

| | **CWR - CNN VAE** | **MFPT - CNN VAE** |
|---|---|---|
| | $\varepsilon = 100\%$ | |
| Classifier | MLP1LDO | MLP1LDO |
| $k$ | 128 | 128 |
| Accuracy | 98.35% ± 0.21% | 92.58% ± 0.43% |
| | $\varepsilon = 25\%$ | |
| Classifier | MLP1LDO | MLP1LDO |
| $k$ | 64 | 128 |
| Accuracy | 97.84% ± 0.14% | 89.34% ± 0.32% |
| | $\varepsilon = 5\%$ | |
| Classifier | MLP1LDO | MLP1LDO |
| $k$ | 64 | 8 |
| Accuracy | 96.37% ± 0.34% | 85.99% ± 0.56% |
| | $\varepsilon = 1\%$ | |
| Classifier | MLP3LDO | MLP3LDO |
| $k$ | 128 | 4 |
| Accuracy | 89.53% ± 1.74% | 82.86% ± 1.48% |

The fault diagnosis results based on the dimensionality reduction from the CNN-VAE are similar to the ones obtained with the deep neural network VAE architecture discussed in Section 7.1. For the CWR dataset, in particular, the best accuracies archived are in a range of ±1% with respect to the results obtained in Section 7.1. For the MFPT, the results based on the CNN-VAE are approximately 1% worse for the case of $\varepsilon = 100\%$ than the ones obtained with the deep neural network VAE; for $\varepsilon = 1\%$, the fault diagnosis accuracies using CNN-VAE data representation are significantly worse: 78.30% accuracy against 89.09% for the deep neural network VAE (See Table 7.2.

Based on these results, the CNN-VAE approach has a significant competitive disadvantage in the fault diagnosis of ball bearings in comparison to the deep neural network VAE as it not only delivers accuracies that are equal or worse than the ones obtained with the deep neural network VAE, but also is considerably slower and computational more demanding for training.

# Chapter 8

# Concluding Remarks

Early damage detection has become an essential task in system health management, allowing engineers to prevent sudden failures in components, thus reducing costs by taking safety precautions. However, it is not always straightforward to analyze the available data, since many times relevant features must be manually extracted requiring expert knowledge and time to process. Moreover, a challenge usually faced when developing fault diagnosis tools based on machine learning techniques from sensor data is the curse of dimensionality: the need to deal with high dimensional data.

It is in this context that this thesis proposed an approach for dimensionality reduction in the context of fault diagnosis based on deep variational auto-encoders. Based on two case studies involving rolling bearing elements, CWR and MFPT, and three data types (spectrogram images, hand-engineered features and raw vibration signals), the proposed approach utilizes the latent representation delivered by a VAE model with custom encoder and decoder architectures as input to shallow neural network based health state classifiers.

Based on this approach, this thesis explored the capabilities of different VAE's architectures as a tool for dimensionality reduction and compared its performance in ball bearing fault diagnosis with a well-known dimensionality reduction method, PCA, and the case where no reduction is performed. The results have shown that improved accuracies in fault diagnosis are achieved by the proposed VAE based approach in comparison to PCA in cases where significant reduction is required, i.e., 2 or 4 dimensions. Both approaches, however, deliver similar performances for higher dimensionalities.

Also of practical interest is the case when the amount of labeled data is scarce but one has abundant unlabeled sensor data. In this context, the proposed VAE architectures are trained in a fully unsupervised way but the fault diagnosis is performed via shallow neural networks trained with a small subset of labeled data transformed using the latent representation provided by the VAE. For the CWR and MFPT datasets, the results have shown that fault diagnosis based on the VAE's data representation are superior to the ones obtained using PCA when scarce labeled data is available. When the amount of labeled data approximates to the full amount of data originally used for training the VAE, then fault classification accuracies are comparable when using both VAE's and PCA's data representations.

Besides the conventional deep neural networks, the use and capability of deep convolutional neural networks for the VAE's encoder and decoder were also explored. These CNN-VAE models delivered dimensionality reductions of the spectrogram based dataset that resulted in comparable fault diagnosis accuracies for both CWR and MFPT case studies in comparison to the ones obtained based on latent representations of the deep conventional neural networks VAE models. These results, however, were achieved at a higher computational cost and training time as the CNN-VAE models involve more complex architectures and greater number of learnable parameters.

Therefore, as a future guideline for the application of VAEs for dimensionality reduction in the context of ball bearings fault diagnosis, the results point to the direction where the proposed approach would be advantageous in scenarios involving significant reduction in dimensionality or when the amount of labeled data is limited. For cases where the labeled data is abundant or cheap to obtain, or yet higher dimensionalities are desired, PCA and VAE have comparable performance, thus making the former a better option as its training is computationally cheaper and faster. With respect to the choice of spectrograms or manually extracted features, results are not conclusive as to which method is better since the CWR case study showed better accuracy results with spectrograms while the opposite is true for the MFPT case study. However, both data types are superior to just feeding the raw vibration data to either the VAE models or PCA.

Finally, it is concluded from the development and results obtained in this thesis, that the objectives stated in Chapter 1 were succesfully completed. The model for the diagnosis of health states using the dimensionality reduction produced by Variational Auto Encoders was implemented and tested. A posterior evaluation of its perfomance through the comparison between it and a second and a third model (PCA based reduction and the model that do not perform a reduction at all) was completed, leading to interesting conclusions about the applicability of VAEs in the context of diagnosis, which were stated above and in Chapter 7.

As a final comment, even though it has been explored and demonstrated how deep VAE could be used in dimensionality reduction for fault diagnosis, the topics covered in this thesis only scratch the tip of the iceberg. Variational Auto-Encoders and Convolutional Neural Networks are just two of the possible deep learning techniques that have been proposed recently and VAEs have yet to be fully explored not only for fault diagnosis but also for prognosis.

# Chapter 9

# Bibliography

[1] H. Liu, L. Li, and J. Ma, "Rolling bearing fault diagnosis based on stft-deep learning and sound signals," *Shock and Vibration*, vol. 2016, 2016.

[2] R. M. Hasani, G. Wang, and R. Grosu, "An automated auto-encoder correlation-based health-monitoring and prognostic method for machine bearings," *arXiv preprint arXiv:1703.06272*, 2017.

[3] V. Meruane and A. Ortiz-Bernardin, "Structural damage assessment using linear approximation with maximum entropy and transmissibility data," *Mechanical Systems and Signal Processing*, vol. 54, pp. 210–223, 2015.

[4] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 3642–3649, IEEE, 2012.

[5] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial intelligence*, vol. 97, no. 1, pp. 245–271, 1997.

[6] D. Verstraete, M. Park, A. Ferrada, E. L. Droguett, V. Meruane, and M. Modarres, "Deep learning enabled fault diagnosis using time-frequency image analysis of rolling element bearings," 2006.

[7] B. Bagheri, H. Ahmadi, and R. Labbafi, "Application of data mining and feature extraction on intelligent fault diagnosis by artificial neural network and k-nearest neighbor," in *Electrical Machines (ICEM), 2010 XIX International Conference on*, pp. 1–7, IEEE, 2010.

[8] F. Zhou, Y. Gao, and C. Wen, "A novel multimode fault classification method based on deep learning," *Journal of Control Science and Engineering*, vol. 2017, 2017.

[9] Z. Pan, A. G. Rust, and H. Bolouri, "Image redundancy reduction for neural network classification using discrete cosine transforms," in *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, vol. 3, pp. 149–

154, IEEE, 2000.

[10] L. Shuang and L. Meng, “Bearing fault diagnosis based on pca and svm,” in *Mechatronics and Automation, 2007. ICMA 2007. International Conference on*, pp. 3503–3507, IEEE, 2007.

[11] W. Sun, J. Chen, and J. Li, “Decision tree and pca-based fault diagnosis of rotating machinery,” *Mechanical Systems and Signal Processing*, vol. 21, no. 3, pp. 1300–1317, 2007.

[12] I. T. Jolliffe and J. Cadima, “Principal component analysis: a review and recent developments,” *Phil. Trans. R. Soc. A*, vol. 374, no. 2065, p. 20150202, 2016.

[13] J. Shlens, “A tutorial on principal component analysis,” *arXiv preprint arXiv:1404.1100*, 2014.

[14] S. W. Choi, C. Lee, J.-M. Lee, J. H. Park, and I.-B. Lee, “Fault detection and identification of nonlinear processes based on kernel pca,” *Chemometrics and intelligent laboratory systems*, vol. 75, no. 1, pp. 55–67, 2005.

[15] F. Jia, E. B. Martin, and A. J. Morris, “Non-linear principal components analysis with application to process fault detection,” *International Journal of Systems Science*, vol. 31, no. 11, pp. 1473–1487, 2000.

[16] V. H. Nguyen and J.-C. Golinval, “Fault detection based on kernel principal component analysis,” *Engineering Structures*, vol. 32, no. 11, pp. 3683–3691, 2010.

[17] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[18] C. Lu, Z.-Y. Wang, W.-L. Qin, and J. Ma, “Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification,” *Signal Processing*, vol. 130, pp. 377–388, 2017.

[19] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[20] T. Salimans, D. Kingma, and M. Welling, “Markov chain monte carlo and variational inference: Bridging the gap,” in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1218–1226, 2015.

[21] D. J. Rezende, S. Mohamed, and D. Wierstra, “Stochastic backpropagation and approximate inference in deep generative models,” *arXiv preprint arXiv:1401.4082*, 2014.

[22] S. Semeniuta, A. Severyn, and E. Barth, “A hybrid convolutional variational autoencoder for text generation,” *arXiv preprint arXiv:1702.02390*, 2017.

[23] W.-N. Hsu, Y. Zhang, and J. Glass, “Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation,” *arXiv preprint*

*arXiv:1707.06265*, 2017.

[24] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[25] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[26] Z. Feng, M. Liang, and F. Chu, "Recent advances in time–frequency analysis methods for machinery fault diagnosis: A review with application examples," *Mechanical Systems and Signal Processing*, vol. 38, no. 1, pp. 165–205, 2013.

[27] C. M. Bishop, *Pattern recognition and machine learning.* springer, 2006.

[28] K. P. Murphy, *Machine learning: a probabilistic perspective.* MIT press, 2012.

[29] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[30] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems (MCSS)*, vol. 2, no. 4, pp. 303–314, 1989.

[31] M. A. Nielsen, "Neural networks and deep learning," 2015.

[32] E. Alpaydin, *Introduction to machine learning.* MIT press, 2014.

[33] "Cs231n convolutional neural networks for visual recognition." `http://cs231n.github.io/convolutional-networks/`. (Accessed on 12/08/2017).

[34] W. A. Smith and R. B. Randall, "Rolling element bearing diagnostics using the case western reserve university data: A benchmark study," *Mechanical Systems and Signal Processing*, vol. 64, pp. 100–131, 2015.

[35] E. Bechhoefer, "A quick introduction to bearing envelope analysis," 2016.

[36] "Mfpt bearing dataset | acoustics and vibration database." `http://data-acoustics.com/measurements/bearing-faults/bearing-2/`. (Accessed on 12/08/2017).

[37] H. Raveendran and D. Thomas, "Image fusion using lep filtering and bilinear interpolation," *arXiv preprint arXiv:1407.3986*, 2014.

[38] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[40] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, “Efficient object localization using convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 648–656, 2015.

[41] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708, 2014.

[42] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” *arXiv preprint arXiv:1312.6229*, 2013.

[43] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[44] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[46] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 52–59, 2011.

