ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

STATISTICAL FEATURE LEARNING AND SIGNAL GENERATION FOR TIME-SERIES SENSOR SIGNALS

Ph.D. THESIS

Erkan KARAKUŞ

Department of Computer Engineering

Computer Engineering Programme

MAY 2022

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

STATISTICAL FEATURE LEARNING AND SIGNAL GENERATION FOR TIME-SERIES SENSOR SIGNALS

Ph.D. THESIS

Erkan KARAKUŞ
(504992432)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Prof. Dr. Hatice KÖSE

MAY 2022

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

ZAMAN SERİSİ SENSÖR İŞARETLERİ İÇİN ÖZNİTELİK ÖĞRENİMİ VE İŞARET ÜRETİMİ

DOKTORA TEZİ

Erkan KARAKUŞ
(504992432)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Hatice KÖSE

MAYIS 2022

Erkan KARAKUŞ, a Ph.D. student of ITU Graduate School student ID 504992432, successfully defended the thesis entitled “STATISTICAL FEATURE LEARNING AND SIGNAL GENERATION FOR TIME-SERIES SENSOR SIGNALS”, which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Hatice KÖSE** İstanbul Technical University	.....
Jury Members :	**Assist. Prof. Dr. İslem REKİK** Istanbul Technical University	.....
	Assist. Prof. Dr. Figen ÖZEN Haliç University	.....
	Prof. Dr. Duygun EROL BARKANA Yeditepe University	.....
	Assoc. Prof. Dr. İlkay ÖKSÜZ Istanbul Technical University	.....

Date of Submission : 23 May 2022
Date of Defense : 31 May 2022

To Sabine and Pythagore,

FOREWORD

First and foremost, I am extremely grateful to my supervisor, Prof. Dr. Hatice KÖSE for her precious advice, guidance, continuous support, and patience during my Ph.D. study. Her immense knowledge and plentiful experience have encouraged me all the time in my academic research.
I would also like to express my sincere gratitude to Ph.D. thesis supervision jury members Assoc. Prof. Figen ÖZEN and Assoc. Prof. İslem REKİK for their invaluable feedback, support, and encouragement. Without their precious supervision and guidance, my thesis study wouldn't be complete.
I would like to extend my gratitude to Prof. Dr. Ali YAPAR, Özgür AKSOY, Sabine BUCHMANN, and my parents for their encouragement and support during my thesis study.

May 2022

Erkan KARAKUŞ
(Computer Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

RBM : Restricted Boltzmann Machine
CRBM : Conditional Restricted Boltzmann Machine
HAR : Human Activity Recognition
UCI : University of California, Irvine
GAN : Generative Adversarial Network
WGAN : Wasserstein Generative Adversarial Network
WGAN-GP : Wasserstein Generative Adversarial Network with Gradient Penalty
FT : Fourier Transform
FFT : Fast Fourier Transform
SFFS-SFBS : Sequential Floating Forward–Backward Search
DWT : Discrete Wavelet Transform
HHT : Hilbert-Huang Transform
MRF : Markov Random Field
KL : Kullback-Leibler
JS : Jensen-Shannon

SYMBOLS

h_i	**:** Binary State of Hidden Layer Unit
v_i	**:** Binary State of Visible Layer Unit
a_i	**:** Visible Layer Unit Bias
b_i	**:** Hidden Layer Unit Bias
w_{ij}	**:** Connection weights between visible and hidden layer units
$E(v;\mathbf{h})$	**:** Energy function of joint configuration of visible and hidden units
$p(v,h)$	**:** Joint probability distribution of visible and hidden layer units
Z	**:** Normalization constant or partition function
$p(v)$	**:** Marginal probability distribution of visible vector
$\sigma(x)$	**:** Logistic Sigmoid Function
D	**:** Discriminator
G	**:** Generator
$V(D,G)$	**:** Value Function
θ_g	**:** Generator Multilayer Perceptron Parameters
θ_d	**:** Discriminator Multilayer Perceptron Parameters
z	: Latent Variable
p_{data}	: Real Data Distribution
p_g	: Generator Distribution
p_z	: Latent Space Distribution
p_r	: Real Data Distribution
$W(\mathbf{P}_r, \mathbf{P}_g)$	: Wasserstein Distance
$\Pi(\mathbf{P}_r, \mathbf{P}_g)$	: Set of All Joint Distributions
f	: 1-Lipschitz function
Q_E	: Euclidean Distance
Q_{Can}	: Canberra Distance
$\mathbb{E}$	: Expected Value
H	: Hypothesis
$\mathfrak{H}$	: Undirected Graph
$\mathbf{G}$	: Directed Graph

LIST OF TABLES

Page

LIST OF FIGURES

Page

STATISTICAL FEATURE LEARNING AND SIGNAL GENERATION FOR TIME-SERIES SENSOR SIGNALS

SUMMARY

The Human Activity Recognition (HAR) problem has attracted substantial attention from academia. HAR has many applications like smart home assisted living systems, healthcare monitoring systems, sports activity monitoring, and monitoring indoor and outdoor activities. HAR applications involve advanced machine learning techniques to identify and classify human activities by leveraging video cameras, wearable sensors, or any other signal like Wi-Fi or radar which eventually encodes the human activity. Human activities are encoded in signals and signal processing techniques are required to pre-process raw signals to filter out high-frequency components and to frame the signals into the fixed-length window.

Wearable smart electronics are widely used in human daily life. Those smart devices contain sensors like accelerometer and gyroscope to measure triaxial acceleration and angular velocity respectively. Smartwatches, smartphones, or any such wearable sensor devices contain out-of-the-box sensors embedded in the device.

Identification and classification of human activities from such signals by leveraging machine learning techniques require features to be extracted from the signal which represents the corresponding human activity. Many feature extraction techniques from such time-series signals exist in the literature.

Time and frequency domain-based feature extraction is a widely used technique for sensor-based human activity classification. To train deep learning models, one needs features to be extracted from the signal. Though time and frequency domain feature extraction techniques are very efficient, the selection of the time and frequency domain features may have a significant impact on the overall classification accuracy. Alternatively, energy-based generative models eliminate the need for a feature extraction layer in the learning pipeline. Deep Belief Networks are alternatives to deep learning models eliminating the need for time and frequency-based feature extraction for sensor-based human activity classification: Restricted Boltzmann Machines (RBM) are the building blocks of Deep Belief Networks. RBMs are energy-based probabilistic graphical models which factorize the probability distribution of a random variable over a binary probability distribution. The visible layer of RBMs represents the real-valued random variable and the hidden layer represents the corresponding binary valued probability distribution.

Conditional Restricted Boltzmann Machine (CRBM) is an extension to RBMs and is strong in capturing temporal dependency information encoded in time-series signals. They can be used in the classification of sensor-based human activities. The capacity of CRBM by factorizing a real-valued random variable probability distribution over a binary valued probability distribution eliminates the need for feature extraction from the signal by applying certain feature extraction techniques.

This work shows how CRBM is trained to learn signal features. Once trained the signal is generated and reconstructed by the trained model. Along with CRBM, the results of other generative models RBM, GAN, WGAN-GP, and predictive model LSTM are also presented. To compare the performance of the models, similarity metrics are used as a performance criterion to show the performance of the generative models in generating the signals closest to the real signals. Euclidean, Canberra, and Dynamic Time Warping (DTW) distances are used as performance criteria. The results indicate that CRBM outperforms GAN, WGAN-GP, and RBM generative models in generating the signal closest to the original signal. LSTM performs close to CRBM. The capacity of the CRBM in generating signals closest to the original signal indicates that CRBM can learn features from the signal and can also be used in supervised classification.

ZAMAN SERİSİ SENSÖR İŞARETLERİ İÇİN ÖZNİTELİK ÖĞRENİMİ VE İŞARET ÜRETİMİ

ÖZET

İnsan Aktivitesi Tanıma (İAT) problemi akademi dünyasında oldukça ilgi gören bir araştırma konusu olmuştur. İAT, akıllı ev destekli yaşam sistemleri, sağlık izleme sistemleri, spor, iç ve dış mekân aktivitelerini izleme gibi birçok farklı uygulamada görülebilmektedir. İAT uygulamaları, video kameralar, giyilebilir sensörler veya insan hareketlerinin kodlanmış şekilde temsil eden WiFi veya radar işareti gibi diğer herhangi bir işaretten yararlanarak insan faaliyetlerini tanımlamak ve sınıflandırmak için gelişmiş makine öğrenimi tekniklerini içerir. Tanımlanmak ve sınıflandırılmak istenilen insan aktiviteleri uygulamadan uygulamaya farklılık gösterebilmektedir.

Yürüme, koşma, oturma, uyuma, ayakta durma, duş alma, yemek pişirme, araç sürme, kapı açma ve normal dışı aktiviteler gibi çeşitli insan hareketleri İAT ile tanınabilir. Belli bir spor etkinliğinde yürüme, koşma, durma veya sprint atma gibi hareketlerin algılanması ve tanınması ilgi alanı dahilinde olabilirken, başka bir spor etkinliğinde o spor etkinliğine özgü hareketleri algılamak ve tanımak söz konusu olabilmektedir. Sağlık izleme uygulamalarında yaşlı, kronik rahatsızlığı bulunan bireylerde yere düşme aktivitesini bu tür sensörler yardımıyla tanıyabilmek hayati önem taşımaktadır.

Giyilebilir akıllı cihazlar insanların günlük yaşamında yaygın olarak kullanılmaktadır. Bu tür akıllı cihazlar, üç eksenli ivme ve açısal hızı ölçmek için ivmeölçer ve jiroskop gibi sensörler içerir. Akıllı saatler, akıllı telefonlar veya bu tür giyilebilir sensör cihazları, cihaza gömülü olarak kullanıma hazır bu tür sensörler ile donatılmıştır.

İnsan aktivitesi tanıma, giyilebilir akıllı cihazlar veya akıllı telefonlar tarafından kaydedilen ivmeölçer, jiroskop gibi sensör verilerinin bilinen hareketlere göre sınıflandırılması sorunudur. Sensörün vücut üzerindeki doğru yerleşimi hareketin sınıflandırılma hassasiyeti açısından önem taşımaktadır. Eğer aynı anda birden fazla aktivite söz konusu olursa, hareketin tanınması zorlaşabilmektedir. Öte yandan eğer sensörler bir ev ortamında kullanılıyorsa, evde yaşayan birden fazla kişinin hareketlerini ayırt etmek ve sınıflandırmak zor hale gelebilmektedir. Aktiviteye bağlı olarak her saniyede sensör tarafından üretilen çok sayıda gözlem, bu gözlemlerin zamana bağımlılığı ve sensör verilerinin hareketlerle ilişiklendirmenin net bir yolunun olmaması göz önüne alındığında İAT zor bir problem olarak karşımıza çıkmaktadır.

İnsan hareketlerinin kodlanmış olduğu bu tür işaretler makina öğrenmesine tabi tutulmadan önce işaret işleme teknikleri kullanılarak bir ön işlemeye tabi tutulur. Ham veriler, bu işaret işleme tekniğinin uygulanması esnasında yüksek frekanslı bileşenler filtrelenir ve işaretler sabit uzunluklu pencere yöntemi ile çerçevelendikten sonra normalize/standardize edilerek makina öğreniminde kullanılmak üzere hazır hale getirilir. Doğrusal olmayan alçak geçiren medyan filtre, kayan pencere medyan filtre, Laplace ve Gauss filtreleri yüksek frekanslı gürültüyü filtre etmek için kullanılan işaret önişleme teknikleri arasında yer almaktadır. İnsan hareketine ait ivmeyi elde

edebilmek için, yerçekimi ivmesinin ivmeölçer sensör işaretinden filtrelenmesi gerekir. Bu amaçla yüksek geçiren bir filtre uygulanarak yerçekimi ivmesi ivmeölçer sensör işaretinden filtrelenebilir.

Literatürde İAT problemine yönelik çeşitli makina öğrenmesi tekniklerinin kullanımı görülmektedir. Karar Ağaçları, K En Yakın Komşu (KNN), Destek Vektör Makinası, Evrişimli Sinirsel Ağlar, insan aktivitesi tanıma ve sınıflandırmada kullanılan makina öğrenme teknikleri arasında gösterilebilir.

Makine Öğrenmesi tekniklerinden yararlanarak bu tür işaretlerden insan faaliyetlerinin tanımlanması ve sınıflandırılması, ilgili insan aktivitesini temsil eden işaretten özniteliklerin belirlenmesi, seçilmesi ve bu özniteliklerin işaretten çıkarımını gerektirir. Özniteliklerin, işaretten insan hareketini en yakın temsil edecek şekilde seçilmesi ve ayırt edici olmayan özniteliklerin elenmesi gerekmektedir. Ardışıl ileri-geri seçim, Pudil algoritması tabanlı yüzen ileri-geri arama bu tür öznitelik belirlemede kullanılan teknikler arasında gösterilebilir. Literatürde bu tür zaman serisi işaretlerinden birçok öznitelik seçimi ve çıkarımı teknikleri bulunmaktadır.

Derin öğrenme modellerini eğitmek amacıyla, bu tür sensör işaretlerinden işarete ait özniteliklerin çıkarılması için çeşitli teknikler literatürde sunulmuştur. Zaman ve frekans bölgesi tabanlı öznitelik çıkarımı, sensör tabanlı insan etkinliği sınıflandırması için yaygın olarak kullanılan bir tekniktir. Ortalama değer, medyan, standart sapma, entropi, korelasyon katsayıları gibi yapısal tabanlı öznitelik çıkarım tekniklerinin yanı sıra, FFT (Fast Fourier Transform), DWT (Discrete Wavelet Transform), HHT (Hilbert Huang Transform) gibi dönüşüm teknikleri de kullanılarak işarete ait özniteliklerin belirlenmesi söz konusu olabilmektedir.

Zaman ve frekans bölgesi öznitelik çıkarma teknikleri çok verimli olmasına rağmen, zaman ve frekans bölgesi özniteliklerinin seçimi, genel sınıflandırma hassasiyeti üzerinde önemli bir etkiye sahip olabilir. Alternatif olarak, enerji tabanlı üretken modeller, öğrenme modelinde bir öznitelik çıkarma katmanına olan ihtiyacı ortadan kaldırır. Derin İnanç Ağları, sensör tabanlı insan etkinliği sınıflandırması için zaman ve frekans tabanlı öznitelik çıkarma ihtiyacını ortadan kaldıran derin öğrenme modellerine alternatiftir: Kısıtlı Boltzmann Makineleri (KBM), Derin İnanç Ağlarının yapı taşlarıdır. KBM'ler, bir rastgele değişkenin olasılık dağılımını ikili bir olasılık dağılımı üzerinde faktorize eden enerji tabanlı olasılıksal grafik modellerdir. Görünür RBM katmanı, gerçek değerli rastgele değişkeni ve gizli katman, karşılık gelen ikili değerli olasılık dağılımını temsil ederler.

Koşullu Kısıtlı Boltzmann Makineleri (KKBM), KBM modelinden farklı olarak zaman serisi işaretlerine özgü işaretin kendi içerisindeki zamana bağımlılığını algılamada güçlü bir model olarak karşımıza çıkmaktadır. KKBM modelinin, KBM modeline göre işaretin zaman içerisindeki özgü bağımlılığını algılama kabiliyetinden ötürü, sensör tabanlı insan aktivitesi tanımlama ve sınıflandırma makina öğrenmesi problemlerinde güçlü bir model olarak karşımıza çıkmaktadır.

Bu çalışmada insan aktivite verisi olarak Kaliforniya Üniversitesi, Irvine (UCI) veri deposundan sağlanan insan aktivite verileri kullanılmıştır. Bu veriler 19-48 yaşlar arasında 30 gönüllüden elde edilen aktivite sensör kayıtlarını içermektedir. Verileri elde etmek için katılımcıların bel hizasına giyilen bir bel çantası içerisine yerleştirilmiş Samsung Galaxy S II akıllı telefon ile aktivite sensör kayıtları gerçekleştirilmiştir. Kullanılan Samsung marka akıllı telefon 3-eksenli doğrusal ivme ve 3-eksenli açısal hızı ölçebilecek sensörler ile donatılmıştır. Deney sırasında video kaydı kullanılmış ve veriler bu video kaydı yardımıyla etiketlenmiştir.

KKBM'nin gerçek değerli bir rastgele değişkene ait olasılık dağılımının, ikili değerli bir olasılık dağılımına dönüştürebilme kapasitesi, makina öğrenmesinde gerekli olan öznitelik çıkarım ihtiyacını zaman serisi işaretleri için ortadan kaldırır. Bu çalışma ile KKBM modelinin sensör tabanlı insan aktivitesi tanıma probleminde, zaman serisi bir sensör işaretinin özniteliklerini öğrenmek için nasıl eğitildiği ve eğitilen modelden işaretin yeniden nasıl elde edilebileceği gösterilmiştir. KKBM model eğitiminin ardından işaret, eğitilen model tarafından üretilir ve yeniden yapılandırılır. İşaretin yeniden elde edilmesi amacıyla eğitim veri kümesinden farklı olarak test veri kümesi kullanılmıştır. KKBM ile birlikte, KBM, GAN, WGAN-GP diğer üretken modellerin sonuçları da sunulmaktadır. Üretken modellerin performansını karşılaştırmak için, üretken modellerin gerçek işaretlere en yakın işaretleri üretmedeki performansını göstermek için bir performans kriteri olarak benzerlik metrikleri kullanılır. Euclid, Canberra ve Dinamik Zaman Bükme (DTW) mesafeleri bu tür performans kriterleri olarak kullanılır.

1. INTRODUCTION

Smart devices like smartphones and wearable devices take a major place in individuals' lives. Wearable devices equipped with a set of sensors accelerometer, gyroscope, magnetometer, and GPS play a key role in the ubiquitous computing domain. In addition to their sensing capacity, the wearable devices come with a choice of connectivity solutions, WIFI, Bluetooth, and 3G, enabling the devices to export sensor data to be processed and saved in a cloud environment. Wearable devices paved the way for advanced products and services for human activity recognition in the healthcare and sports domains. Human activity recognition has attracted substantial attention both from the academic and commercial world. Smart watches and smart bands currently available in the retail market have certain capabilities for activity recognition of individuals exercising certain sports activities. The inertial sensors embedded in those smart devices help classify human activities by applying certain machine learning algorithms. Accelerometers can be used to identify human activities requiring high-energy actions like walking, jogging, jumping, etc. This way individuals get feedback about their performance during their fitness and sports activities without the need for a third eye to evaluate their performance. Besides, wearable devices can be easily deployed on the human body without posing any obstruction problems to detect movements contrary to the case of video-based applications (Lara & Labrador, 2013).

There has been considerable interest in time-series forecasting by academia. Communication, energy, financial, industrial, health, sports, and business applications can be counted among the fields where time-series forecasting is applied to forecast time-series signals (Baarnhielm, 2017; Denton, Frank and Feaver, Christine and Spencer, 2005; Hu et al., 2015; R. Krishnan et al., 2020; Li et al., 2020; Mishra et al., 2020; Sezer et al., 2020; Vengertsev, 2014). Recent advancements in machine learning research came up with innovative machine learning models to tackle time-series forecasting problems. Machine learning models for time series require a different approach than other feature-ready supervised models. Many machine learning models

require feature vectors to be identified and to be used in model training. On the other hand, in some machine learning applications, features don't exist readily as in the case of time-series data-driven models. Feature extraction and feature learning is an essential step in time series unsupervised, supervised, and semi-supervised machine learning applications (Lei & Wu, 2020; Liu & Fulcher, 2018). Many time and frequency domain feature extraction techniques help identify salient features to be used in model training. In addition to time and frequency domain extraction techniques, as an unsupervised learning method, statistical feature learning methods have attracted the attention of researchers. To help feature extraction, energy-based unsupervised models have gained substantial ground in time-series forecasting applications (Jakšić et al., 2020; Kuremoto et al., 2014; Längkvist et al., 2014; Osogami, 2017; Raymond et al., 2017; Taylor & Hinton, 2009).

1.1 Motivations

Human activity recognition (HAR) is a rapidly expanding field with a variety of applications in healthcare, sports, home monitoring and supported living, daily sports activities, martial arts, and performance sports, to name a few (Avci et al., 2010). Wearable sensors are placed on the human body to aid in the recognition of human actions (N. C. Krishnan et al., 2009). Wearable technologies have been used by researchers to recognize and categorize human activities (Lara et al., 2012; Lara & Labrador, 2013). Many wearable devices come equipped with embedded sensors like accelerometers, gyroscopes, and magnetometers. Such wearable devices placed on the human body aid in recognizing human activities (Mannini & Sabatini, 2010). The goal of activity recognition is to categorize human movement or gestures into predetermined activity sets namely walking, leaping, running, standing, lying, and climbing or descending steps. Activity recognition entails classification, which necessitates feature learning, or the discovery of meaningful data representations (Thomas Plötz, 2011). The accuracy of the classification model is mostly determined by the features that are the most discriminative (Mitchell et al., 2013). Sensor readings representing human activity are structured as univariate time-series data. To train the model, time, frequency, and discrete domain feature extraction techniques are applied to discover the most salient features from the time-series data (Figo et al., 2010). In

addition to such feature extraction techniques, generative models can be employed to find a meaningful representation of sequential data (Nie et al., 2015).

Generative Adversarial Networks (GAN) are generative models that employ deep learning techniques and exploit information theory to generate samples. GANs consist of generator and discriminator networks. The generator network tries to fake the discriminator and the Discriminator network tries to distinguish if the sample comes from a real or fake distribution. As an unsupervised learning model, GANs are used to draw samples from an unknown probability distribution. Wasserstein GAN (WGAN), unlike GAN, minimizes an estimate of the Earth-distance Mover's (EM) rather than the Jensen-Shannon divergence. WGAN leads to more stable training than GAN.

Energy-based probabilistic graphical models can aid in feature learning for time-series data. Restricted Boltzmann Machine (RBM) is an energy-based statistical model which factorizes a random value probability distribution over a binary valued probability distribution. As a more complex variant of RBM, Conditional Restricted Boltzmann Machine (CRBM) can embody the temporal information in time-series data.

Wearable devices placed on the human body equipped with sensors like a 3-axis accelerometer and gyroscope produce signals representing the corresponding human activity in time-series data format. Deep learning models require features to be determined in model training. Feature extraction is an important step in such deep learning models. Time, frequency, and discrete domain feature extraction techniques should be applied to the signal to discover the features to be employed in model training for classification. CRBM is capable of capturing temporal information in time-series data and can be used for feature learning from the signal itself. CRBM visible layer represents the real-valued random variable probability distribution and CRBM hidden layer represents the data in the binary valued probability distribution. The ability of feature learning from the time-series data eliminates the need for any time, frequency, and discrete domain feature extraction steps.

1.2 Objectives and Scope

This thesis study shows a comparison of different generative and predictive models' performances in generating time-series signals obtained from wearable devices. It is

shown how CRBM, an energy-based probabilistic graphical model, is capable of capturing temporal information and generating human activity sensor signals the closest to true signals representing human activities. The capability of CRBM to generate a signal closest to the true signal also confirms that feature learning from the signal itself is achieved and CRBM's hidden layer output represents the signal features in the binary domain. For this aim, generative and predictive models, GAN, WGAN-GP, RBM, CRBM, and LSTM have been tested for performance comparison. Similarity metrics, Euclidean, Canberra, and Dynamic Time Warping (DTW) distances, have been used as performance criteria to compare the performance of the models in generating signals. In the experiments, only the accelerometer x-axis body acceleration sensor signal has been used for the model's performance criterion in signal generation.

1.3 Contributions

This study aims to show a comparison framework for the selected generative and predictive models' performances in generating signals. CRBM is a powerful model for capturing temporal dependency information embedded in time-series signals. The ability of the CRBM model in generating the closest signal to the original signal also implies that real valued signals can be represented as binary valued features on the hidden layer of the CRBM model. Such an ability eliminates the need for feature extraction and feature selection techniques in time-series signals. A similarity metric framework is also presented to compare the performance of the generative and predictive models in generating time-series signals. The following list presents the contributions of this study.