# Appendix A

# List of Manually Extracted Features

Table A.1: List of hand-engineered features used as a preprocessing methodology for the generation of datasets.

| **Original Signal** | **Component** |
|---|---|
| Maximum amplitude | 1 |
| Root Mean Square (RMS) | 2 |
| Peak to peak amplitude | 3 |
| Creast Factor | 4 |
| Arithmetic Mean | 5 |
| Variance | 6 |
| Skewness | 7 |
| Kurtosis | 8 |
| Centered moments (k=5 to 11) | 9 to 15 |
| Arithmetic mean of the Fourier amplitude, divided in 25 frequency bands | 16 to 40 |
| RMS of the first 5 IMFs (Empirical Mode Decomposition) | 41 to 45 |
| Shannon Entropy of the First 5 IMFs (Empirical Mode Decomposition) | 51 to 55 |
| RMS of the first 5 PFs (Local Mean Decomposition) | 56 to 60 |
| Percent energy of the first 5 PFs (Local Mean Decomposition) | 61 to 65 |
| Shannon entropy of the first 5 PFs (Local Mean Decomposition) | 66 to 70 |
| **Derivative of the Original Signal** | **Component** |
| Maximum amplitude | 71 |
| Root Mean Square (RMS) | 72 |
| Peak to peak amplitude | 73 |
| Creast Factor | 74 |
| Arithmetic Mean | 75 |
| Variance | 76 |
| Skewness | 77 |
| Kurtosis | 78 |
| Centered Moment (k=5 to 11) | 79 to 85 |
| **Integral of the Original Signal** | **Component** |
| Maximum amplitude | 86 |
| Root Mean Square (RMS) | 87 |
| Peak to peak amplitude | 88 |
| Creast Factor | 89 |
| Arithmetic Mean | 90 |
| Variance | 91 |
| Skewness | 92 |
| Kurtosis | 93 |
| Centered Moment (k=5 to 11) | 94 to 100 |

# Appendix B

# Tables of Complete Results

Table B.1: Average accuracies and standard deviations for CWR and MFPT case studies based on spectrograms.

| | | | CWR | | | MFPT | | |
|---|---|---|---|---|---|---|---|---|
| **Classifier** | $\varepsilon$% | $k$ | **PCA** | **VAE** | **Baseline** | **PCA** | **VAE** | **Baseline** |
| MLP1L | 0.01 | 2 | 43.14% ± 0.79% | 86.34% ± 1.26% | 48.00% ± 2.77% | 65.14% ± 2.04% | 86.11% ± 0.73% | 53.93% ± 3.14% |
| MLP1L | 0.01 | 4 | 71.49% ± 2.07% | 89.05% ± 1.43% | 48.00% ± 2.77% | 73.77% ± 3.16% | 88.35% ± 1.74% | 53.93% ± 3.14% |
| MLP1L | 0.01 | 8 | 79.71% ± 1.74% | 88.32% ± 1.80% | 48.00% ± 2.77% | 81.13% ± 2.32% | 81.64% ± 2.82% | 53.93% ± 3.14% |
| MLP1L | 0.01 | 16 | 75.20% ± 2.56% | 87.40% ± 2.91% | 48.00% ± 2.77% | 71.77% ± 2.38% | 76.80% ± 1.27% | 53.93% ± 3.14% |
| MLP1L | 0.01 | 64 | 41.12% ± 1.93% | 85.55% ± 0.94% | 48.00% ± 2.77% | 51.62% ± 2.99% | 70.88% ± 3.10% | 53.93% ± 3.14% |
| MLP1L | 0.01 | 128 | 32.49% ± 3.38% | 84.80% ± 1.84% | 48.00% ± 2.77% | 44.17% ± 3.98% | 69.65% ± 3.65% | 53.93% ± 3.14% |
| MLP1L | 0.05 | 2 | 46.60% ± 0.53% | 89.76% ± 0.40% | 77.17% ± 2.08% | 67.30% ± 0.72% | 86.86% ± 0.21% | 79.25% ± 1.16% |
| MLP1L | 0.05 | 4 | 79.00% ± 0.88% | 94.94% ± 0.31% | 77.17% ± 2.08% | 79.10% ± 1.42% | 91.74% ± 0.41% | 79.25% ± 1.16% |
| MLP1L | 0.05 | 8 | 89.60% ± 0.89% | 95.46% ± 0.41% | 77.17% ± 2.08% | 87.33% ± 1.37% | 88.40% ± 1.28% | 79.25% ± 1.16% |
| MLP1L | 0.05 | 16 | 92.07% ± 0.52% | 96.05% ± 0.41% | 77.17% ± 2.08% | 86.86% ± 0.66% | 86.28% ± 0.72% | 79.25% ± 1.16% |
| MLP1L | 0.05 | 64 | 74.17% ± 1.24% | 95.90% ± 0.78% | 77.17% ± 2.08% | 72.24% ± 1.38% | 85.38% ± 1.02% | 79.25% ± 1.16% |
| MLP1L | 0.05 | 128 | 65.29% ± 1.77% | 96.49% ± 0.42% | 77.17% ± 2.08% | 64.18% ± 2.15% | 86.07% ± 1.26% | 79.25% ± 1.16% |
| MLP1L | 0.25 | 2 | 48.18% ± 0.34% | 90.22% ± 0.24% | 93.52% ± 0.72% | 68.79% ± 0.38% | 87.12% ± 0.13% | 87.89% ± 0.47% |
| MLP1L | 0.25 | 4 | 82.34% ± 0.45% | 96.31% ± 0.29% | 93.52% ± 0.72% | 82.59% ± 0.47% | 92.78% ± 0.31% | 87.89% ± 0.47% |
| MLP1L | 0.25 | 8 | 92.56% ± 0.75% | 97.00% ± 0.26% | 93.52% ± 0.72% | 89.63% ± 0.44% | 91.21% ± 0.59% | 87.89% ± 0.47% |
| MLP1L | 0.25 | 16 | 96.45% ± 0.26% | 97.87% ± 0.33% | 93.52% ± 0.72% | 88.35% ± 0.35% | 88.93% ± 0.54% | 87.89% ± 0.47% |
| MLP1L | 0.25 | 64 | 92.28% ± 0.51% | 97.71% ± 0.26% | 93.52% ± 0.72% | 84.34% ± 1.03% | 89.50% ± 0.58% | 87.89% ± 0.47% |
| MLP1L | 0.25 | 128 | 89.18% ± 0.55% | 97.84% ± 0.19% | 93.52% ± 0.72% | 82.71% ± 0.92% | 90.36% ± 0.64% | 87.89% ± 0.47% |
| MLP1L | 1.00 | 2 | 49.21% ± 0.77% | 90.49% ± 0.12% | 98.35% ± 0.18% | 69.83% ± 0.36% | 87.10% ± 0.16% | 90.85% ± 0.44% |
| MLP1L | 1.00 | 4 | 83.78% ± 0.34% | 96.82% ± 0.19% | 98.35% ± 0.18% | 84.32% ± 0.27% | 93.23% ± 0.22% | 90.85% ± 0.44% |
| MLP1L | 1.00 | 8 | 94.32% ± 0.22% | 97.76% ± 0.20% | 98.35% ± 0.18% | 91.01% ± 0.50% | 92.55% ± 0.43% | 90.85% ± 0.44% |
| MLP1L | 1.00 | 16 | 97.57% ± 0.29% | 98.39% ± 0.19% | 98.35% ± 0.18% | 90.32% ± 0.35% | 91.29% ± 0.35% | 90.85% ± 0.44% |
| MLP1L | 1.00 | 64 | 97.45% ± 0.42% | 98.49% ± 0.18% | 98.35% ± 0.18% | 89.77% ± 1.10% | 91.00% ± 0.50% | 90.85% ± 0.44% |
| MLP1L | 1.00 | 128 | 96.57% ± 0.32% | 98.55% ± 0.17% | 98.35% ± 0.18% | 89.75% ± 0.66% | 91.47% ± 0.36% | 90.85% ± 0.44% |
| MLP1LDO | 0.01 | 2 | 42.71% ± 2.43% | 85.27% ± 1.46% | 31.61% ± 2.87% | 66.91% ± 1.31% | 85.93% ± 0.65% | 53.91% ± 3.27% |
| MLP1LDO | 0.01 | 4 | 75.90% ± 1.67% | 88.93% ± 1.77% | 31.61% ± 2.87% | 78.63% ± 2.24% | 89.09% ± 1.39% | 53.91% ± 3.27% |
| MLP1LDO | 0.01 | 8 | 84.77% ± 1.10% | 89.92% ± 1.37% | 31.61% ± 2.87% | 87.10% ± 1.47% | 84.90% ± 1.17% | 53.91% ± 3.27% |
| MLP1LDO | 0.01 | 16 | 88.76% ± 1.08% | 88.30% ± 2.16% | 31.61% ± 2.87% | 84.45% ± 2.45% | 82.33% ± 1.87% | 53.91% ± 3.27% |
| MLP1LDO | 0.01 | 64 | 77.66% ± 1.12% | 87.19% ± 1.57% | 31.61% ± 2.87% | 73.27% ± 2.25% | 80.43% ± 2.55% | 53.91% ± 3.27% |