- A benchmarking methodology is proposed for comparing and evaluating the performances of GAN, WGANGP, RBM, LSTM, and CRBM generative and predictive models in generating human activity sensor signals.
- It is demonstrated that the CRBM model outperforms the RBM, GAN, and WGAN-GP generative models in terms of generating samples that are the most similar to true human activity signal data. LSTM shows similar performance as CRBM.

- The capacity of the CRBM model in generating the closest samples to true signal also means that on its hidden unit layer, CRBM can capture the best representation of the signal in the binary domain. As a result, the CRBM hidden unit layer can be used to represent signal features in the binary domain, obviating the necessity for time and frequency domain feature extraction.

•

2. LITERATURE OVERVIEW

Human activity recognition became a research field in which researchers have tried to identify and classify human activities by bodily worn sensors called wearable devices. The corporal movement is defined as actions generated by muscles (Caspersen CJ, Powell KE, 1985). There are two methods proposed for activity recognitions. Researchers have used vision sensors for activity recognition (D.M. Gavrila, 1999; Pentland, 2000). Vision sensors raise the problem of intrusiveness and disruptiveness and violate the privacy of the users (Boyle et al., 2000; Hong et al., 2009). Inertial sensor systems, have emerged as a low-cost alternative solution eliminating the privacy concerns of individuals (N. C. Krishnan et al., 2009). An Inertial Measurement Unit consists of an accelerometer and a gyroscope.

Identification and classification of human activity from raw sensor signals involves multi-step processing and computation: pre-processing, segmentation, feature extraction, and classification (Osogami, 2017). Raw signal data is obtained from sensor units like gyroscopes and accelerometers. Pre-processing step involves applying filters to raw signal data. The signal readings are separated into fixed or adjustable length subintervals during the partitioning process. In the feature extraction step, a meaningful representation of the segmented signal data is obtained through some statistical or structural techniques. And finally, the last step involves the classification of each segmented part of the signal into a predefined activity set.

2.1 Applications

Technological advancement in the semiconductor field made it possible to produce quite smaller sensors to be used by humans in many applications such as medical applications, home monitoring and assisted living, and sports and leisure applications (Avci et al., 2010).

2.1.1 Medical applications

Monitoring and diagnosis, rehabilitation, the correlation between movement and emotions, and child and elderly care can be counted among the medical application where the sensors are used (Avci et al., 2010).

2.1.2 Home monitoring and assisted living

Sensors have found applications in home environments to provide supervision and aiding in the residents to provide well-being, health, and safety. Assisted living systems provide services such as tracking, fall detection, and security (Hou et al., 2007). Tracking, monitoring, emergency help, assistance for people with cognitive disorders, and assistance for people with chronic conditions are the categories where the use of sensors finds application in home monitoring and assisted living (Avci et al., 2010).

2.1.3 Sports and leisure applications

In sports, sensors are used for sports activity recognition as well as to monitor and track sports people (Avci et al., 2010). Cycling, football, and running are among the daily sports activities. Activity recognition in this field has been worked on by many researchers. Some researchers worked on supervised and unsupervised models for activity recognition (Avci et al., 2010; Ermes et al., 2008).

2.1.4 Exergaming

Exergaming involves incorporating human activity, exercise, and health information into video games. In exergaming applications, wearable sensors aid in activity recognition (Alshurafa et al., 2014; Ishii et al., 2020; Johnson et al., 2019; Mortazavi et al., 2016). Body-worn sensors are used to detect motion sequences in martial arts. Gyroscopes and accelerometers aided in increasing the interaction in martial arts video games (Heinz et al., 2007).

2.1.5 Performance sports

Performance sports activities are those which require high-energy activities. Sports activities like running, jogging, playing football, and basketball are among such performance activities. Some commercial brands developed special purpose sensors to

be used in such activities. Nike has rolled out a sensor called Nike+ to be placed in the shoe to track sports activities while keeping the training history (O.w.f. Nike+, 2009).

2.2 Activity Recognition Process

Identification and classification of sports activities from raw sensor signals involve multi-step complex processing and computation. These steps can be listed as follows: preprocessing, segmentation, feature extraction, dimensionality reduction, and classification (Avci et al., 2010; N. C. Krishnan et al., 2009). Figure 2.1 shows a flow of steps involved in activity recognition.

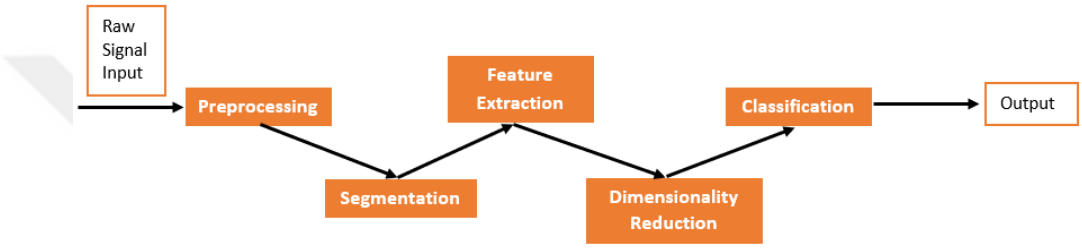

Figure 2.1 : Steps involved in activity recognition.

2.2.1 Preprocessing and signal representation

Raw sensor signals need to be pre-processed to eliminate noises. Non-linear low passed median, Laplacian and Gaussian filters can be applied to remove high-frequency noise (Bidargaddi et al., 2007; Karantonis et al., 2006; N. C. Krishnan et al., 2008). Sliding window median filter is used to remove the noise and a Kalman filter to produce more prominent patterns in the data (Sajjad Hossain et al., 2017). To extract human activity-related acceleration from the accelerometer signal, gravitational acceleration should be removed from the accelerometer signal. A high-pass filter can be used to eliminate gravitational acceleration (Yang et al., 2008).

2.2.2 Piecewise linear representation (PLR)

Approximation of time-series T of length n, with K straight lines is a technique used in the literature (Keogh et al., 2001).

2.2.3 Fourier transforms (FTs)

Fourier transforms are used for frequency analysis of signals. FT preserves the main frequency which provides retaining the primary information of the signal. FT also

enables dimensionality reduction of the data (He et al., 2008). Channel-wise Fourier analysis is performed on the raw signal data of an analysis frame (Thomas Plötz, 2011).

2.2.4 Wavelet transforms (WTs)

In the case of a signal containing discontinuities and sharp spikes, FT may not result in good signal processing (Ayrulu-Erdem & Barshan, 2011; Charle K Chui, 1992). Discrete Wavelet Transform is used in discrete-time sampled signals. In activity recognition applications, DWT are widely used (Kunze et al., 2005; Song & Wang, 2005).

2.2.5 Hilbert-Huang transform (HHT)

Hilbert-Huang Transform is a better signal processing technique for handling those univariate time series having nonlinearity and non-stationary characteristics. Huang proposed a novel signal analysis theory called HHT in the 1990s (Huang et al., 1998). Xu and colleagues proposed an HHT model to extract multi-features from raw signals (Xu et al., 2016).

2.2.6 Segmentation

A sensor signal in activity recognition represents the activities performed in a certain period in the form of a discrete signal sampled at a certain frequency. We cannot directly apply activity recognition to whole signal data. Instead, we need to slice the signal data into chunks to identify the activities.

Extracting meaningful representation of raw signal data is key to higher accuracy classification. Keogh et al., present a method to approximate a time-series T of length n, with K straight lines as a Piecewise Linear Representation technique (Keogh et al., 2001).

2.2.7 Activity windowing

There are many methods for splitting the sensor signals for activities. Some methods for signal windowing are Activity Windowing (AW), Fixed Sample Windowing (FSW), Time Weighted Windowing (TWW) Mutual Information Windowing (MI), Dynamic Windowing (DW), and Fixed Time Windowing (FTW). Fixed Time Windowing is a common approach in literature and is used interchangeably with the

name Fixed Size Sliding Window. In Sliding Windows technique, a fixed-length of data frame is used to extract the features from the raw signal (Ahmadi et al., 2014; Ayrulu-Erdem & Barshan, 2011; Mitchell et al., 2013; Roberto Shinmoto Torres, Damith Ranasinghe, 2014; Sajjad Hossain et al., 2017). This fixed-length frame is slid over the time axis to identify the next activity with a certain percentage of overlapping data frames. In the case where no vicinal spaces overlay each other then it is called Fixed-Sized Non-Overlapping Sliding Window (FNSW) (Janidarmian et al., 2017).

2.2.8 Top-Down

The top-down algorithm has been studied in many fields like cartography, image processing, etc. In machine learning, the top-down algorithm was introduced by Duda and Harts (Clarke et al., 1974). In the top-down segmentation approach, the assumption is that when a user changes the activity, there should be a significant change in sensor signals (Wu et al., 2012).

2.2.9 Bottom-Up

Starting from the finest possible approximation, segments are merged until some stopping criteria are met (Keogh et al., 2001).

2.2.10 Sliding window and bottom-up (SWAB)

SWAB is a mixture of Bottom-Up and Sliding Window techniques. A buffer size of w which is capable of segmenting 5 or 6 segments is stored. Each time the leftmost segment is removed a new segment is added to the buffer by using the sliding window method (Keogh et al., 2004).

2.2.11 Linear interpolation

Each segment in the time series is presented by a line connecting two sampling points (Keogh et al., 2001, 2004).

2.2.12 Linear regression

Approximating line for each segment is taken to be the best fitting line in the least square sense (Keogh et al., 2001, 2004).

2.3 Feature Selection, Feature Extraction, and Classification

Feature selection helps detect and discard those features having no discriminative property on classification. Feature selection is implemented by a sub-optimal search algorithm like the sequential forward-backward selection (SFS, SBS), the Pudil algorithm based on a Sequential Floating Forward-Backward Search (SFFS-SFBS) (Mannini & Sabatini, 2010; Olszewski et al., 2001). The greedy forward feature selection method is a technique used for time-series classification (Fulcher & Jones, 2014). In general, there are two approaches for feature extraction, statistical and structural. Statistical pattern recognition involves discrimination based on the quantitative feature of the data whereas structural-based feature extraction involves the interrelationship among the data. Mean, median, standard deviation, entropy, correlation coefficients, FFT (Fast Fourier Transform), DWT (Discrete Wavelet Transform), and HHT (Hilbert-Huang Transform) are among the techniques used for statistical feature extraction (Ayrulu-Erdem & Barshan, 2011; Figo et al., 2010; Olszewski et al., 2001). Time and frequency domain feature extraction from 3D accelerometer data is another feature extraction technique used in the literature (Reiss & Stricker, 2011). DWT is used in discrete-time sampled signals. In activity recognition applications, DWT is used to extract discriminative features from the signal. DWT decomposes a signal into a group of discrete wavelet coefficients and correlates the signal with a mother wavelet function. The choice of the length of time, called window length, to separate activities has a direct effect on the classification performance. The classification accuracy may depend on the selection of different mother wavelet functions, DWT level and window length (Mitchell et al., 2013). It is shown that the accuracy of the leg motion classification from gyroscope signal is very dependent on the choice of mother wavelet and high accuracy can be achieved by performing experimental work (Ayrulu-Erdem & Barshan, 2011). The power of detail signals is used for the given levels obtained through DWT of the windowed signal (Kunze et al., 2005). DWT is used to extract features from the windowed signal by using five-level signal decomposition and low-frequency components (Song & Wang, 2005). Discriminative features are extracted from accelerometer data by applying DWT with Daubechies 4 wavelet “db4” (Ahmadi et al., 2014). HHT is a better signal processing technique for dealing with signals having nonlinearity and non-stationary properties. In the 1990s, Huang presented a new signal analysis theory titled

HHT.(Huang et al., 1998). Xu and colleagues proposed an HHT model to extract multi-features from raw signals (Xu et al., 2016).

Some researchers applied PCA (Principal Component Analysis) to learn automatically the features from the training dataset (He & Jin, 2009). PCA can be applied to raw data to model a linear transformation of raw features to a lower dimensional subspace. PCA is not reliable for modeling complex non-linear transformations of raw features. RBM has been also used both for feature detection and classification. A deep learning network with stacked RBMs has been trained to detect features from sensor data (Thomas Plötz, 2011). Stacked RBMs are used to classify soccer player activities (Sajjad Hossain et al., 2017). Deep Neural Networks have been proposed to extract more meaningful features (Zeng et al., 2015). Deep Convolutional Neural Network proved to be one of the best feature extraction models for more discriminative features (Jiang & Yin, 2015). Figure 2.2 shows the preprocessing techniques for feature extraction used in time-series classification (Figo et al., 2010).

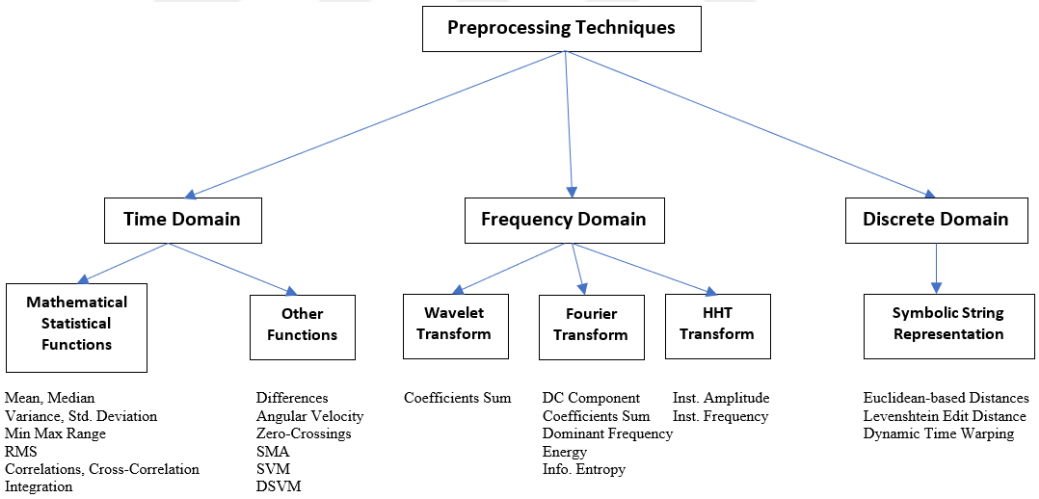

Figure 2.2 : Classification of techniques applied to signals for feature extraction.

2.4 Learning

Raw data provided by sensing devices should be fed into a machine learning tool to recognize activities corresponding to raw sensor signals. The human activity recognition (HAR) system uses such machine learning methods to describe, analyze and predict data. In HAR systems, obtaining and labeling the training data is a challenging problem. Assigning a label to each activity is not an easy task and requires

an expert to manually examine the samples and assign a label to the corresponding activity (Lara & Labrador, 2013; Stikic et al., 2009).

2.4.1 Supervised learning

The machine learning system is trained with the labels provided with the training dataset in supervised learning, and once the trained system is obtained, the new data is classified by applying it to the trained model. There are many methods for supervised learning. Decision Trees, Bayesian models, Similarity Based Learning, Support Vector Machines, Artificial Neural Networks, Deep Learning, and Ensemble of Classifiers are among the models to be used in training and classification in HAR (Lara & Labrador, 2013). Decision Trees are hierarchical models in which the local region is split recursively into nodes and edges (Alpaydin, 2014). Each branch from root to leaf is a classification rule (Lara & Labrador, 2013). Bayesian classifiers use Bayesian decision theory to calculate the posterior probability from class conditional probabilities and class distributions. In Similarity Based Learning, an instance is classified as per its similarity to other instances in the training data set. SVM uses kernel functions to find hyperplanes for class separation. In Neural Network based classification, a multilayer perceptron is used with non-linear activation function and biases to find hyperplanes for class separation. Artificial Neural Networks are universal function approximators. Deep Learning is a model based on neural networks in which a multilayer structure contains more layers and/or perceptron. Ensemble of Classifiers combines the output of different classifiers to improve overall classification accuracy. Bagging, boosting, and stacking are examples of such classifiers. Table 2.1 shows the classifiers used in HAR systems (Lara & Labrador, 2013).

Table 2.1 : The classifiers used by different HAR systems.

Type	Classifier	Reference
Decision tree	C4.5, ID3, Random Forest	(Ahmadi et al., 2014), (Reiss & Stricker, 2011), (L. Liu et al., 2016), (Tapia et al., 2007), (Jatobá et al., 2008), (Maurer et al., 2006)
Bayesian	Naive Bayes/Bayesian Networks	(Mitchell et al., 2013), (Tapia et al., 2007), (Jatobá et al., 2008)
Similarity Based	k-nearest neighbors	(Mitchell et al., 2013), (Xu et al., 2016), (Janidarmian et al., 2017), (Reiss & Stricker, 2011), (Jatobá et al., 2008), (Maurer et al., 2006), (Vollmer et al., 2013)
Deep Learning	Convolutional NN, Recurrent NN, RBM, Multilayer Perceptron	(Mitchell et al., 2013), (Sajjad Hossain et al., 2017), (Ayrulu-Erdem & Barshan, 2011), (Xu et al., 2016), (Zeng et al., 2015), (Jiang & Yin, 2015), (Singh et al., 2017), (Randell & Muller, 2000), (Iervolino et al., 2017), (J. B. Yang et al., 2015)
Domain transform	SVM	(Mitchell et al., 2013), (Janidarmian et al., 2017), (D. Wang et al., 2018)
Fuzzy Logic	Fuzzy Basis Function Fuzzy Inference System	(Chen et al., 2008)
Regression methods	Logistic Regression	(L. Liu et al., 2016)
Markov models	Hidden Markov Models Conditional Random Fields	(Wu et al., 2012), (Mannini & Sabatini, 2010)
Classifier ensembles	Boosting and Bagging	(Lara et al., 2012), (Janidarmian et al., 2017), (Minnen et al., 2007)

2.4.2 HAR architecture

Figure 2.3 shows the steps involved in a human activity recognition system. A sensor placed on the human body will provide IMU sensor signals to help recognize the activities.

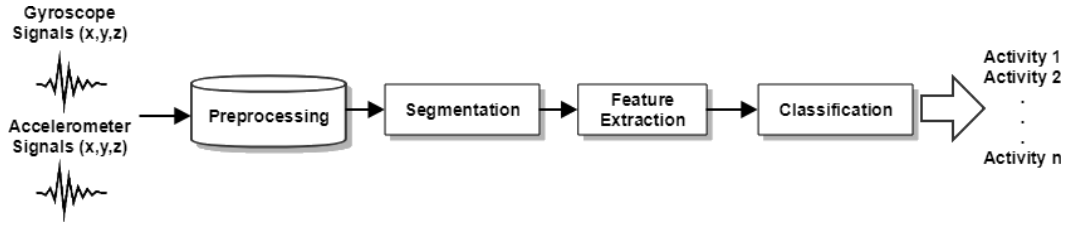

Figure 2.3 : Visual representation of Activity Recognition System.

3. PROPOSED APPROACH

A Restricted Boltzmann Machine is a graphical probabilistic parameterized generative model. An introduction and tutorial on Restricted Boltzmann Machine are given in (Fischer & Igel, 2014). Following is a summary of the tutorial.

3.1 Graphical Models

A graph is a representation of a data structure G consisting of a set of nodes, $X = \{X_1, \dots, X_n\}$ and a set of edges connecting the nodes. An edge is said to be a directed edge if there is one directional dependency between two edges $X_i \rightarrow X_j$. An edge is said to be an undirected edge if there is no directional dependency between the two edges $X_i - X_j$. Figure 3.1 shows an example of an undirected graph.

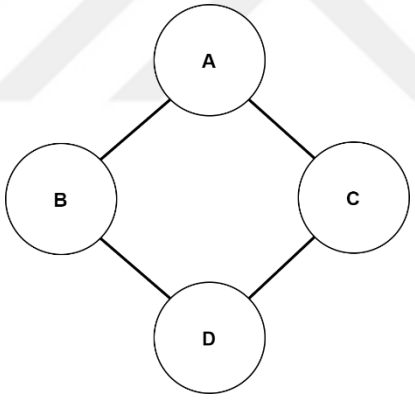

Figure 3.1 : An undirected graph.

Conditional dependence and independence between random variables can be represented by such a graph structure and it is called the Probabilistic Graphical Model. Factor graphs, Bayesian networks (directed graph), and Markov random fields (undirected graph) are examples of different graph structures (Fischer & Igel, 2014).

3.2 Undirected Graphs and Markov Random Fields

An undirected graph $\mathfrak{H}$ is represented by a finite set of nodes V, and undirected edges connecting the nodes, E. A node, *m* is said to be part of the neighborhood of node *n* if there exists an edge connecting the two nodes *m* and *n*. If some nodes in V are pairwise

connected, then those nodes construct a clique. The pairs {A,B}, {A,C}, {C,D} and {B,D} are examples of cliques. If adding a node doesn't change the status of the clique, then the clique is called a maximal clique. Let's denote by *C* the set of all maximal cliques in an undirected graph. The nodes in an undirected graph can be associated with random variable X. Then the random variables associated with the nodes are called Markov Random Field (MRF) if the joint probability distribution *p* satisfies the global Markov property. A strictly positive distribution p fulfills the Markov property for an undirected graph if and only if *p* factorizes over the $\mathfrak{H}$. A distribution is said to factorize about an undirected graph $\mathfrak{H}$ with maximal cliques c if there exists a set of non-negative functions $\{\psi_c\}_{c\subset C}$, called potential functions.

and

$$p(x) = \frac{1}{Z}\prod_{c\epsilon C}\psi_c(x) \tag{3.1}$$

where

$$Z = \sum\nolimits_x \prod\nolimits_{c\epsilon C}\psi_c(x_c) \tag{3.2}$$

the normalization constant, Z, is called the partition function.

$$p(x) = \frac{1}{Z}\prod\nolimits_{c\epsilon C}\psi_c(x_c) = \frac{1}{Z}e^{\sum_{c\epsilon C} ln\psi_c(x_c)} = \frac{1}{Z}e^{-E(x)} \tag{3.3}$$

where $E = -\sum_{c\epsilon C} ln\psi_c(x_c)$ is called the energy function (Fischer & Igel, 2014).

3.3 Maximum-likelihood Learning

We want to learn MRF of unknown distribution by using sample data, S. We assume that the graphical model is parameterized with θ and by unsupervised learning we adjust the parameters θ so that unknown distribution dependent on parameters, $p(x|\theta)$, is learned by the graphical model. Then the likelihood function can be written as,

$$\mathcal{L}(\theta|S) = \prod_{i=1}^{\ell} p(x_i|\theta) \tag{3.4}$$

And log-likelihood is given by (Fischer & Igel, 2014),

$$ln\mathcal{L}(\theta|S) = \ln \prod_{i=1}^{\ell} p(x_i|\theta) = \sum_{i=1}^{\ell} \ln p(x_i|\theta) \tag{3.5}$$

3.4 Gibbs Distribution and Latent Variables

In an MRF, the nodes are split into visible, $V = (V_1, \dots, V_m)$ and hidden (latent), $H = (H_1, \dots, H_n)$ nodes. Visible nodes represent the observable random variable and latent nodes help describe unknown observed complex distributions by conditional distributions. A distribution is said to be a Gibbs distribution if it can be parametrized by a set of factors. In MRF, The Gibbs distribution is the joint probability distribution of (V, H), and the marginal distribution of V is given by

$$p(v) = \sum_{h} p(v,h) = \frac{1}{Z} \sum_{h} e^{-E(v,h)} \tag{3.6}$$

where $Z = \sum_{v,h} e^{-E(v,h)}$ is a normalizing constant (Fischer & Igel, 2014).