| MLP1LDO | 0.01 | 128 | 70.41% ± 2.90% | 86.40% ± 3.05% | 31.61% ± 2.87% | 67.86% ± 1.71% | 78.75% ± 2.36% | 53.91% ± 3.27% |
| MLP1LDO | 0.05 | 2 | 47.36% ± 1.49% | 89.16% ± 0.65% | 37.57% ± 5.39% | 68.46% ± 0.88% | 86.61% ± 0.68% | 73.65% ± 7.63% |
| MLP1LDO | 0.05 | 4 | 81.61% ± 0.60% | 95.33% ± 0.48% | 37.57% ± 5.39% | 82.96% ± 0.89% | 92.15% ± 0.41% | 73.65% ± 7.63% |
| MLP1LDO | 0.05 | 8 | 92.20% ± 0.55% | 96.03% ± 0.33% | 37.57% ± 5.39% | 90.61% ± 0.76% | 91.02% ± 0.57% | 73.65% ± 7.63% |
| MLP1LDO | 0.05 | 16 | 96.36% ± 0.19% | 96.47% ± 0.38% | 37.57% ± 5.39% | 90.29% ± 0.57% | 89.20% ± 0.38% | 73.65% ± 7.63% |
| MLP1LDO | 0.05 | 64 | 94.34% ± 0.71% | 96.18% ± 0.38% | 37.57% ± 5.39% | 86.42% ± 0.67% | 89.20% ± 0.58% | 73.65% ± 7.63% |
| MLP1LDO | 0.05 | 128 | 92.98% ± 0.80% | 96.99% ± 0.43% | 37.57% ± 5.39% | 85.31% ± 0.72% | 90.14% ± 0.75% | 73.65% ± 7.63% |
| MLP1LDO | 0.25 | 2 | 49.69% ± 0.38% | 89.97% ± 0.22% | 73.68% ± 7.61% | 69.61% ± 0.52% | 87.20% ± 0.15% | 86.24% ± 1.65% |
| MLP1LDO | 0.25 | 4 | 83.74% ± 0.35% | 96.20% ± 0.23% | 73.68% ± 7.61% | 84.90% ± 0.37% | 92.81% ± 0.37% | 86.24% ± 1.65% |
| MLP1LDO | 0.25 | 8 | 94.26% ± 0.24% | 97.21% ± 0.22% | 73.68% ± 7.61% | 92.41% ± 0.32% | 92.71% ± 0.36% | 86.24% ± 1.65% |
| MLP1LDO | 0.25 | 16 | 97.74% ± 0.25% | 97.67% ± 0.19% | 73.68% ± 7.61% | 92.04% ± 0.42% | 91.67% ± 0.34% | 86.24% ± 1.65% |
| MLP1LDO | 0.25 | 64 | 98.12% ± 0.30% | 97.56% ± 0.33% | 73.68% ± 7.61% | 90.44% ± 0.49% | 92.10% ± 0.21% | 86.24% ± 1.65% |
| MLP1LDO | 0.25 | 128 | 98.10% ± 0.24% | 98.12% ± 0.16% | 73.68% ± 7.61% | 91.09% ± 0.26% | 92.82% ± 0.31% | 86.24% ± 1.65% |
| MLP1LDO | 1.00 | 2 | 49.99% ± 0.11% | 90.44% ± 0.20% | 53.10% ± 26.43% | 70.61% ± 0.15% | 87.18% ± 0.15% | 90.92% ± 0.27% |
| MLP1LDO | 1.00 | 4 | 84.39% ± 0.30% | 96.57% ± 0.26% | 53.10% ± 26.43% | 84.93% ± 0.14% | 93.14% ± 0.30% | 90.92% ± 0.27% |
| MLP1LDO | 1.00 | 8 | 94.96% ± 0.27% | 97.56% ± 0.28% | 53.10% ± 26.43% | 93.19% ± 0.24% | 93.21% ± 0.34% | 90.92% ± 0.27% |
| MLP1LDO | 1.00 | 16 | 98.24% ± 0.23% | 98.22% ± 0.22% | 53.10% ± 26.43% | 93.35% ± 0.38% | 93.03% ± 0.34% | 90.92% ± 0.27% |
| MLP1LDO | 1.00 | 64 | 98.85% ± 0.13% | 98.33% ± 0.18% | 53.10% ± 26.43% | 93.24% ± 0.55% | 93.33% ± 0.32% | 90.92% ± 0.27% |
| MLP1LDO | 1.00 | 128 | 99.08% ± 0.14% | 98.44% ± 0.17% | 53.10% ± 26.43% | 93.44% ± 0.68% | 93.91% ± 0.30% | 90.92% ± 0.27% |
| MLP3L | 0.01 | 2 | 40.85% ± 1.20% | 79.41% ± 4.75% | 32.60% ± 1.72% | 65.17% ± 2.04% | 84.27% ± 2.25% | 54.41% ± 1.41% |
| MLP3L | 0.01 | 4 | 73.78% ± 1.44% | 88.78% ± 1.54% | 32.60% ± 1.72% | 77.46% ± 1.02% | 88.63% ± 1.15% | 54.41% ± 1.41% |
| MLP3L | 0.01 | 8 | 85.45% ± 1.91% | 89.48% ± 1.55% | 32.60% ± 1.72% | 86.44% ± 2.07% | 86.24% ± 1.61% | 54.41% ± 1.41% |
| MLP3L | 0.01 | 16 | 91.22% ± 1.08% | 90.20% ± 1.11% | 32.60% ± 1.72% | 84.29% ± 1.28% | 84.35% ± 1.53% | 54.41% ± 1.41% |
| MLP3L | 0.01 | 64 | 86.31% ± 1.41% | 90.10% ± 1.63% | 32.60% ± 1.72% | 79.44% ± 1.64% | 83.33% ± 1.51% | 54.41% ± 1.41% |
| MLP3L | 0.01 | 128 | 83.64% ± 1.12% | 90.22% ± 1.41% | 32.60% ± 1.72% | 77.38% ± 2.03% | 81.20% ± 2.13% | 54.41% ± 1.41% |
| MLP3L | 0.05 | 2 | 41.52% ± 1.66% | 82.04% ± 4.22% | 33.25% ± 1.91% | 66.22% ± 2.35% | 85.60% ± 2.51% | 57.14% ± 2.59% |
| MLP3L | 0.05 | 4 | 78.22% ± 1.03% | 93.91% ± 1.27% | 33.25% ± 1.91% | 79.77% ± 1.32% | 90.54% ± 1.19% | 57.14% ± 2.59% |
| MLP3L | 0.05 | 8 | 91.61% ± 0.58% | 95.91% ± 0.52% | 33.25% ± 1.91% | 89.64% ± 0.89% | 90.21% ± 0.76% | 57.14% ± 2.59% |
| MLP3L | 0.05 | 16 | 96.74% ± 0.24% | 96.31% ± 0.50% | 33.25% ± 1.91% | 89.22% ± 0.76% | 89.54% ± 1.19% | 57.14% ± 2.59% |
| MLP3L | 0.05 | 64 | 96.62% ± 0.30% | 95.82% ± 0.63% | 33.25% ± 1.91% | 86.87% ± 0.89% | 88.54% ± 1.05% | 57.14% ± 2.59% |
| MLP3L | 0.05 | 128 | 96.19% ± 0.49% | 96.52% ± 0.47% | 33.25% ± 1.91% | 87.16% ± 0.63% | 89.77% ± 0.90% | 57.14% ± 2.59% |
| MLP3L | 0.25 | 2 | 41.38% ± 1.08% | 82.26% ± 2.18% | 23.48% ± 0.73% | 67.65% ± 1.20% | 86.84% ± 0.25% | 58.65% ± 1.29% |
| MLP3L | 0.25 | 4 | 79.53% ± 1.02% | 95.42% ± 0.48% | 23.48% ± 0.73% | 81.59% ± 1.34% | 92.02% ± 0.80% | 58.65% ± 1.29% |
| MLP3L | 0.25 | 8 | 93.41% ± 0.56% | 96.88% ± 0.42% | 23.48% ± 0.73% | 90.83% ± 0.79% | 91.56% ± 0.59% | 58.65% ± 1.29% |
| MLP3L | 0.25 | 16 | 97.64% ± 0.23% | 97.44% ± 0.28% | 23.48% ± 0.73% | 90.65% ± 1.00% | 90.94% ± 0.67% | 58.65% ± 1.29% |
| MLP3L | 0.25 | 64 | 98.16% ± 0.34% | 96.97% ± 0.22% | 23.48% ± 0.73% | 90.00% ± 0.94% | 91.07% ± 0.79% | 58.65% ± 1.29% |
| MLP3L | 0.25 | 128 | 98.28% ± 0.24% | 97.75% ± 0.24% | 23.48% ± 0.73% | 90.93% ± 0.25% | 91.56% ± 0.98% | 58.65% ± 1.29% |
| MLP3L | 1.00 | 2 | 41.36% ± 1.75% | 83.93% ± 1.48% | 24.13% ± 0.08% | 67.80% ± 1.12% | 86.77% ± 0.35% | 50.12% ± 0.19% |
| MLP3L | 1.00 | 4 | 78.92% ± 1.21% | 95.63% ± 0.67% | 24.13% ± 0.08% | 82.84% ± 0.67% | 92.92% ± 0.35% | 50.12% ± 0.19% |
| MLP3L | 1.00 | 8 | 93.64% ± 0.29% | 97.30% ± 0.42% | 24.13% ± 0.08% | 91.99% ± 0.28% | 92.65% ± 0.55% | 50.12% ± 0.19% |
| MLP3L | 1.00 | 16 | 98.11% ± 0.18% | 97.94% ± 0.17% | 24.13% ± 0.08% | 92.15% ± 0.22% | 91.88% ± 0.42% | 50.12% ± 0.19% |
| MLP3L | 1.00 | 64 | 98.91% ± 0.21% | 97.68% ± 0.40% | 24.13% ± 0.08% | 90.89% ± 0.56% | 92.23% ± 0.26% | 50.12% ± 0.19% |
| MLP3L | 1.00 | 128 | 98.97% ± 0.20% | 98.19% ± 0.17% | 24.13% ± 0.08% | 91.65% ± 0.37% | 92.93% ± 0.53% | 50.12% ± 0.19% |