3.5 Kullback-Liebler Divergence

By learning, we want that the distance between the distribution q under S and the distribution p of the Markov Random Field is minimized by maximizing the log-likelihood. This is called Kullback-Liebler Divergence and is expressed as (Fischer & Igel, 2014),

$$KL(q||p) = \sum_{x \epsilon \Omega} q(x) \ln \frac{q(x)}{p(x)} = \sum_{x \epsilon \Omega} \mathrm{q(x)} \ln q(x) - \sum_{x \epsilon \Omega} q(x) \ln p(x) \tag{3.7}$$

3.6 MRF Log-Likelihood Gradient

The learning rule for an MRF is based on the gradient ascent on the log-likelihood. The log-likelihood of the model given the parameters θ is given by,

$$ln\mathcal{L}(\theta|v) = ln\mathcal{L}(\mathrm{v}|\theta)$$
$$= \frac{1}{Z} \sum_{h} e^{-E(v,h)} = ln \sum_{h} e^{-E(v,h)} - ln \sum_{v,h} e^{-E(v,h)} \tag{3.8}$$

And the gradient ascent can be obtained by the partial derivative of the log-likelihood for the parameter θ.

$$\frac{\partial ln\mathcal{L}(\theta|v)}{\partial\theta} = \frac{\partial}{\partial\theta}\left(ln\sum_{h} e^{-E(v,h)}\right) - \frac{\partial}{\partial\theta}\left(ln\sum_{v,h} e^{-E(v,h)}\right)$$
$$= \frac{1}{\sum_h e^{-E(v,h)}}\sum_{h} e^{-E(v,h)}\frac{\partial E(v,h)}{\partial\theta} + \frac{1}{\sum_h e^{-E(v,h)}}\sum_{v,h} e^{-E(v,h)}\frac{\partial E(v,h)}{\partial\theta} \quad (3.9)$$

$$p(h|v) = \frac{p(v,h)}{p(v)} = \frac{\frac{1}{Z}e^{-E(v,h)}}{\frac{1}{Z}\sum_h e^{-E(v,h)}} = \frac{e^{-E(v,h)}}{\sum_h e^{-E(v,h)}} \quad (3.10)$$

By using the conditional probability equation above, the gradient ascent learning rule is written as follows.

$$\frac{\partial ln\mathcal{L}(\theta|v)}{\partial\partial} = -\sum_{h} p(h|v)\frac{\partial E(v,h)}{\partial\partial} + \sum_{v,h} p(v,h)\frac{\partial E(v,h)}{\partial\partial} \quad (3.11)$$

The above equation states that the gradient of the log-likelihood with respect to parameters θ is the difference between the expected value of the energy function under the model distribution and the expected value of the energy function under the conditional probability of the latent variable given the visible values. Calculating this equation with each probability value of both model and conditional distribution leads to complexity. To ease the calculation, the expectation values can be approximated by drawing samples from the corresponding probability distributions. Gibbs sampling is used to draw samples from a complex multivariate joint probability distribution and can be applied to this equation for approximated values of the expectations (Fischer & Igel, 2014).

3.7 Gibbs Sampling and Contrastive Divergence

The second term in equation 11 is intractable. Gibbs sampling can be used to obtain an approximation to the second term. A random sample v_r from the training data is used to start the Markov chain. The first sample vector of hidden variables is formed by sampling the first sample of each hidden variable independently. Following that, given the first sample of the hidden variable vector, the second sample of the observable variables is taken in the same way. This procedure loops back and forth

between the hidden and observed variables until the necessary number of samples K is reached (Yu, 2017). Figure 3.2 shows the Gibbs sampling for K steps in RBM.

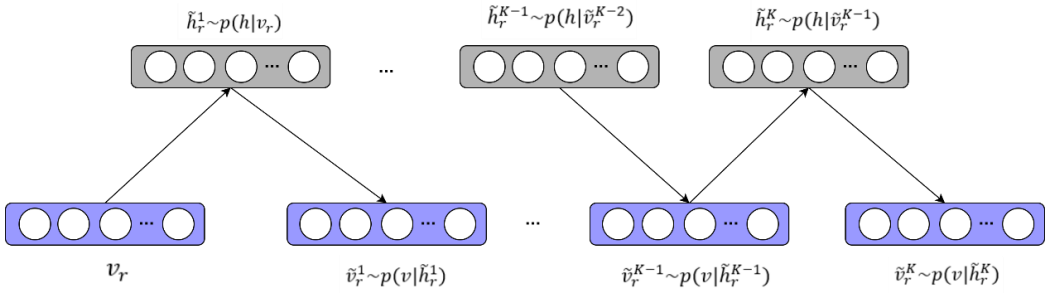

Figure 3.2 : K-step Gibbs Sampling in RBM**.**

3.8 Restricted Boltzmann Machine

Feature-based classification models require features to be well identified from the data and the model be trained with those features. When it comes to signals, feature identification in signals is not straightforward as in tabular data. Time and frequency domain feature extraction techniques are the most popular techniques to identify the features from signals. Although time and frequency domain techniques may work for most cases, such techniques have their drawbacks. The most significant drawback of using time domain feature extraction techniques stems from a non-stationary property of the signals, and alteration of signal statistical properties over time (N. C. Krishnan et al., 2008). When it comes to frequency-domain extraction techniques like FFT, the main disadvantage of using such a technique is the high computational costs (Yang et al., 2008). On the other hand, RBMs are statistical learning models which are capable of learning features from signals.

Figure 3.3 shows a typical RBM. RBM is an undirected graphical model based on the energy that factorizes a binary random variable's probability distribution over its hidden units. The idea behind RBM was first introduced in harmony theory in (Sajjad Hossain et al., 2017). RBM was later used in unsupervised learning to learn features on binary data (Yang et al., 2008). The RBM is also considered the product of experts (Keogh et al., 2001).

As shown in Figure 3.3, RBM is made up of two layers: a hidden layer (v) and a visible layer (h). The visible layer is used for the binary data inputs, and the hidden layer units represent the binary probability distribution of visible units. Both visible and hidden

layers represent variables in the binary domain. Both layers are connected fully to each other. Both visible and hidden units are not connected at the layer level.

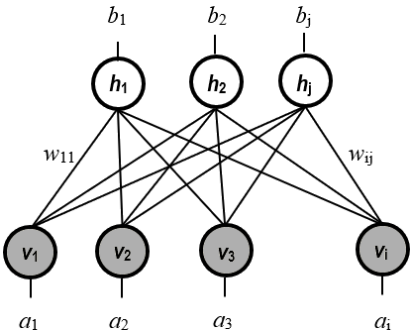

Figure 3.3 : Hidden and visible units of an RBM.

The RBMs are energy-based undirected probabilistic graphical models. The following is an energy function for a joint configuration (v, h) of hidden (h) and visible (v) units. (Hinton, 2012).

$$E(v;h) = -\sum_{i} a_i v_i - \sum_{j} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{3.12}$$

The binary states of $\boldsymbol{j}$th hidden unit and $\boldsymbol{i}$th visible unit are given by $\boldsymbol{h_j}$ and $\boldsymbol{v_i}$. $\boldsymbol{a_i}, \boldsymbol{b_j}$ are the biases of visible and hidden units, respectively, $\boldsymbol{w_{ij}}$ are the weights of the connections between hidden and visible units. The joint probability distribution of the whole configuration is given by

$$p(v,h) = \frac{1}{Z} e^{-E(v,h)} \tag{3.13}$$

where $\boldsymbol{Z}$ is the normalization constant or the partition function given by summing over all visible and hidden binary states as follows:

$$Z = \sum_{v,h} e^{-E(v,h)} \tag{3.14}$$

The marginal probability distribution of the visible vector, $\boldsymbol{v}$, is given by summing over all possible hidden vectors:

$$p(v) = \frac{1}{Z} \sum_{h} e^{-E(v,h)} \tag{3.15}$$

To assign a high probability to a visible vector, weights can be updated to assign lower energy (Hinton et al., 1984). The weights can be updated by taking a partial derivative

of the log probability of the visible vector with respect to the weights (Ackley et al., 1985),

$$\frac{\partial \log p(v)}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{3.16}$$

$\langle v_i h_j \rangle_{data}$ and $\langle v_i h_j \rangle_{model}$ denote the expectations of the distributions. And the weights can be updated as follows, where $\in$ denotes the learning rate.

$$\Delta w_{ij} = \in \left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \right) \tag{3.17}$$

The conditional probability that RBM sets the hidden unit binary state to 1, given the visible state is

$$p\left(h_j = 1 \middle| v\right) = \sigma\left(b_j + \sum_i v_i w_{ij} \right) \tag{3.18}$$

where $\boldsymbol{\sigma(x)}$ is the logistic sigmoid function.

And the conditional probability that RBM sets the visible unit binary state to 1, given hidden state is

$$p\left(v_j = 1 \middle| h\right) = \sigma\left(a_i + \sum_j h_j w_{ij} \right) \tag{3.19}$$

RBM visible layer units represent a binary valued random variable. We may have continuous-time data, as well. When visible nodes of an RBM represent a continuous random variable instead of a binary random variable, then the energy function is given by (Taylor et al., 2007),

$$E(v; h) = -\sum_i \frac{(v_i - c_i)^2}{2\sigma_i^2} - \sum_j b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij} \tag{3.20}$$

where $\boldsymbol{\sigma_i}$ is the standard deviation of Gaussian noise on visible units. The data is scaled to unit variance and a zero mean.

The conditional probability of hidden units can be easily derived given the states of the visible units. RBM conditional probability distributions, with unit variance $(\boldsymbol{\sigma_i = 1})$ and zero mean, can be given as follows:

$$p(h_j = 1|v) = f\left(b_j + \sum_i v_i w_{ij}\right) \tag{3.21}$$

$$p(v_i|h) = \mathcal{N}\left(c_i + \sum_j h_j w_{ij}, 1\right) \tag{3.22}$$

where $\boldsymbol{f}$ is the logistic function, $\boldsymbol{\mathcal{N}}$ is Gaussian, $\boldsymbol{b_j}$ is the bias of the jth of the hidden unit, $\boldsymbol{c_i}$ is the bias of the ith visible unit and $\boldsymbol{w_{ij}}$ are the weights of the undirected connection between the ith visible unit and the jth hidden unit.

By contrastive divergence, the learning rule for weights and hidden biases are given by

$$\Delta w_{ij}\ \alpha\ \langle v_i h_j\rangle_{data} - \langle v_i h_j\rangle_{model} \tag{3.23}$$

$$\Delta b_j\ \alpha\ \langle h_j\rangle_{data} - \langle h_j\rangle_{model} \tag{3.24}$$

where the first expectation is with respect to the data distribution and the second expectation is with respect to the distribution of reconstructed data. The reconstructions are obtained by Gibbs sampling (Fischer & Igel, 2014).

3.9 Conditional Restricted Boltzmann Machine

RBM does not take into consideration of the temporal information inherent in time series data. A different version of the RBM model was proposed which takes into account the temporal information from past data (Taylor et al., 2007; Taylor & Hinton, 2009). In the RBM model, the visible units represent only the static data while in CRBM, the visible unit layer is constructed not only with visible units but also with conditional units which hold the past data time segments. In RBM, the temporal information is not encoded in the model while in CRBM the temporal information is represented by the conditional units on the visible layer of the model. The model is encoded with this temporal information. The CRBM architecture is shown in Figure 3.4 (Mnih et al., 2011).

As seen in Figure 3.4, with the presence of delayed units, the CRBM model now consists of two more weight groups to be learned. Those two weights represent the connection weights, $\boldsymbol{w_{uv}}$, from delayed units to current visible units and connection weights, $\boldsymbol{w_{uh}}$, from delayed units to hidden units [79].

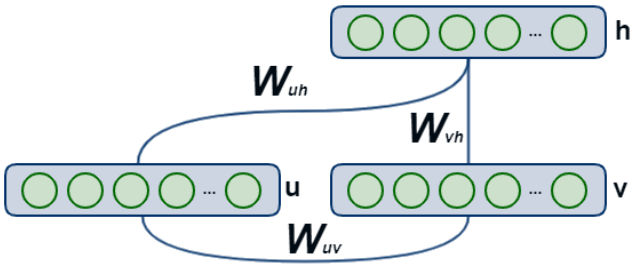

Figure 3.4 : A CRBM with visible (v), hidden (h) and delay (u) units.

The update rule for w_{uh} is given by:

$$\Delta(w_{uh})_{ij} \alpha\, u_i\left(\langle h_j \rangle_{data} - \langle h_j \rangle_{model}\right) \tag{3.25}$$

where w_{uh} is the weight connecting time-delayed visible unit i to hidden unit j.

The update rule for w_{uv} is then given by:

$$\Delta(w_{uv})_{ki} \alpha\, u_k(v_i - \langle v_i \rangle_{model}) \tag{3.26}$$

where w_{uv} is the weight from time-delayed visible unit k to visible unit i.

4. DATASET

Body-worn sensor signal data was obtained from the UCI repository. This dataset was obtained from 30 volunteers aged between 19-48 years old. The volunteers were asked to perform six different activities three of which are static postures (lying, standing, sitting) and the other are dynamic activities (walking upstairs, walking, walking downstairs). The postural transition data is also included in the dataset. The postural transition data represent the transitions between the activities: standing and sitting, sitting and lying, standing and lying. During human activity sensor recording experiments, a smartphone, Samsung Galaxy S II, was used to capture and record sensor signals. Accelerometer and gyroscope sensors embedded in the smartphone helped to capture 3-axial linear acceleration and 3-axial angular velocity signals. The signals were recorded with a constant rate of 50 Hz and to manually label the data, the experiments were video recorded.

A preprocessing step was applied to the accelerometer and gyroscope signals to remove the noise. After this preprocessing step, the signals were sampled in fixed width sliding window of 2.56 sec and 50% overlap. This corresponds to 128 readings per window. To separate body acceleration from the gravitational acceleration component, a Butterworth low-pass filter was applied to accelerometer signals. A filter with a cutoff frequency of 0.3 Hz has been applied to accelerometer signals to filter out the low frequency of the gravitational force component (Reyes-Ortiz et al., 2016).

Figure 4.1 shows the replacement of a smartphone in the waist bag of participants. The smartphone was used to record accelerometer and gyroscope signals corresponding to human activities.

Figure 4.1 : Smartphone placement into the waist bag.

Figure 4.2 shows the pictures taken during the six activity recordings. A video is available on YouTube showing the whole recording process (URL-1).

Figure 4.2 : Sensor signal recordings from left to right: Standing, Sitting, Lying, Walking, Walking Downstairs, and Walking Upstairs.

The visualization of the training dataset is shown in Figure 4.3. The smartphone Samsung Galaxy S II with embedded accelerometer and gyroscope sensors was used for recording the six human activities. Since the acceleration signal included both gravitational and body acceleration components, the body acceleration data was obtained by filtering out low-frequency components, namely the gravitational force.

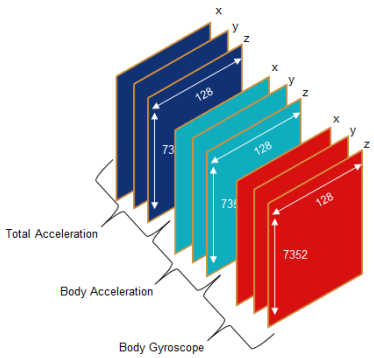

Figure 4.3 : Training dataset visualization.

Accelerometer and gyroscope signals were scaled to fit in the [-1,+1] region. Accelerometer signals from four different window frames for walking upstairs and walking downstairs activities are shown in Figure 4.4 and Figure 4.5, respectively.

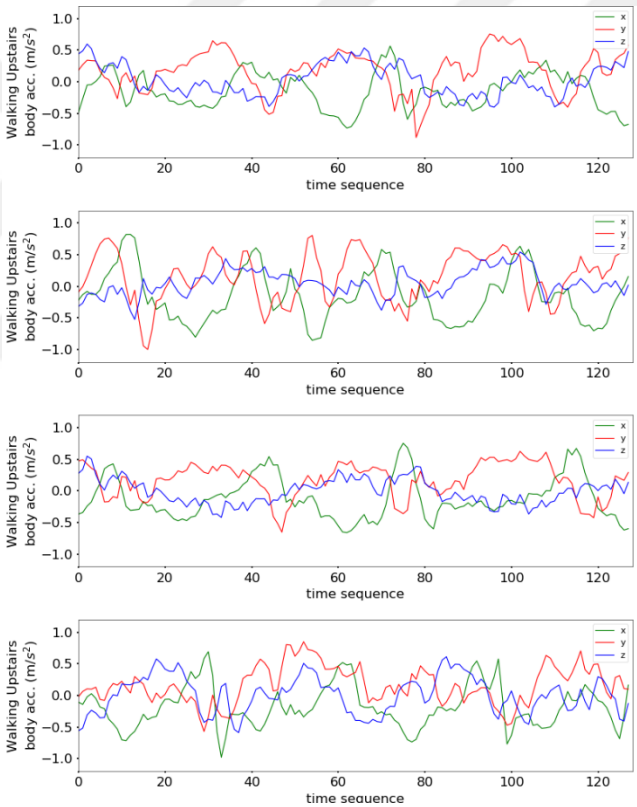

Figure 4.4 : x-y-z-axis body accelerometer signals while walking upstairs (data scaled into [-1,+1] region).

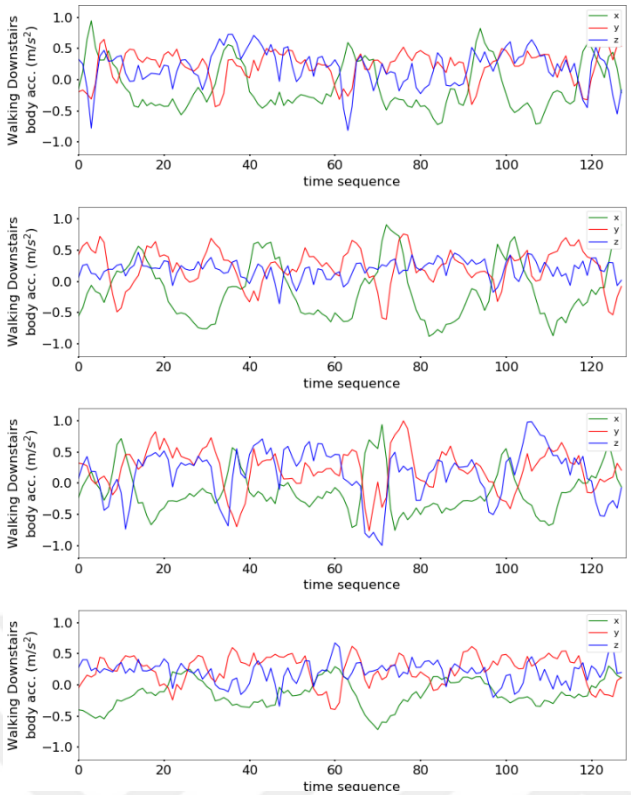

Figure 4.5 : x-y-z-axis body accelerometer signals while walking downstairs (data scaled into [-1,+1] region).

5. GENERATIVE MODELS

RBM is a generative model which consists of only visible and hidden layer units. Like RBM, CRBM is also a generative model, and in addition to visible and hidden units in RBM, CRBM also includes conditional units to encode the temporal information in the model. CRBM uses maximum likelihood explicit density approximation is trained almost the same way as RBM. The comparison and performance evaluation of synthetic signal generation of CRBM, RBM, GAN, and WGAN-GP are presented in Section 8.

5.1 Generative Adversarial Net (GAN)

GAN consists of Generator and Discriminator networks and both networks play a two-player minimax game with value function $V(D,G)$. Both the networks are constructed with vanilla multilayer perceptron. In this two-player minimax game, Generator tries to fake the Discriminator and Discriminator tries to identify if the data on its input comes from a real or fake distribution. Generator $G(z;\theta_g)$ multilayer perceptron with parameters θ_g is trained to generate a distribution p_g over data X and Discriminator $D(x;\theta_d)$ multilayer perceptron with parameters θ_d is trained to output a single scalar (I. J. Goodfellow et al., 2014). Detailed math behind how GAN and WGAN work to generate samples can be found at (Weng, 2019).

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right] \tag{5.1}$$

Where $p_r(x)$ and $p_z(z)$ indicate data distribution over real sample x, and data distribution over noise input respectively.

5.1.1 Binary cross-entropy loss function

In equation 28, $\hat{y}$ and y indicate the generated and real data respectively. The label for the data coming from $p_r(x)$ is $y = 1$ and $\hat{y} = D(x)$.

$$L(\hat{y}, y) = y log\hat{y} + (1 - y)\log(1 - \hat{y}) \tag{5.2}$$

And for the data coming from generator G(z) the label is y = 0 and $\hat{y} = D(G(z))$.

$$(D(x),1) = \log(D(x)) \tag{5.3}$$

$$L(D(G(z)),0) = (1 - 0)\log\Big(1 - D\big(G(z)\big)\Big) = \log(1 - D(G(z))) \tag{5.4}$$

The objective of the discriminator is to correctly classify fake vs real data. For this, the left-hand side of equations (5.3) and (5.4) should be maximized for the discriminator. Hence, the discriminator objective is

$$\max\Big\{\log\big(D(x)\big) + \log\Big(1 - D\big(G(z)\big)\Big)\Big\} \tag{5.5}$$

To reach this maximization objective, the discriminator will try to make D(x)=1 and D(G(z))=0. On the other hand, the generator will try to minimize $\log\Big(1 - D\big(G(z)\big)\Big)$. For this, the generator's objective is

$$\min\{\log\Big(1 - D\big(G(z)\big)\Big)\}. \tag{5.6}$$

And overall binary cross-entropy loss function can be given as

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}\Big[\log\Big(1 - D\big(G(z)\big)\Big)\Big]. \tag{5.7}$$

The optimal value for D can be derived as below.

$$L(G, D) = \int \big(p_r(x)\log\big(D(x)\big) + p_g(x)\log\big(1 - D(x)\big)\big)\, dx \tag{5.8}$$

$$\tilde{x} = D(x), A = p_r(x), B = p_g(x) \tag{5.9}$$

$$f(\tilde{x}) = Alog(\tilde{x}) + Blog(1 - \tilde{x}) \tag{5.10}$$

$$\frac{df(\tilde{x})}{d\tilde{x}} = A\frac{1}{ln10}\frac{1}{\tilde{x}} - B\frac{1}{ln10}\frac{1}{1-\tilde{x}} = \frac{1}{ln10}\left(\frac{A}{\tilde{x}} - \frac{B}{1-\tilde{x}}\right) = \frac{1}{ln10}\frac{A - (A+B)\tilde{x}}{\tilde{x}(1-\tilde{x})} \tag{5.11}$$

$$\frac{df(\tilde{x})}{d\tilde{x}} = 0 \tag{5.12}$$

$$D^*(x) = \tilde{x}^* = \frac{A}{A+B} = \frac{p_r(x)}{p_r(x) + p_g(x)} \in [0,1] \tag{5.13}$$

When $p_r = p_g$, $D^*(x)$ becomes ½.