Table B.2: Average accuracies and standard deviations for CWR and MFPT case studies based on manually extracted features.

| | | | **CWR - Features** | | | **MFPT - Features** | | |
|---|---|---|---|---|---|---|---|---|
| **Classifier** | $\varepsilon$% | $k$ | **PCA** | **VAE** | **Baseline** | **PCA** | **VAE** | **Baseline** |
| MLP1L | 0.01 | 2 | 18.46% ± 1.91% | 40.13% ± 2.12% | 12.22% ± 1.47% | 63.13% ± 2.05% | 96.64% ± 0.80% | 47.07% ± 3.28% |
| MLP1L | 0.01 | 4 | 17.10% ± 1.68% | 54.33% ± 1.42% | 12.22% ± 1.47% | 66.20% ± 5.34% | 96.79% ± 1.19% | 47.07% ± 3.28% |
| MLP1L | 0.01 | 8 | 19.85% ± 1.69% | 40.98% ± 2.34% | 12.22% ± 1.47% | 67.32% ± 3.23% | 92.05% ± 1.66% | 47.07% ± 3.28% |
| MLP1L | 0.01 | 16 | 16.74% ± 2.02% | 28.09% ± 1.18% | 12.22% ± 1.47% | 57.58% ± 5.13% | 84.33% ± 2.15% | 47.07% ± 3.28% |
| MLP1L | 0.01 | 64 | 12.00% ± 0.74% | 19.82% ± 2.04% | 12.22% ± 1.47% | 44.64% ± 3.02% | 74.46% ± 2.52% | 47.07% ± 3.28% |
| MLP1L | 0.05 | 2 | 24.20% ± 0.69% | 42.32% ± 1.27% | 31.03% ± 2.26% | 67.42% ± 0.95% | 95.57% ± 0.32% | 75.81% ± 1.98% |
| MLP1L | 0.05 | 4 | 22.74% ± 1.10% | 64.59% ± 0.85% | 31.03% ± 2.26% | 77.52% ± 1.48% | 98.29% ± 0.25% | 75.81% ± 1.98% |
| MLP1L | 0.05 | 8 | 26.44% ± 0.67% | 60.82% ± 2.22% | 31.03% ± 2.26% | 82.10% ± 1.09% | 97.23% ± 0.83% | 75.81% ± 1.98% |
| MLP1L | 0.05 | 16 | 37.27% ± 1.64% | 55.34% ± 1.31% | 31.03% ± 2.26% | 84.75% ± 1.31% | 95.44% ± 0.74% | 75.81% ± 1.98% |
| MLP1L | 0.05 | 64 | 27.14% ± 1.31% | 49.18% ± 1.46% | 31.03% ± 2.26% | 73.51% ± 2.23% | 94.43% ± 1.08% | 75.81% ± 1.98% |
| MLP1L | 0.25 | 2 | 27.37% ± 0.60% | 44.81% ± 0.29% | 65.20% ± 0.66% | 69.84% ± 0.89% | 95.74% ± 0.23% | 93.18% ± 0.57% |
| MLP1L | 0.25 | 4 | 29.19% ± 0.96% | 69.32% ± 0.66% | 65.20% ± 0.66% | 82.87% ± 0.62% | 98.72% ± 0.17% | 93.18% ± 0.57% |
| MLP1L | 0.25 | 8 | 33.29% ± 1.74% | 70.75% ± 1.30% | 65.20% ± 0.66% | 86.85% ± 0.89% | 99.02% ± 0.21% | 93.18% ± 0.57% |
| MLP1L | 0.25 | 16 | 51.37% ± 1.31% | 73.49% ± 0.90% | 65.20% ± 0.66% | 92.68% ± 0.49% | 98.49% ± 0.21% | 93.18% ± 0.57% |
| MLP1L | 0.25 | 64 | 64.40% ± 0.93% | 81.61% ± 0.89% | 65.20% ± 0.66% | 92.58% ± 0.86% | 98.01% ± 0.44% | 93.18% ± 0.57% |
| MLP1L | 1.00 | 2 | 29.17% ± 0.45% | 48.85% ± 0.31% | 88.41% ± 0.54% | 71.01% ± 0.76% | 97.99% ± 0.15% | 97.89% ± 0.14% |
| MLP1L | 1.00 | 4 | 34.05% ± 0.98% | 71.79% ± 0.26% | 88.41% ± 0.54% | 85.05% ± 0.41% | 99.31% ± 0.13% | 97.89% ± 0.14% |
| MLP1L | 1.00 | 8 | 43.56% ± 0.73% | 77.23% ± 0.83% | 88.41% ± 0.54% | 91.57% ± 0.48% | 99.53% ± 0.15% | 97.89% ± 0.14% |
| MLP1L | 1.00 | 16 | 68.01% ± 0.96% | 83.15% ± 0.47% | 88.41% ± 0.54% | 95.88% ± 0.50% | 99.14% ± 0.25% | 97.89% ± 0.14% |
| MLP1L | 1.00 | 64 | 88.38% ± 0.67% | 92.78% ± 0.33% | 88.41% ± 0.54% | 98.01% ± 0.32% | 99.23% ± 0.20% | 97.89% ± 0.14% |
| MLP1LDO | 0.01 | 2 | 20.91% ± 1.02% | 35.53% ± 3.79% | 22.49% ± 1.83% | 57.29% ± 4.01% | 96.50% ± 1.50% | 62.95% ± 3.37% |
| MLP1LDO | 0.01 | 4 | 21.41% ± 2.15% | 55.28% ± 2.65% | 22.49% ± 1.83% | 69.30% ± 7.18% | 95.54% ± 1.67% | 62.95% ± 3.37% |
| MLP1LDO | 0.01 | 8 | 21.82% ± 2.61% | 46.34% ± 1.82% | 22.49% ± 1.83% | 70.92% ± 3.83% | 93.90% ± 1.31% | 62.95% ± 3.37% |
| MLP1LDO | 0.01 | 16 | 24.74% ± 2.32% | 35.12% ± 2.64% | 22.49% ± 1.83% | 70.61% ± 3.65% | 91.96% ± 1.61% | 62.95% ± 3.37% |
| MLP1LDO | 0.01 | 64 | 22.64% ± 1.73% | 34.56% ± 2.90% | 22.49% ± 1.83% | 61.04% ± 3.59% | 87.83% ± 2.42% | 62.95% ± 3.37% |
| MLP1LDO | 0.05 | 2 | 25.43% ± 2.33% | 39.98% ± 2.88% | 39.11% ± 0.79% | 66.91% ± 1.70% | 95.02% ± 0.58% | 89.95% ± 0.79% |
| MLP1LDO | 0.05 | 4 | 28.43% ± 2.26% | 66.66% ± 0.66% | 39.11% ± 0.79% | 81.39% ± 0.91% | 98.22% ± 0.26% | 89.95% ± 0.79% |
| MLP1LDO | 0.05 | 8 | 33.99% ± 1.45% | 67.13% ± 1.56% | 39.11% ± 0.79% | 87.60% ± 1.68% | 97.59% ± 0.67% | 89.95% ± 0.79% |
| MLP1LDO | 0.05 | 16 | 43.45% ± 1.61% | 62.99% ± 1.40% | 39.11% ± 0.79% | 89.88% ± 0.98% | 97.73% ± 0.30% | 89.95% ± 0.79% |
| MLP1LDO | 0.05 | 64 | 40.46% ± 1.83% | 63.77% ± 1.23% | 39.11% ± 0.79% | 87.69% ± 1.24% | 98.36% ± 0.53% | 89.95% ± 0.79% |
| MLP1LDO | 0.25 | 2 | 28.65% ± 0.77% | 42.70% ± 0.36% | 79.85% ± 1.25% | 68.66% ± 0.87% | 95.28% ± 0.36% | 98.05% ± 0.17% |
| MLP1LDO | 0.25 | 4 | 34.23% ± 0.63% | 69.54% ± 0.49% | 79.85% ± 1.25% | 83.37% ± 0.29% | 98.61% ± 0.18% | 98.05% ± 0.17% |
| MLP1LDO | 0.25 | 8 | 41.56% ± 0.42% | 74.64% ± 0.47% | 79.85% ± 1.25% | 91.36% ± 0.51% | 98.71% ± 0.23% | 98.05% ± 0.17% |
| MLP1LDO | 0.25 | 16 | 62.79% ± 0.74% | 81.57% ± 0.45% | 79.85% ± 1.25% | 95.14% ± 0.47% | 99.31% ± 0.22% | 98.05% ± 0.17% |
| MLP1LDO | 0.25 | 64 | 81.66% ± 0.87% | 89.49% ± 0.94% | 79.85% ± 1.25% | 98.07% ± 0.39% | 99.33% ± 0.20% | 98.05% ± 0.17% |
| MLP1LDO | 1.00 | 2 | 29.37% ± 0.28% | 47.13% ± 0.72% | 95.59% ± 0.25% | 68.73% ± 0.48% | 97.94% ± 0.17% | 99.62% ± 0.19% |
| MLP1LDO | 1.00 | 4 | 35.76% ± 0.18% | 71.62% ± 0.15% | 95.59% ± 0.25% | 84.08% ± 0.34% | 99.24% ± 0.16% | 99.62% ± 0.19% |
| MLP1LDO | 1.00 | 8 | 44.82% ± 0.59% | 78.11% ± 0.42% | 95.59% ± 0.25% | 92.49% ± 0.50% | 99.54% ± 0.14% | 99.62% ± 0.19% |
| MLP1LDO | 1.00 | 16 | 70.07% ± 0.83% | 85.58% ± 0.49% | 95.59% ± 0.25% | 96.87% ± 0.32% | 99.57% ± 0.17% | 99.62% ± 0.19% |
| MLP1LDO | 1.00 | 64 | 95.64% ± 0.53% | 95.25% ± 0.25% | 95.59% ± 0.25% | 99.62% ± 0.14% | 99.55% ± 0.13% | 99.62% ± 0.19% |
| MLP3L | 0.01 | 2 | 20.30% ± 1.68% | 34.35% ± 1.94% | 26.09% ± 2.12% | 54.31% ± 3.45% | 95.82% ± 2.36% | 69.39% ± 2.60% |
| MLP3L | 0.01 | 4 | 20.92% ± 1.63% | 47.48% ± 4.13% | 26.09% ± 2.12% | 61.36% ± 8.29% | 96.11% ± 1.40% | 69.39% ± 2.60% |
| MLP3L | 0.01 | 8 | 21.02% ± 1.92% | 47.93% ± 1.77% | 26.09% ± 2.12% | 71.92% ± 2.67% | 95.15% ± 0.78% | 69.39% ± 2.60% |
| MLP3L | 0.01 | 16 | 25.37% ± 2.09% | 40.04% ± 2.56% | 26.09% ± 2.12% | 75.69% ± 2.67% | 92.36% ± 2.34% | 69.39% ± 2.60% |
| MLP3L | 0.01 | 64 | 24.50% ± 2.28% | 39.77% ± 1.60% | 26.09% ± 2.12% | 68.84% ± 2.24% | 91.91% ± 2.83% | 69.39% ± 2.60% |
| MLP3L | 0.05 | 2 | 22.04% ± 2.52% | 33.76% ± 2.61% | 23.52% ± 1.62% | 59.55% ± 3.30% | 94.81% ± 0.58% | 93.04% ± 0.89% |
| MLP3L | 0.05 | 4 | 22.01% ± 1.81% | 48.97% ± 2.59% | 23.52% ± 1.62% | 70.90% ± 1.65% | 97.79% ± 0.58% | 93.04% ± 0.89% |
| MLP3L | 0.05 | 8 | 22.79% ± 1.04% | 45.55% ± 1.65% | 23.52% ± 1.62% | 82.29% ± 0.93% | 97.59% ± 0.50% | 93.04% ± 0.89% |
| MLP3L | 0.05 | 16 | 26.31% ± 1.75% | 38.06% ± 2.10% | 23.52% ± 1.62% | 90.75% ± 0.57% | 98.31% ± 0.27% | 93.04% ± 0.89% |
| MLP3L | 0.05 | 64 | 22.91% ± 1.16% | 36.58% ± 2.14% | 23.52% ± 1.62% | 91.78% ± 0.86% | 98.66% ± 0.45% | 93.04% ± 0.89% |
| MLP3L | 0.25 | 2 | 22.69% ± 1.86% | 33.36% ± 1.88% | 23.30% ± 1.05% | 61.46% ± 1.92% | 95.52% ± 0.40% | 97.22% ± 0.32% |
| MLP3L | 0.25 | 4 | 24.37% ± 1.01% | 48.07% ± 3.33% | 23.30% ± 1.05% | 71.50% ± 1.97% | 98.58% ± 0.22% | 97.22% ± 0.32% |
| MLP3L | 0.25 | 8 | 24.19% ± 1.67% | 44.30% ± 2.09% | 23.30% ± 1.05% | 82.73% ± 1.40% | 98.55% ± 0.39% | 97.22% ± 0.32% |
| MLP3L | 0.25 | 16 | 25.94% ± 0.58% | 37.60% ± 1.35% | 23.30% ± 1.05% | 91.54% ± 1.09% | 99.34% ± 0.11% | 97.22% ± 0.32% |
| MLP3L | 0.25 | 64 | 23.61% ± 1.37% | 34.53% ± 2.46% | 23.30% ± 1.05% | 94.29% ± 1.23% | 99.58% ± 0.15% | 97.22% ± 0.32% |
| MLP3L | 1.00 | 2 | 23.27% ± 1.65% | 33.97% ± 1.90% | 23.09% ± 1.26% | 59.69% ± 2.89% | 97.93% ± 0.15% | 95.79% ± 1.55% |
| MLP3L | 1.00 | 4 | 24.27% ± 0.63% | 48.93% ± 2.03% | 23.09% ± 1.26% | 70.83% ± 1.81% | 99.26% ± 0.13% | 95.79% ± 1.55% |
| MLP3L | 1.00 | 8 | 24.38% ± 1.06% | 45.22% ± 1.59% | 23.09% ± 1.26% | 82.79% ± 1.15% | 99.31% ± 0.11% | 95.79% ± 1.55% |
| MLP3L | 1.00 | 16 | 25.33% ± 1.48% | 36.69% ± 1.21% | 23.09% ± 1.26% | 91.20% ± 1.24% | 99.45% ± 0.18% | 95.79% ± 1.55% |
| MLP3L | 1.00 | 64 | 22.66% ± 2.23% | 35.59% ± 1.53% | 23.09% ± 1.26% | 93.89% ± 1.62% | 99.46% ± 0.11% | 95.79% ± 1.55% |

Table B.3: Average accuracies and standard deviations for CWR and MFPT case studies based on chunks of the raw vibration signal.

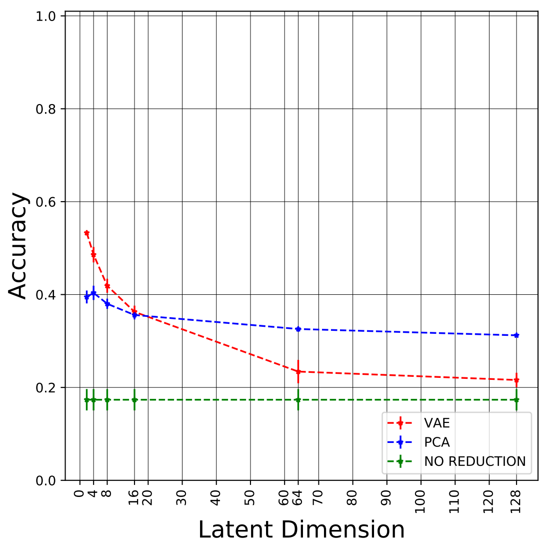
| | | | **CWR - Raw Vibration Signal** | | | **MFPT - Raw Vibration Signal** | | |
|---|---|---|---|---|---|---|---|---|
| **Classifier** | $\varepsilon$% | $k$ | **PCA** | **VAE** | **Baseline** | **PCA** | **VAE** | **Baseline** |
| MLP1L | 0.01 | 2 | 39.48% ± 1.39% | 53.28% ± 0.54% | 17.36% ± 2.30% | 45.18% ± 1.56% | 60.38% ± 1.38% | 35.63% ± 4.26% |
| MLP1L | 0.01 | 4 | 40.37% ± 1.54% | 48.59% ± 1.66% | 17.36% ± 2.30% | 42.30% ± 2.79% | 54.37% ± 2.09% | 35.63% ± 4.26% |
| MLP1L | 0.01 | 8 | 38.02% ± 1.11% | 41.90% ± 1.52% | 17.36% ± 2.30% | 42.76% ± 1.99% | 56.66% ± 2.81% | 35.63% ± 4.26% |
| MLP1L | 0.01 | 16 | 35.60% ± 0.95% | 36.28% ± 1.34% | 17.36% ± 2.30% | 39.27% ± 2.25% | 48.47% ± 1.55% | 35.63% ± 4.26% |
| MLP1L | 0.01 | 64 | 32.58% ± 0.59% | 23.42% ± 2.51% | 17.36% ± 2.30% | 37.92% ± 1.81% | 39.88% ± 1.58% | 35.63% ± 4.26% |
| MLP1L | 0.01 | 128 | 31.22% ± 0.42% | 21.61% ± 1.56% | 17.36% ± 2.30% | 38.89% ± 4.19% | 37.71% ± 3.02% | 35.63% ± 4.26% |
| MLP1L | 0.05 | 2 | 44.88% ± 1.25% | 52.58% ± 0.64% | 28.56% ± 1.01% | 48.93% ± 0.97% | 50.62% ± 0.32% | 38.74% ± 1.58% |
| MLP1L | 0.05 | 4 | 48.40% ± 1.25% | 52.98% ± 0.71% | 28.56% ± 1.01% | 46.30% ± 1.26% | 57.03% ± 0.93% | 38.74% ± 1.58% |
| MLP1L | 0.05 | 8 | 49.52% ± 1.01% | 52.17% ± 1.52% | 28.56% ± 1.01% | 46.00% ± 1.33% | 62.96% ± 1.51% | 38.74% ± 1.58% |
| MLP1L | 0.05 | 16 | 46.25% ± 1.03% | 47.83% ± 1.16% | 28.56% ± 1.01% | 46.83% ± 1.21% | 62.92% ± 0.93% | 38.74% ± 1.58% |
| MLP1L | 0.05 | 64 | 38.64% ± 1.11% | 34.02% ± 0.87% | 28.56% ± 1.01% | 45.15% ± 1.65% | 49.25% ± 1.72% | 38.74% ± 1.58% |
| MLP1L | 0.05 | 128 | 35.22% ± 0.93% | 32.72% ± 1.00% | 28.56% ± 1.01% | 44.17% ± 0.75% | 42.40% ± 1.42% | 38.74% ± 1.58% |
| MLP1L | 0.25 | 2 | 47.46% ± 0.71% | 53.32% ± 0.35% | 38.64% ± 0.44% | 51.28% ± 0.80% | 51.18% ± 0.55% | 41.42% ± 0.27% |
| MLP1L | 0.25 | 4 | 53.00% ± 0.65% | 54.29% ± 0.47% | 38.64% ± 0.44% | 49.75% ± 0.88% | 57.48% ± 1.53% | 41.42% ± 0.27% |
| MLP1L | 0.25 | 8 | 57.72% ± 0.77% | 56.23% ± 0.95% | 38.64% ± 0.44% | 51.77% ± 1.21% | 64.40% ± 1.44% | 41.42% ± 0.27% |
| MLP1L | 0.25 | 16 | 61.65% ± 1.36% | 53.94% ± 0.92% | 38.64% ± 0.44% | 54.76% ± 1.10% | 67.12% ± 0.67% | 41.42% ± 0.27% |
| MLP1L | 0.25 | 64 | 56.79% ± 0.99% | 48.56% ± 0.49% | 38.64% ± 0.44% | 65.65% ± 1.23% | 63.02% ± 1.45% | 41.42% ± 0.27% |
| MLP1L | 0.25 | 128 | 52.25% ± 0.71% | 45.92% ± 0.95% | 38.64% ± 0.44% | 59.03% ± 1.06% | 54.87% ± 0.79% | 41.42% ± 0.27% |
| MLP1L | 1.00 | 2 | 49.17% ± 0.36% | 55.36% ± 0.44% | 59.32% ± 0.55% | 52.49% ± 0.62% | 62.07% ± 0.15% | 53.74% ± 0.82% |
| MLP1L | 1.00 | 4 | 54.98% ± 0.65% | 56.16% ± 0.43% | 59.32% ± 0.55% | 52.47% ± 1.17% | 57.17% ± 0.91% | 53.74% ± 0.82% |
| MLP1L | 1.00 | 8 | 62.39% ± 0.85% | 56.99% ± 0.78% | 59.32% ± 0.55% | 58.71% ± 1.32% | 65.95% ± 0.84% | 53.74% ± 0.82% |
| MLP1L | 1.00 | 16 | 70.13% ± 0.70% | 57.85% ± 0.85% | 59.32% ± 0.55% | 64.03% ± 1.03% | 68.67% ± 0.69% | 53.74% ± 0.82% |
| MLP1L | 1.00 | 64 | 75.92% ± 0.58% | 60.23% ± 0.72% | 59.32% ± 0.55% | 80.41% ± 0.79% | 72.57% ± 0.67% | 53.74% ± 0.82% |
| MLP1L | 1.00 | 128 | 75.62% ± 1.23% | 60.22% ± 0.85% | 59.32% ± 0.55% | 78.25% ± 0.91% | 69.56% ± 0.53% | 53.74% ± 0.82% |
| MLP1LDO | 0.01 | 2 | 37.92% ± 2.77% | 50.93% ± 3.75% | 27.82% ± 1.27% | 47.52% ± 1.87% | 62.04% ± 1.06% | 42.46% ± 1.74% |
| MLP1LDO | 0.01 | 4 | 38.16% ± 2.52% | 51.72% ± 1.02% | 27.82% ± 1.27% | 43.97% ± 2.12% | 60.16% ± 2.03% | 42.46% ± 1.74% |
| MLP1LDO | 0.01 | 8 | 37.13% ± 1.84% | 45.11% ± 1.27% | 27.82% ± 1.27% | 44.74% ± 3.77% | 64.94% ± 0.91% | 42.46% ± 1.74% |
| MLP1LDO | 0.01 | 16 | 34.94% ± 9.18% | 39.98% ± 1.06% | 27.82% ± 1.27% | 42.97% ± 2.62% | 55.22% ± 2.25% | 42.46% ± 1.74% |
| MLP1LDO | 0.01 | 64 | 33.36% ± 8.46% | 33.68% ± 0.87% | 27.82% ± 1.27% | 43.13% ± 2.04% | 48.78% ± 1.41% | 42.46% ± 1.74% |
| MLP1LDO | 0.01 | 128 | 33.79% ± 1.46% | 33.83% ± 0.95% | 27.82% ± 1.27% | 44.88% ± 1.72% | 45.75% ± 1.33% | 42.46% ± 1.74% |
| MLP1LDO | 0.05 | 2 | 43.80% ± 2.58% | 52.58% ± 0.74% | 34.56% ± 5.09% | 50.05% ± 1.81% | 52.43% ± 0.57% | 46.16% ± 1.89% |
| MLP1LDO | 0.05 | 4 | 49.05% ± 2.16% | 54.96% ± 0.70% | 34.56% ± 5.09% | 49.94% ± 1.62% | 63.64% ± 0.96% | 46.16% ± 1.89% |
| MLP1LDO | 0.05 | 8 | 52.91% ± 1.98% | 56.93% ± 1.45% | 34.56% ± 5.09% | 48.77% ± 3.60% | 70.10% ± 0.55% | 46.16% ± 1.89% |
| MLP1LDO | 0.05 | 16 | 54.59% ± 1.79% | 52.17% ± 1.49% | 34.56% ± 5.09% | 53.11% ± 2.58% | 67.80% ± 1.06% | 46.16% ± 1.89% |
| MLP1LDO | 0.05 | 64 | 46.59% ± 0.89% | 42.10% ± 0.62% | 34.56% ± 5.09% | 56.23% ± 0.74% | 59.22% ± 1.31% | 46.16% ± 1.89% |
| MLP1LDO | 0.05 | 128 | 43.60% ± 1.48% | 41.78% ± 1.53% | 34.56% ± 5.09% | 54.77% ± 2.86% | 51.85% ± 1.54% | 46.16% ± 1.89% |
| MLP1LDO | 0.25 | 2 | 49.11% ± 0.42% | 53.52% ± 0.39% | 50.75% ± 1.36% | 52.39% ± 1.26% | 52.49% ± 0.48% | 52.15% ± 0.77% |
| MLP1LDO | 0.25 | 4 | 55.20% ± 0.74% | 56.36% ± 0.67% | 50.75% ± 1.36% | 53.25% ± 1.58% | 63.39% ± 1.40% | 52.15% ± 0.77% |
| MLP1LDO | 0.25 | 8 | 61.58% ± 0.69% | 59.44% ± 0.61% | 50.75% ± 1.36% | 61.16% ± 2.05% | 70.95% ± 0.72% | 52.15% ± 0.77% |
| MLP1LDO | 0.25 | 16 | 67.66% ± 0.52% | 58.80% ± 0.79% | 50.75% ± 1.36% | 64.63% ± 2.05% | 70.53% ± 0.51% | 52.15% ± 0.77% |
| MLP1LDO | 0.25 | 64 | 68.97% ± 0.61% | 57.45% ± 0.92% | 50.75% ± 1.36% | 79.68% ± 0.88% | 67.05% ± 1.25% | 52.15% ± 0.77% |
| MLP1LDO | 0.25 | 128 | 66.10% ± 1.27% | 54.42% ± 0.93% | 50.75% ± 1.36% | 76.74% ± 3.31% | 61.93% ± 1.92% | 52.15% ± 0.77% |
| MLP1LDO | 1.00 | 2 | 50.04% ± 0.23% | 55.51% ± 0.39% | 81.91% ± 0.53% | 52.92% ± 0.14% | 62.20% ± 0.41% | 68.36% ± 0.36% |
| MLP1LDO | 1.00 | 4 | 56.65% ± 0.33% | 57.57% ± 0.53% | 81.91% ± 0.53% | 54.29% ± 0.37% | 63.63% ± 1.20% | 68.36% ± 0.36% |
| MLP1LDO | 1.00 | 8 | 64.35% ± 0.41% | 59.61% ± 0.44% | 81.91% ± 0.53% | 63.42% ± 0.42% | 72.81% ± 0.96% | 68.36% ± 0.36% |
| MLP1LDO | 1.00 | 16 | 72.59% ± 0.67% | 61.51% ± 0.42% | 81.91% ± 0.53% | 68.64% ± 0.29% | 72.05% ± 0.53% | 68.36% ± 0.36% |
| MLP1LDO | 1.00 | 64 | 81.94% ± 1.00% | 64.09% ± 1.01% | 81.91% ± 0.53% | 85.18% ± 0.39% | 73.47% ± 0.61% | 68.36% ± 0.36% |
| MLP1LDO | 1.00 | 128 | 84.49% ± 0.56% | 63.36% ± 0.53% | 81.91% ± 0.53% | 88.92% ± 0.33% | 74.62% ± 0.49% | 68.36% ± 0.36% |
| MLP3L | 0.01 | 2 | 34.03% ± 1.23% | 51.37% ± 1.75% | 30.93% ± 4.34% | 47.36% ± 3.27% | 58.51% ± 2.38% | 46.42% ± 1.26% |
| MLP3L | 0.01 | 4 | 33.24% ± 0.54% | 50.45% ± 1.03% | 30.93% ± 4.34% | 45.88% ± 2.07% | 55.54% ± 3.23% | 46.42% ± 1.26% |
| MLP3L | 0.01 | 8 | 33.16% ± 0.80% | 46.92% ± 1.34% | 30.93% ± 4.34% | 43.98% ± 1.81% | 63.06% ± 2.17% | 46.42% ± 1.26% |
| MLP3L | 0.01 | 16 | 32.99% ± 0.75% | 42.35% ± 1.53% | 30.93% ± 4.34% | 44.72% ± 2.69% | 55.43% ± 1.38% | 46.42% ± 1.26% |
| MLP3L | 0.01 | 64 | 33.52% ± 1.56% | 35.82% ± 0.99% | 30.93% ± 4.34% | 46.17% ± 1.27% | 49.52% ± 1.87% | 46.42% ± 1.26% |
| MLP3L | 0.01 | 128 | 33.99% ± 1.25% | 34.66% ± 0.92% | 30.93% ± 4.34% | 47.38% ± 1.34% | 47.28% ± 1.50% | 46.42% ± 1.26% |
| MLP3L | 0.05 | 2 | 36.22% ± 1.57% | 48.95% ± 2.14% | 33.68% ± 5.54% | 49.86% ± 1.78% | 51.67% ± 1.12% | 47.51% ± 2.07% |
| MLP3L | 0.05 | 4 | 35.73% ± 0.58% | 50.70% ± 1.37% | 33.68% ± 5.54% | 47.94% ± 1.78% | 61.29% ± 2.29% | 47.51% ± 2.07% |
| MLP3L | 0.05 | 8 | 36.53% ± 1.95% | 50.01% ± 1.45% | 33.68% ± 5.54% | 48.18% ± 1.90% | 66.54% ± 0.94% | 47.51% ± 2.07% |
| MLP3L | 0.05 | 16 | 36.29% ± 1.40% | 46.73% ± 1.54% | 33.68% ± 5.54% | 46.23% ± 2.89% | 65.40% ± 1.31% | 47.51% ± 2.07% |
| MLP3L | 0.05 | 64 | 35.81% ± 1.50% | 37.01% ± 0.96% | 33.68% ± 5.54% | 50.71% ± 2.04% | 60.29% ± 1.25% | 47.51% ± 2.07% |
| MLP3L | 0.05 | 128 | 36.73% ± 0.92% | 35.72% ± 1.05% | 33.68% ± 5.54% | 52.83% ± 1.75% | 49.99% ± 2.42% | 47.51% ± 2.07% |
| MLP3L | 0.25 | 2 | 35.58% ± 1.10% | 50.01% ± 1.23% | 25.24% ± 1.46% | 50.62% ± 1.81% | 49.56% ± 4.81% | 44.86% ± 2.64% |
| MLP3L | 0.25 | 4 | 35.89% ± 1.07% | 50.75% ± 1.17% | 25.24% ± 1.46% | 50.80% ± 1.97% | 60.13% ± 3.06% | 44.86% ± 2.64% |
| MLP3L | 0.25 | 8 | 36.07% ± 1.12% | 50.48% ± 1.74% | 25.24% ± 1.46% | 50.20% ± 1.91% | 68.35% ± 1.50% | 44.86% ± 2.64% |
| MLP3L | 0.25 | 16 | 36.53% ± 1.86% | 46.08% ± 2.26% | 25.24% ± 1.46% | 49.50% ± 1.98% | 65.71% ± 1.19% | 44.86% ± 2.64% |
| MLP3L | 0.25 | 64 | 36.79% ± 2.65% | 35.48% ± 1.09% | 25.24% ± 1.46% | 47.13% ± 3.17% | 59.08% ± 1.42% | 44.86% ± 2.64% |
| MLP3L | 0.25 | 128 | 35.26% ± 1.04% | 34.54% ± 0.99% | 25.24% ± 1.46% | 49.64% ± 2.48% | 50.05% ± 1.38% | 44.86% ± 2.64% |
| MLP3L | 1.00 | 2 | 35.83% ± 1.46% | 52.38% ± 0.98% | 23.50% ± 2.21% | 50.68% ± 1.14% | 60.35% ± 1.49% | 47.01% ± 2.84% |
| MLP3L | 1.00 | 4 | 36.24% ± 1.05% | 52.06% ± 1.19% | 23.50% ± 2.21% | 51.02% ± 1.67% | 52.26% ± 4.59% | 47.01% ± 2.84% |
| MLP3L | 1.00 | 8 | 36.57% ± 1.62% | 50.67% ± 1.78% | 23.50% ± 2.21% | 52.75% ± 1.26% | 69.51% ± 1.41% | 47.01% ± 2.84% |
| MLP3L | 1.00 | 16 | 36.38% ± 1.80% | 44.75% ± 1.25% | 23.50% ± 2.21% | 51.25% ± 1.76% | 66.56% ± 1.43% | 47.01% ± 2.84% |
| MLP3L | 1.00 | 64 | 36.95% ± 1.51% | 35.43% ± 1.08% | 23.50% ± 2.21% | 53.60% ± 2.68% | 56.60% ± 1.07% | 47.01% ± 2.84% |
| MLP3L | 1.00 | 128 | 34.68% ± 0.73% | 33.93% ± 0.66% | 23.50% ± 2.21% | 54.02% ± 2.34% | 49.86% ± 1.56% | 47.01% ± 2.84% |