5.1.2 Global optimum

By using the optimal values of G and D, $p_r = p_g$ and $D^*(x) = 1/2$, the global optimum can be derived by the following equations.

$$L(G, D^*) = \int \left(p_r(x) \log\left(D^*(x)\right) + p_g(x) \log\left(1 - D^*(x)\right)\right) dx \tag{5.14}$$

$$L(G, D^*) = log\frac{1}{2} \int \left(p_r(x) + p_g(x)\right) dx \tag{5.15}$$

$$L(G, D^*) = log\frac{1}{2} 2 \int p_r(x) dx = -2log2 \tag{5.16}$$

5.1.3 Jensen-Shannon divergence

KL divergence presented in 3.5, is a measure of distance between two probability distributions q and p. KL divergence is asymmetric. When $p(x)$ is close to zero and $q(x)$ takes values significantly greater than zero, KL divergence may not achieve good results. JS divergence is another measure of distance between two probability distributions. JS divergence is symmetric and is given by the following equation.

$$D_{JS}(q||\mathrm{p}) = \frac{1}{2} D_{KL}(\mathrm{p}||\frac{\mathrm{p}+\mathrm{q}}{2}) + \frac{1}{2} D_{KL}(\mathrm{q}||\frac{\mathrm{p}+\mathrm{q}}{2}) \tag{5.17}$$

5.1.4 The loss function represents a JS divergence

The JS divergence between p_r and p_g can be obtained by the following equations.

$$D_{JS}(p_r||p_g) = \frac{1}{2} D_{KL}(p_r||\frac{p_r + p_g}{2}) + \frac{1}{2} D_{KL}(p_g||\frac{p_r + p_g}{2}) \tag{5.18}$$

$$\begin{aligned} D_{JS}(p_r||p_g) = {} & \frac{1}{2}\left(log2 + \int p_r(x) log \frac{p_r(x)}{p_r(x) + p_g(x)} dx\right) \\ & + \frac{1}{2}\left(log2 + \int p_g(x) log \frac{p_g(x)}{p_r(x) + p_g(x)} dx\right) \end{aligned} \tag{5.19}$$

$$D_{JS}(p_r||p_g) = \frac{1}{2}(log4 + L(G, D^*)) \tag{5.20}$$

$$L(G, D^*) = 2D_{JS}(p_r||p_g) - 2log2 \tag{5.21}$$

The loss function is minimized when the JS divergence between the real data distribution p_r and generated data distribution $p_g$disappears. Then the minimized loss function becomes

$$L(G^*, D^*) = -2log2. \tag{5.22}$$

Although GAN may be a promising model for synthetic data generation, it has its peculiarities and disadvantages. GAN training is difficult, and the training may never converge, and mode collapses are among the main issues of GAN. Both the Generator and the Discriminator networks are implemented with vanilla neural networks. The Generator tries to fake the Discriminator by generating fake samples to trick it. On the other hand, the Discriminator tries to decide if the data on its input is fake or real. Training of GAN may become quite tricky. Since GAN is a two-player minimax game, GAN training requires finding a Nash equilibrium in a zero-sum game. In addition to its disadvantages regarding its training, the GAN model cannot encode temporal information meaning, it cannot generate the next time frame by processing previous time frames. The following list enumerates some of the difficulties faced in GAN training (I. Goodfellow, 2017; Salimans et al., 2016).

- **Mode collapse:** the generated samples may fall into a specific mode.
- **Non-convergence:** Nash equilibrium may not be achieved, the model parameters fluctuate, unsettle, and never converge,
- **Diminished gradient:** vanishing generator gradient problem may occur when the discriminator performs very well.
- **Sensitivity:** Highly sensitive to the hyperparameter selections.

5.2 Wasserstein Generative Adversarial Net (WGAN)

Because of the disadvantages of GAN listed above, an alternative model, WGAN, was presented as an improved model for GAN. WGAN uses earth mover distance also known as Wasserstein distance to measure the distance between two data distributions:

real data distribution P_r and generative data distribution P_g. The Wasserstein distance is given by

$$W(\mathrm{P}_r, \mathrm{P}_g) = \inf_{\gamma \epsilon \Pi(\mathrm{P}_r, \mathrm{P}_g)} \mathrm{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{5.23}$$

where $\Pi(\mathrm{P}_r, \mathrm{P}_g)$ denotes the set of all joint distributions (x, y) whose marginals are P_r and P_g, respectively (Arjovsky Martin et al., 2016).

Wasserstein distance given by the above equation is not tractable. This equation can be simplified by using the Kantorovich-Rubinstein duality resulting in the following equation.

$$W(\mathrm{P}_r, \mathrm{P}_g) = \sup_{\|f\|_L \leq 1} \mathrm{E}_{x\sim\mathrm{P}_r}[f(x)] - \mathrm{E}_{x\sim\mathrm{P}_\theta}[f(x)] \tag{5.24}$$

where sup is the least upper bound and f is a 1-Lipschitz function satisfying the following constraint.

$$|f(x_1) - f(x_2)| \leq |x_1 - x_2| \tag{5.25}$$

WGAN-with Gradient Penalty denoted as WGAN-GP was presented as an improved model for WGAN. In WGAN, during the training of the discriminator (critic), the weights are clipped as per the Lipschitz constraint. Weight clipping may introduce difficulties for the proper training of the critic (Gulrajani et al., 2017). Unlike WGAN, WGAN-GP uses gradient-penalty instead of weight clipping to fulfill the Lipschitz constraint. In WGAN-GP the objective is given by,

$$L = \mathop{\mathbb{E}}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathop{\mathbb{E}}_{x\sim\mathbb{P}_r}[D(x)] + \lambda \mathop{\mathbb{E}}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x}) - 1\|_2)^2]. \tag{5.26}$$

The first two terms are the original critical loss, and the last term is the gradient penalty. $\hat{x}$ is sampled from $\tilde{x}$ and x with t uniformly sampled between 0 and 1 as indicated by the following equation.

$$\hat{x} = t\tilde{x} + (1 - t)x \text{ with } 0 \leq t \leq 1 \tag{5.27}$$

6. SIMILARITY METRICS

As a performance criterion for the generative models, some similarity metrics are presented to assess the performance of the generative models. The generated signal and the true signal were compared by using the following three similarity metrics.

6.1 L2 Norm (||.||2) and Canberra Distance

L_2 norm or the Euclidean Distance can be used to measure the distance between two time-series data. Given two time series X and ϒ on $\mathbb{R}^n$,

$$\mathrm{X} = [x_1\ x_2\ x_3\ \dots x_n] \tag{6.1}$$

$$\Upsilon = [y_1\ y_2\ y_3\ \dots y_n] \tag{6.2}$$

Euclidean and Canberra distances are given by the following equations, respectively (Li et al., 2016).

$$Q_E = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{6.3}$$

$$Q_{Can} = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|} \tag{6.4}$$

6.2 Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW) algorithm can help measuring the degree of resemblance between two time series data. DTW works by locally extending or compressing the data in order to make one time series data resemble the other. Once the stretching is performed on the time-series data, DTW distance between two time-series data is measured by summing the distances of the aligned elements. In the comparison tables, namely Table 8.7, Table 8.8, and Table 8.9, DTW distance is presented as a similarity metric. Figure 6.1 and Figure 6.2 show the alignment of two signals (Giorgino, 2009).

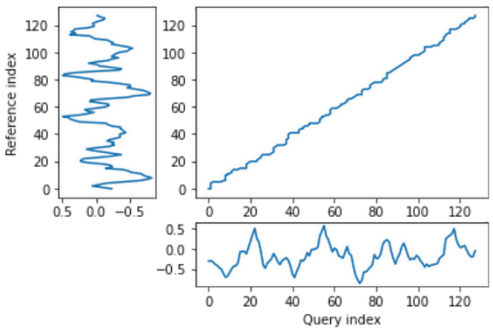

Figure 6.1 : Three-way plot of the asymmetric alignment between two signals.

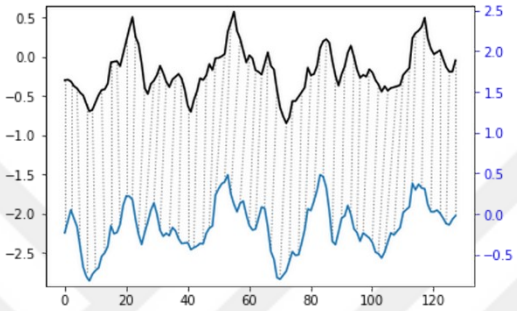

Figure 6.2 : Dynamic Time Warping Alignment between two signals.

6.3 R2 Score

R^2 is a measure of goodness-of-fit and is given by

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum(y - \hat{y})^2}{\sum(y - \bar{y})^2} \tag{6.5}$$

where y is the dependent variable, $\hat{y}$ is the prediction, and $\bar{y}$ is the average value of y.

RSS and TSS are Residual Sum of Squares and Total Sum of Squares respectively and they are given by,

$$RSS = \sum (y - \hat{y})^2 \tag{6.6}$$

$$TSS = \sum (y - \bar{y})^2. \tag{6.7}$$

In short, R^2 indicates how the fitted model explains the variance in the independent variable.

7. SYSTEM BLOCK DIAGRAM

Preprocessing, model training, and sample generation steps are shown below in Figure 7.1. The raw data is stored on the disk. The raw data is read from the disk and fed into the preprocessing block. Preprocessing step involves scaling the signals into the [-1,+1] range and splitting the data into training and test datasets. The models are trained with pre-processed training data. The signal generation process is executed once the model training is achieved.

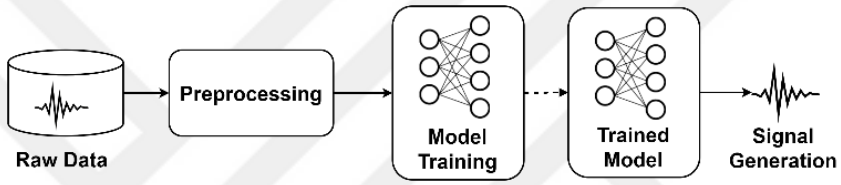

Figure 7.1 : Logical Block Diagram of Model Training and Signal Generation.

A logical block diagram of RBM model training is presented in Figure 7.2. As mentioned before, the raw signal data resides on the disk and before it is fed into the model for training, it is preprocessed to be scaled into the [-1,+1] region and to be split into test and training datasets.

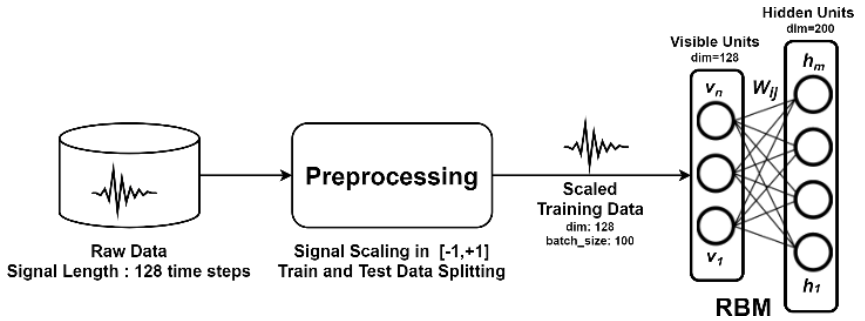

Figure 7.2 : Logical Block Diagram of RBM Model Training.

Training of CRBM is similar to RBM. Figure 7.3 shows the block diagram of the CRBM training process. Like RBM, raw signal data is preprocessed to be scaled into the [-1,+1] region and split into test and train datasets. Different from RBM, CRBM has conditional and visible units. Gaussian time-delayed unit and visible unit datasets are obtained by preprocessing the raw signal data. During the training, time-delayed

and visible unit data are shifted by one time-step forward at each iteration of the training epoch.

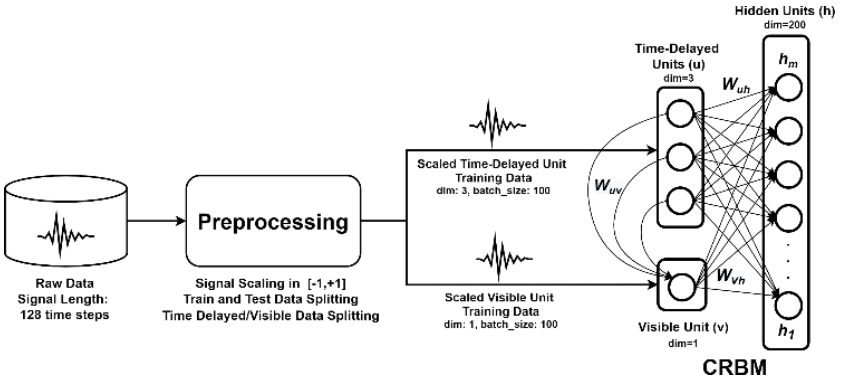

Figure 7.3 : Logical Block Diagram of CRBM Model Training.

Figure 7.4 and Figure 7.5 show the block diagrams of training of GAN and WGAN-GP models, respectively. Like in RBM and CRBM models, by preprocessing step, the raw signal data are scaled into [-1,+1] region and split into test and training data. In both GAN and WGAN-GP models, a normal distribution with zero mean and variance of one is used to sample from the latent space for both models. In the GAN model, the Generator's role is to generate samples to dupe the discriminator as if they come from the real data distribution. Unlike GAN, WGAN-GP uses a critic function instead of a sigmoid function on its output.