Table B.4: Average accuracies and standard deviations for CWR and MFPT case studies based spectrograms and the usage of the CNN-VAE model.

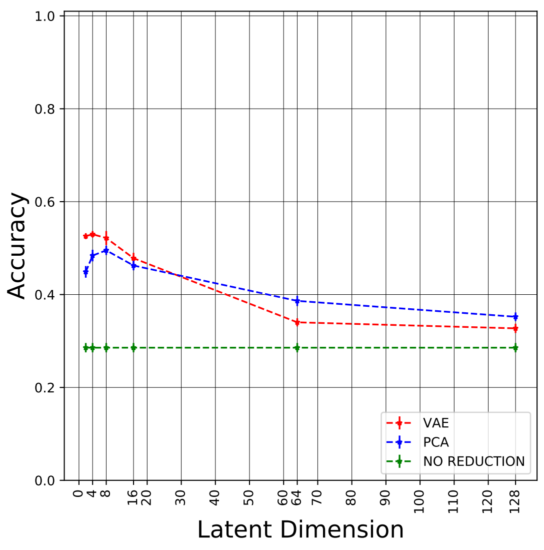
| **Classifier** | $\varepsilon$% | $k$ | **CWR - CNN VAE Arch. #2** | **MFPT - CNN VAE Arch. #2** |
|---|---|---|---|---|
| MLP1L | 0.01 | 2 | 79.89% ± 1.59% | 77.78% ± 2.27% |
| MLP1L | 0.01 | 4 | 85.63% ± 2.03% | 79.46% ± 2.18% |
| MLP1L | 0.01 | 8 | 82.45% ± 1.67% | 74.70% ± 2.65% |
| MLP1L | 0.01 | 16 | 80.84% ± 2.53% | 66.16% ± 3.88% |
| MLP1L | 0.01 | 64 | 81.61% ± 2.20% | 49.01% ± 3.72% |
| MLP1L | 0.01 | 128 | 84.85% ± 2.34% | 47.06% ± 1.60% |
| MLP1L | 0.05 | 2 | 86.62% ± 0.71% | 80.84% ± 0.56% |
| MLP1L | 0.05 | 4 | 95.09% ± 0.39% | 84.51% ± 0.90% |
| MLP1L | 0.05 | 8 | 93.59% ± 0.67% | 83.41% ± 1.06% |
| MLP1L | 0.05 | 16 | 94.71% ± 0.29% | 83.03% ± 1.00% |
| MLP1L | 0.05 | 64 | 95.58% ± 0.59% | 74.77% ± 1.47% |
| MLP1L | 0.05 | 128 | 94.92% ± 0.57% | 69.33% ± 1.60% |
| MLP1L | 0.25 | 2 | 88.70% ± 0.25% | 81.44% ± 0.28% |
| MLP1L | 0.25 | 4 | 96.67% ± 0.23% | 86.67% ± 0.50% |
| MLP1L | 0.25 | 8 | 96.07% ± 0.30% | 85.24% ± 0.62% |
| MLP1L | 0.25 | 16 | 96.64% ± 0.21% | 85.34% ± 0.81% |
| MLP1L | 0.25 | 64 | 97.51% ± 0.31% | 85.43% ± 0.69% |
| MLP1L | 0.25 | 128 | 97.54% ± 0.28% | 84.88% ± 0.69% |
| MLP1L | 1.00 | 2 | 91.57% ± 0.25% | 83.46% ± 0.34% |
| MLP1L | 1.00 | 4 | 96.83% ± 0.17% | 88.46% ± 0.22% |
| MLP1L | 1.00 | 8 | 97.19% ± 0.24% | 88.53% ± 0.85% |
| MLP1L | 1.00 | 16 | 96.92% ± 0.20% | 87.56% ± 0.59% |
| MLP1L | 1.00 | 64 | 98.01% ± 0.20% | 88.34% ± 0.76% |
| MLP1L | 1.00 | 128 | 98.29% ± 0.22% | 89.66% ± 0.61% |
| MLP1LDO | 0.01 | 2 | 80.42% ± 2.06% | 78.87% ± 1.71% |
| MLP1LDO | 0.01 | 4 | 84.36% ± 2.07% | 81.83% ± 2.15% |
| MLP1LDO | 0.01 | 8 | 82.25% ± 2.97% | 78.69% ± 2.13% |
| MLP1LDO | 0.01 | 16 | 84.35% ± 2.08% | 76.47% ± 2.25% |
| MLP1LDO | 0.01 | 64 | 83.94% ± 2.09% | 65.76% ± 1.49% |
| MLP1LDO | 0.01 | 128 | 86.45% ± 2.04% | 64.30% ± 1.93% |
| MLP1LDO | 0.05 | 2 | 86.58% ± 0.85% | 80.86% ± 0.30% |
| MLP1LDO | 0.05 | 4 | 95.14% ± 0.28% | 85.74% ± 0.57% |
| MLP1LDO | 0.05 | 8 | 94.19% ± 0.57% | 85.99% ± 0.56% |
| MLP1LDO | 0.05 | 16 | 95.45% ± 0.30% | 85.97% ± 0.68% |
| MLP1LDO | 0.05 | 64 | 96.37% ± 0.34% | 83.66% ± 1.52% |
| MLP1LDO | 0.05 | 128 | 95.73% ± 0.39% | 83.36% ± 1.63% |
| MLP1LDO | 0.25 | 2 | 88.57% ± 0.27% | 81.47% ± 0.17% |
| MLP1LDO | 0.25 | 4 | 96.88% ± 0.18% | 86.91% ± 0.31% |
| MLP1LDO | 0.25 | 8 | 96.41% ± 0.25% | 87.87% ± 0.33% |
| MLP1LDO | 0.25 | 16 | 97.14% ± 0.22% | 88.98% ± 0.54% |
| MLP1LDO | 0.25 | 64 | 97.84% ± 0.14% | 88.65% ± 0.56% |
| MLP1LDO | 0.25 | 128 | 97.80% ± 0.38% | 89.34% ± 0.32% |
| MLP1LDO | 1.00 | 2 | 91.43% ± 0.14% | 83.45% ± 0.19% |
| MLP1LDO | 1.00 | 4 | 96.61% ± 0.18% | 88.28% ± 0.36% |
| MLP1LDO | 1.00 | 8 | 96.99% ± 0.20% | 90.36% ± 0.28% |
| MLP1LDO | 1.00 | 16 | 96.99% ± 0.17% | 90.89% ± 0.41% |
| MLP1LDO | 1.00 | 64 | 97.98% ± 0.22% | 91.87% ± 0.20% |
| MLP1LDO | 1.00 | 128 | 98.35% ± 0.21% | 92.58% ± 0.43% |
| MLP3L | 0.01 | 2 | 66.13% ± 3.38% | 77.21% ± 2.11% |
| MLP3L | 0.01 | 4 | 84.83% ± 1.74% | 82.86% ± 1.48% |
| MLP3L | 0.01 | 8 | 84.82% ± 2.08% | 80.99% ± 1.48% |
| MLP3L | 0.01 | 16 | 84.90% ± 1.76% | 78.51% ± 1.76% |
| MLP3L | 0.01 | 64 | 86.26% ± 1.61% | 71.04% ± 2.99% |
| MLP3L | 0.01 | 128 | 89.53% ± 1.74% | 71.01% ± 1.92% |
| MLP3L | 0.05 | 2 | 70.95% ± 2.97% | 79.97% ± 0.84% |
| MLP3L | 0.05 | 4 | 94.09% ± 0.43% | 82.90% ± 1.26% |
| MLP3L | 0.05 | 8 | 93.20% ± 0.72% | 85.02% ± 0.39% |
| MLP3L | 0.05 | 16 | 94.80% ± 0.69% | 85.37% ± 1.05% |
| MLP3L | 0.05 | 64 | 96.15% ± 0.47% | 83.89% ± 0.75% |
| MLP3L | 0.05 | 128 | 96.03% ± 0.35% | 84.45% ± 0.77% |
| MLP3L | 0.25 | 2 | 71.68% ± 1.86% | 80.53% ± 0.60% |
| MLP3L | 0.25 | 4 | 95.89% ± 0.23% | 85.03% ± 1.06% |
| MLP3L | 0.25 | 8 | 95.01% ± 0.53% | 86.05% ± 0.81% |
| MLP3L | 0.25 | 16 | 96.78% ± 0.36% | 87.41% ± 1.09% |
| MLP3L | 0.25 | 64 | 97.57% ± 0.25% | 86.98% ± 0.73% |
| MLP3L | 0.25 | 128 | 97.56% ± 0.21% | 88.36% ± 1.12% |
| MLP3L | 1.00 | 2 | 71.24% ± 3.42% | 82.29% ± 0.39% |
| MLP3L | 1.00 | 4 | 95.09% ± 0.28% | 87.45% ± 0.44% |
| MLP3L | 1.00 | 8 | 95.67% ± 0.44% | 88.18% ± 0.55% |
| MLP3L | 1.00 | 16 | 96.37% ± 0.27% | 88.49% ± 0.41% |
| MLP3L | 1.00 | 64 | 97.80% ± 0.23% | 86.84% ± 0.76% |
| MLP3L | 1.00 | 128 | 98.07% ± 0.24% | 87.39% ± 0.77% |

# Appendix C

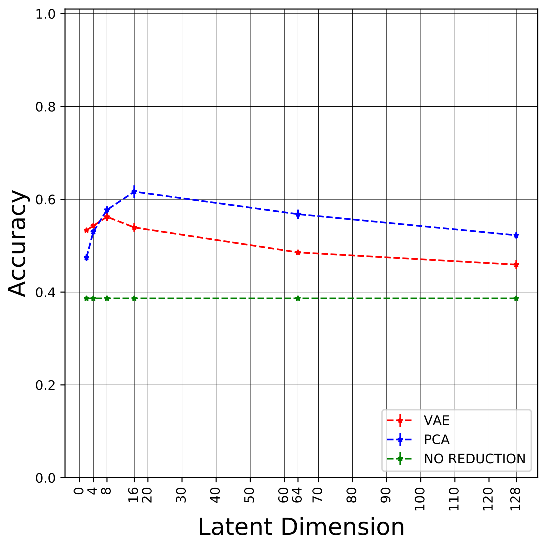
# Raw Vibration Signal Dataset Plots



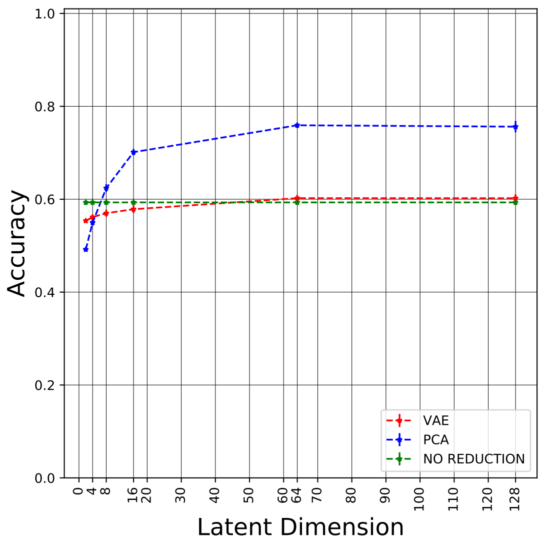
(a) CWR Raw Data - MLP1L - $\varepsilon = 1\%$



(b) CWR Raw Data - MLP1L - $\varepsilon = 5\%$

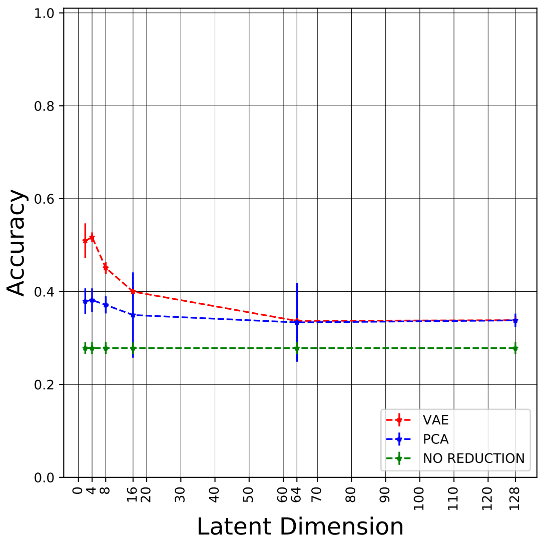


(c) CWR Raw Data - MLP1L - $\varepsilon = 25\%$

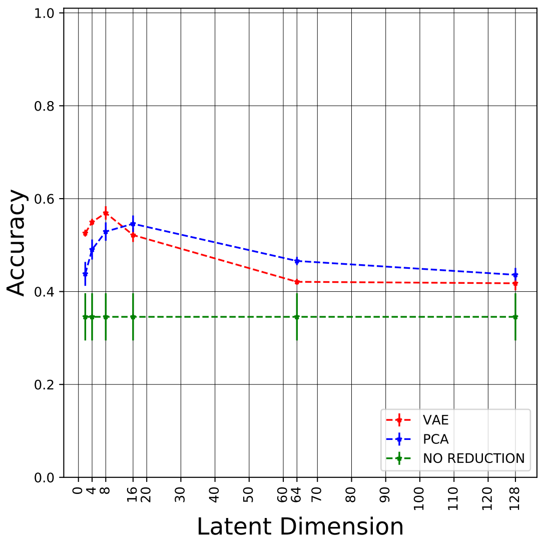


(d) CWR Raw Data - MLP1L - $\varepsilon = 100\%$

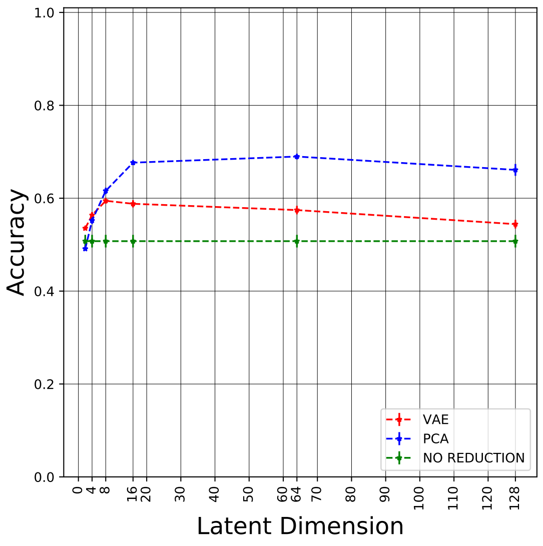
Figure C.1: Average accuracy versus latent space dimension for the CWR dataset and MLP1L classifier with chunks of the original vibration signal.



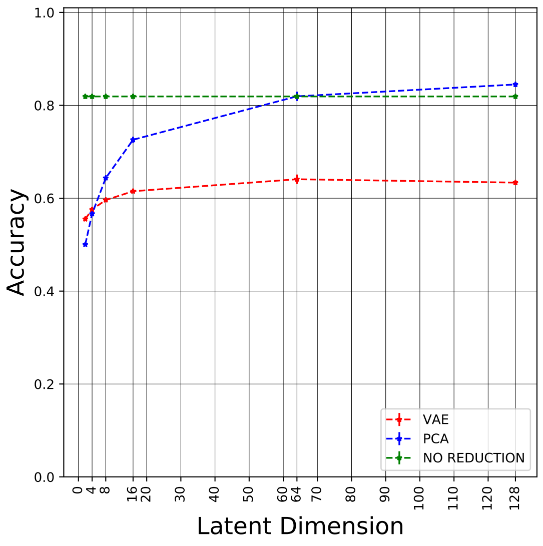
(a) CWR Raw Data - MLP1LDO - $\varepsilon = 1\%$



(b) CWR Raw Data - MLP1LDO - $\varepsilon = 5\%$

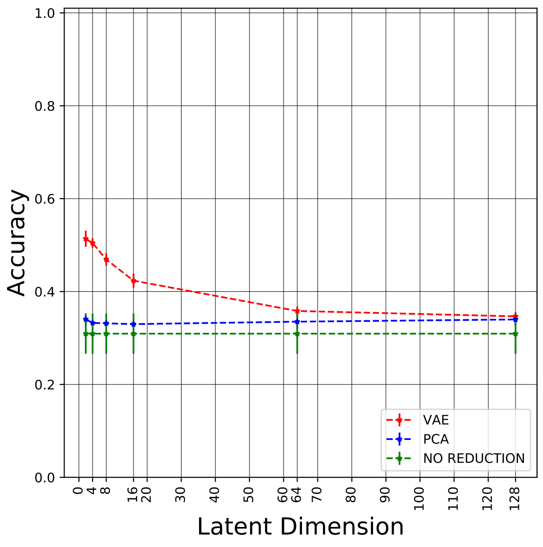


(c) CWR Raw Data - MLP1LDO - $\varepsilon = 25\%$

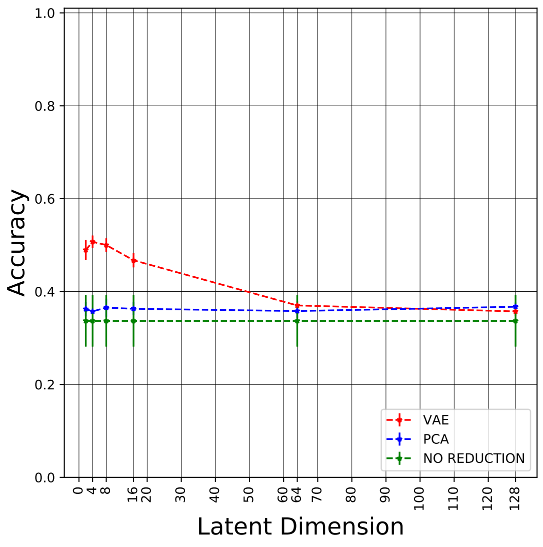


(d) CWR Raw Data - MLP1LDO - $\varepsilon = 100\%$

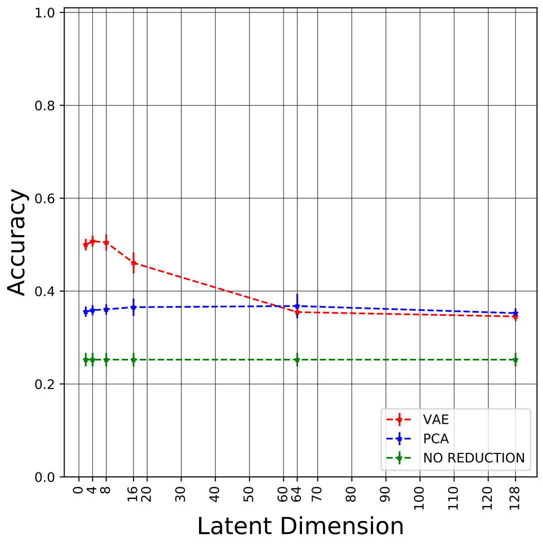
Figure C.2: Average accuracy versus latent space dimension for the CWR dataset and MLP1LDO classifier with chunks of the original vibration signal.



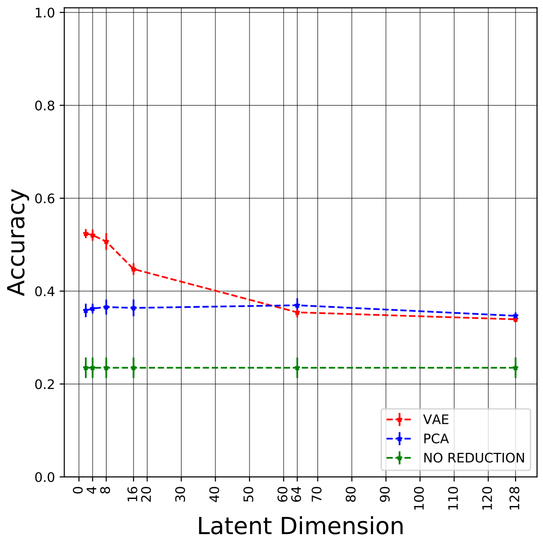
(a) CWR Raw Data - MLP3LDO - $\varepsilon = 1\%$



(b) CWR Raw Data - MLP3LDO - $\varepsilon = 5\%$

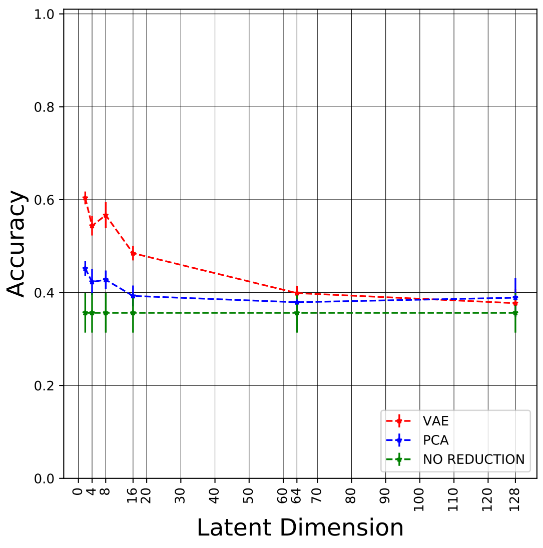


(c) CWR Raw Data - MLP3LDO - $\varepsilon = 25\%$

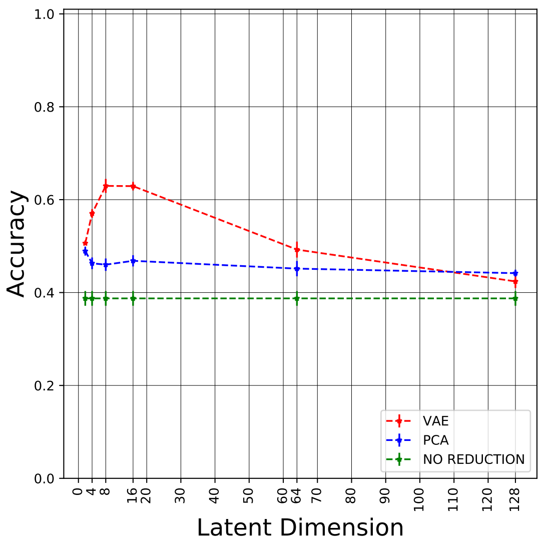


(d) CWR Raw Data - MLP3LDO - $\varepsilon = 100\%$

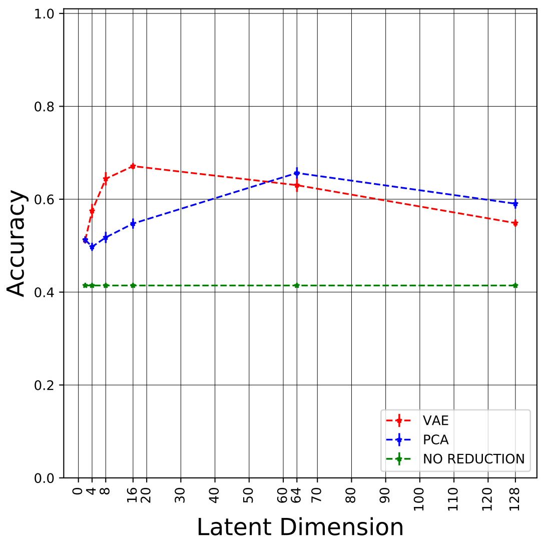
Figure C.3: Average accuracy versus latent space dimension for the CWR dataset and MLP3LDO classifier with chunks of the original vibration signal.



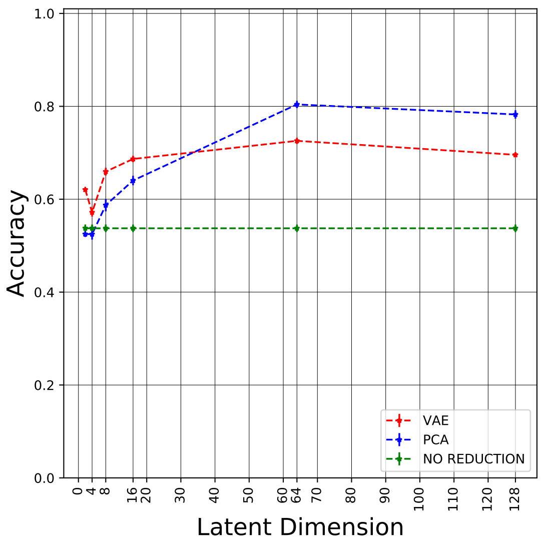
(a) MFPT Raw Data - MLP1L - $\varepsilon = 1\%$



(b) MFPT Raw Data - MLP1L - $\varepsilon = 5\%$

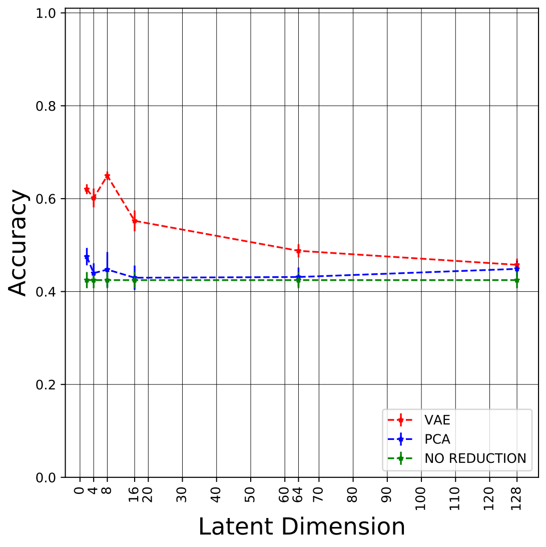


(c) MFPT Raw Data - MLP1L - $\varepsilon = 25\%$

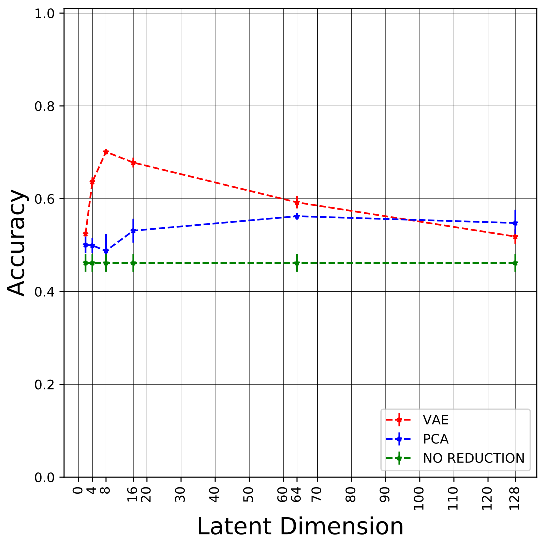


(d) MFPT Raw Data - MLP1L - $\varepsilon = 100\%$

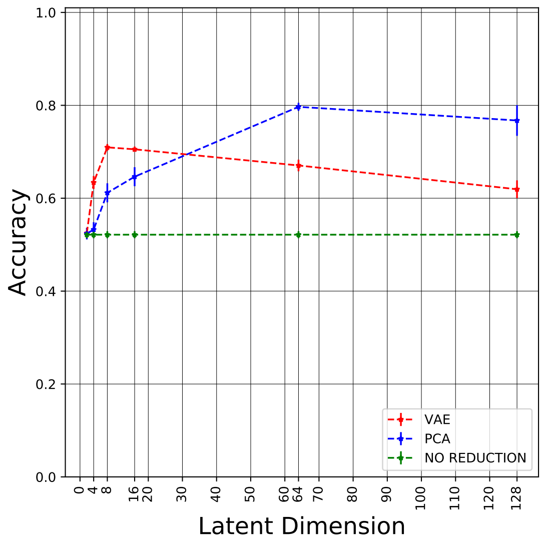
Figure C.4: Average accuracy versus latent space dimension for the MFPT dataset and MLP1L classifier with chunks of the original vibration signal.



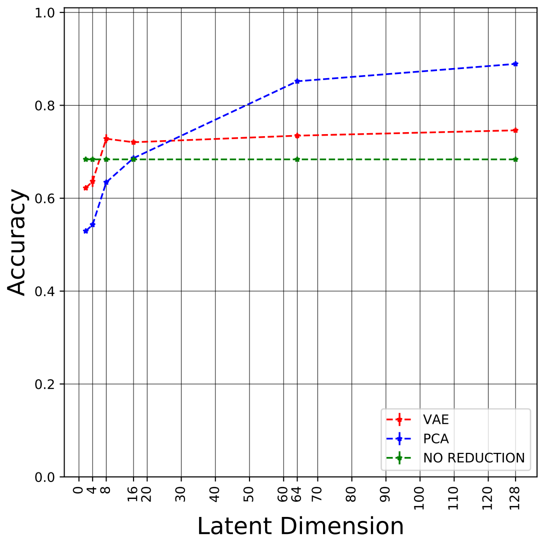
(a) MFPT Raw Data - MLP1LDO - $\varepsilon = 1\%$



(b) MFPT Raw Data - MLP1LDO - $\varepsilon = 5\%$

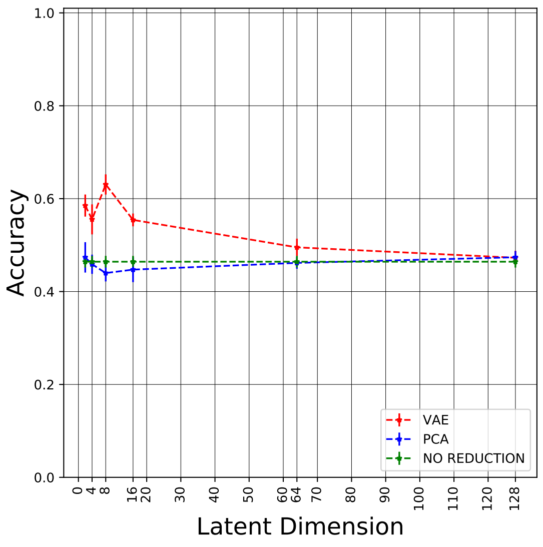


(c) MFPT Raw Data - MLP1LDO - $\varepsilon = 25\%$

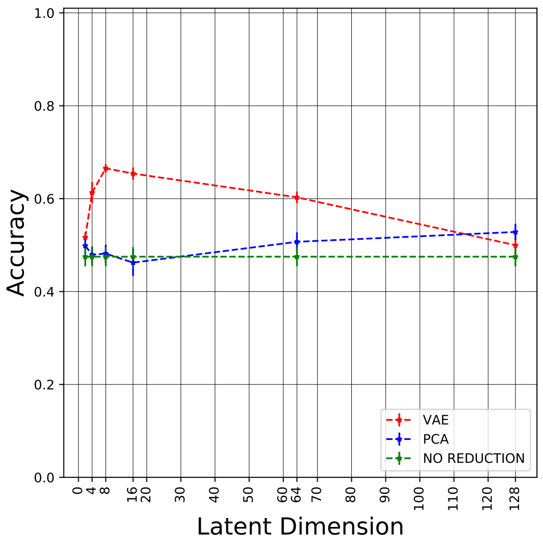


(d) MFPT Raw Data - MLP1LDO - $\varepsilon = 100\%$

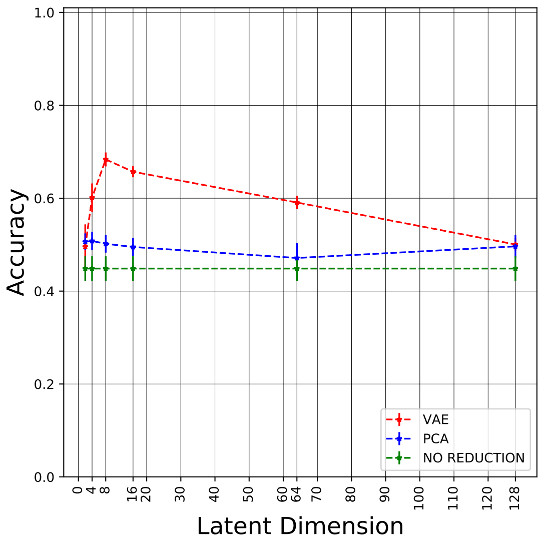
Figure C.5: Average accuracy versus latent space dimension for the MFPT dataset and MLP1LDO classifier with chunks of the original vibration signal.



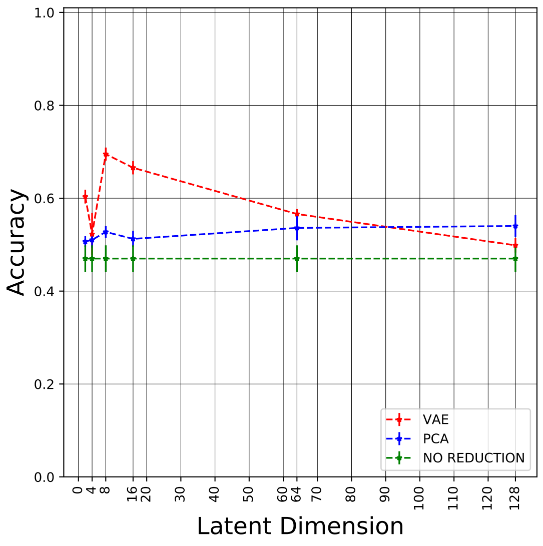
(a) MFPT Raw Data - MLP3LDO - $\varepsilon = 1\%$



(b) MFPT Raw Data - MLP3LDO - $\varepsilon = 5\%$



(c) MFPT Raw Data - MLP3LDO - $\varepsilon = 25\%$



(d) MFPT Raw Data - MLP3LDO - $\varepsilon = 100\%$

Figure C.6: Average accuracy versus latent space dimension for the MFPT dataset and MLP3LDO classifier with chunks of the original vibration signal.