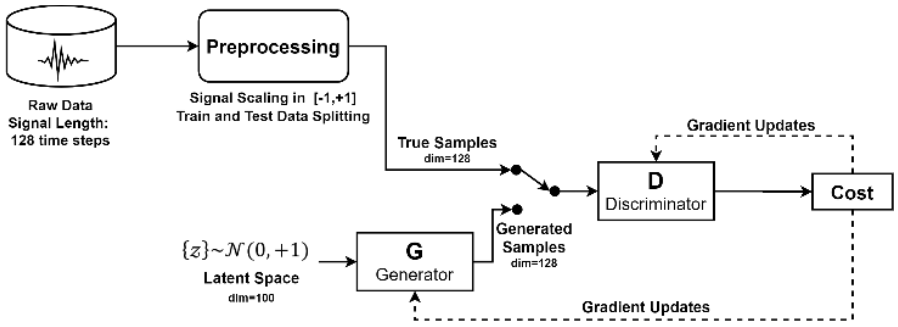

Figure 7.4 : Logical Block Diagram of GAN Model Training.

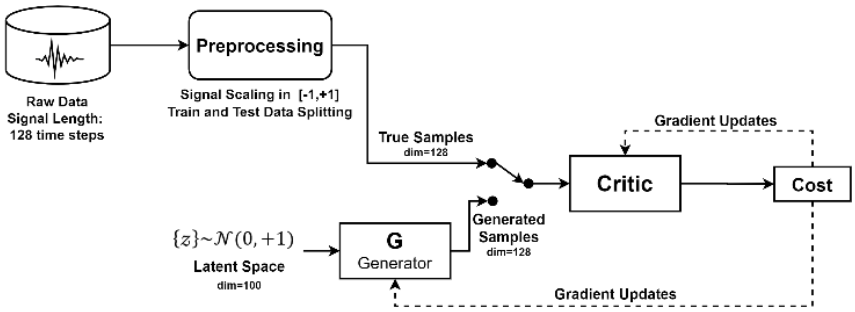

Figure 7.5 : Logical Block Diagram of WGAN-GP Model Training.

LSTM model training block diagram is given below in Figure 7.6. Likewise, raw data are scaled into [-1,+1] region and split into test and training datasets. LSTM model consists of 1 hidden layer of 3 units.

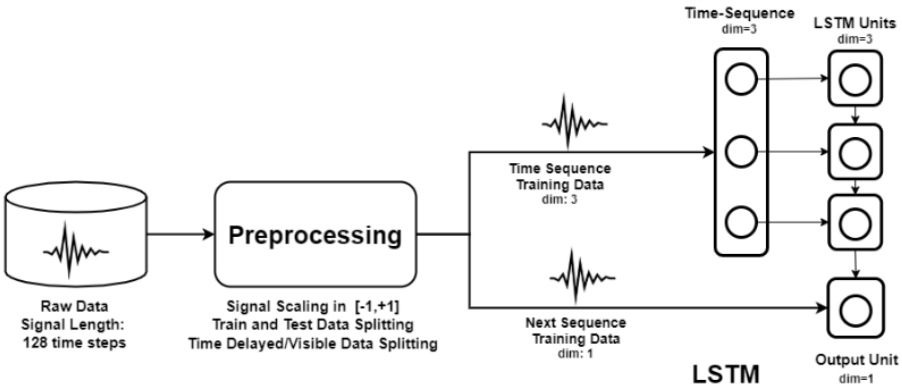

Figure 7.6 : Logical Block Diagram of LSTM Model Training**.**

8. EXPERIMENTS

8.1 Computation Platform

The aforementioned generative models were implemented by using Python programming language along with PyTorch and TensorFlow Python libraries. GAN and WGAN-GP generative model Python implementation code for training and generation were executed on the Google Colab backend cloud environment. Figure 8.1 shows a screenshot of the Google Colab Backend Engine used for running the Python codes. The GPU (Graphical Processing Unit) compute backend engine uses the Python 3 version with 25 GB of RAM.

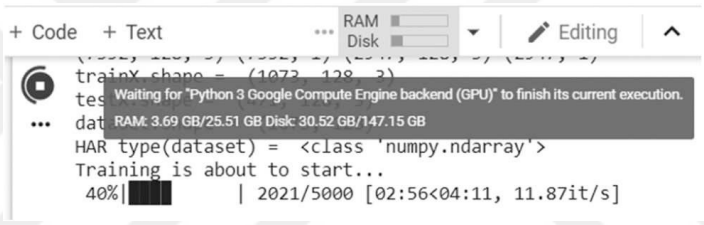

Figure 8.1 : Google Colab Python compute engine backend**.**

Coding of RBM and CRBM models training, and generation algorithms have been realized again with Python programming language. Unlike GAN and WGAN-GP, RBM and CRBM model implementation code for training and generation were executed on a laptop computer. The main reason behind this is that RBM and CRBM models required fewer resources than GAN and WGAN-GP models. Jupyter Notebook has been used as a Python coding environment to ease running the code. Figure 8.2 shows the screenshot of the Jupyter Notebook.

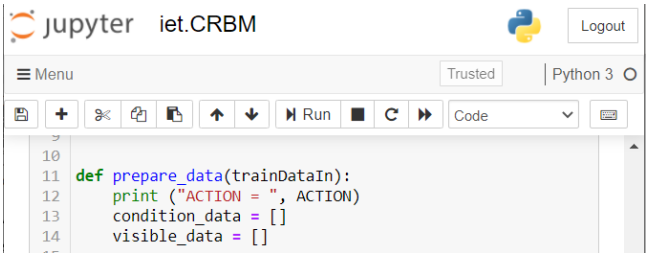

Figure 8.2 : Anaconda Jupyter Notebook.

Table 8.1 shows two computing platforms' capacities and features used for the experiments.

Table 8.1 : Computing platforms used.

	Computing Platforms	
	Laptop	**Google Colaboratory**
CPU/GPU	CPU Intel Core i7-7500	GPU NVIDIA-SMI 450.66 Driver v: 418.67 CUDA v: 10.1
RAM	16 GB	35 GB
Operating System	MS Windows 10 Pro	Ubuntu 18.04
Frameworks & Software Development Environment	Python v 3.7.6 Pytorch - v 1.5.1+cpu Pytorch - cuDNN v : None Tensorflow v 1.15.0 Jupyter Notebook Anaconda 3	Python v 3.6.9 Pytorch - v 1.6.0+cu101 Pytorch - cuDNN v 7603 Tensorflow v 2.3.0 Colab Jupyter Notebook

8.2 Experiment Setup

Among the aforementioned six activities only walking, walking upstairs, and walking downstairs activities were chosen to be used in the experiments. These activities' x-axis accelerometer readings were considered during the experiments. X-axis accelerometer signal readings were preprocessed to be scaled into the [−1, +1] region. Hyperparameters used during training of the four generative models GAN, WGAN-GP, RBM, CRBM, and predictive model LSTM are presented in Table 8.2, Table 8.3, Table 8.4, Table 8.5 and Table 8.6. Once the training is finished, a sample generation algorithm is executed to draw samples from the trained models. In the CRBM model, the dimension of the hidden and time-delayed units was chosen by trying different configurations and the configuration giving the best result in terms of signal generation performance was chosen and fixed for the rest of the experiment. WGAN-GP is an enhanced model of WGAN, and it uses Wasserstein distance to train and generate samples as if they come from the real data distribution. WGAN-GP uses conv1D and MaxPooling layers in its critic and generator networks. On the other hand, GAN uses a vanilla neural network with a smaller size of layers. Because of that, WGAN-GP is

a computationally expensive model than GAN. WGAN-GP training required more training epochs to converge when compared to GAN and WGAN-GP converge rates are longer than the GAN model. A comparison of model training times is presented in the discussion section.

Table 8.2 : RBM model hyperparameters.

RBM Model Parameters	Value
# of hidden units (h)	200
# of visible units (u)	128
Learning Rate	0.01
Training Epochs	100
Batch Size	100

Table 8.3 : CRBM model hyperparameters.

CRBM Model Parameters	Value
# of hidden units (h)	70
# of time-delayed units (u)	30
# of visible units (v)	1
Learning Rate	0.01
Training Epochs	30
Batch Size	100

Table 8.4 : LSTM model hyperparameters.

LSTM Model Parameters	Value
Input	3
# of hidden units	3
Learning Rate	0.01
Training Epochs	15
Batch Size	64

Table 8.5 : GAN model hyperparameters.

GAN Architecture Parameters		
Learning Rate	0.01	
Training Epochs	800	
Batch Size	100	
Generator		
Layer Type	**Parameters**	**Output Shape**
Input	Noise	(None,100)
Dense	128, LeakyReLU	(None,128)
Dense	256, LeakyReLU	(None,128)
Dense (Output)	128, tanh	(None,128)
Discriminator		
Layer Type	**Parameters**	**Output Shape**
Input	128	(None,128)
Dense	128, LeakyReLU	(None,128)
Dense	256, LeakyReLU	(None,128)
Dense (Output)	1, sigmoid	(None,1)

Table 8.6 : WGAN-GP model hyperparameters.

WGAN-GP Architecture Parameters		
Learning Rate	0.01	
Training Epochs	600	
Batch Size	100	
Generator		
Layer Type	**Parameters**	**Output Shape**
Input	Noise	(None,100)
Dense	100, LeakyReLU(0.2)	(None,100)
Conv1D	(100,32),LeakyReLU(0.2)	(None,100,32)
Conv1D	(200,32),LeakyReLU(0.2)	(None,200,32)
Conv1D	(400,32),LeakyReLU(0.2)	(None,400,32)
Conv1D	(800,32),LeakyReLU(0.2)	(None,800,1)
Dense (Output)	128, LeakyReLU(0.2)	(None,128)
Critic		
Layer Type	**Parameters**	**Output Shape**
Input	128	(None,128)
Conv1D	(100,32),LeakyReLU(0.2)	(None, 100,32)
MaxPooling	Pool Size = 2	(None, 50,32)
Conv1D	(50,32),LeakyReLU(0.2)	(None, 50,32)
MaxPooling	Pool Size = 2	(None, 25,32)
Conv1D	(25,32),LeakyReLU(0.2)	(None, 25,32)
Flatten		(None, 800)
Dense	50, LeakyReLU(0.2)	(None, 50)
Dense	15, LeakyReLU(0.2)	(None, 15)
Dense (Output)	1, LeakyReLU(0.2)	(None, 1)

8.3 The Shortest Euclidean Distance for GAN and WGAN-GP Models

Since GAN and WGAN-GP models try to generate signals in the same data distribution as real signals, we search for the generated signal in the generated sample set having the closest distance to the real sample drawn from the test dataset.

To assess models' signal generation capabilities, a distance measure on true and generated time-series can be defined to evaluate the models' signal generation capacities. Such a distance function to evaluate the performance of generative models is already given by (Meehan et al., 2020). The distance function below is used for the performance evaluation of the generative models.

Let $d: \mathcal{X} \rightarrow \mathbb{R}$ be a distance function from points in $\mathcal{X}$ drawn from generative set $\mathbb{Q}$ to a true sample $t \in \mathbb{T}$.

$$d(x) = argmin_{x \in \mathbb{Q}} \|x - t\|_2 \quad (8.1)$$

Then $d(x)$ is the shortest Euclidean distance between the generated sample $x \in \mathbb{Q}$ and true sample $t \in \mathbb{T}$.

The pseudocode in Figure 8.3 below shows the sampling algorithm for GAN and WGAN-GP models.

GAN/WGAN-GP: Sample Generation Algorithm

***n**: total sample size (value = 100)*
***m**: exhaustive search sample size (value=50000)*
***L2DistanceArr**: Array for storing Euclidean Distances*
***DTWDistanceArr**: Array for storing DTW Distances*
***CnbrDistanceArr**: Array for storing Canberra Distances*
***l2min**: The global min L_2 distance (initial value = 1000)*

for *i = 1, ...,n* **do**
$d \leftarrow$ Sample from test data $d \sim \mathcal{P}_t$
  Sample latent variable $\{z_{j,k=1}^{m,n}\} \sim \mathcal{N}(0,1)$
  GenSamplesArr ⟵ Sample from generator $\{g_{j=1}^{m}\} \sim G(z)$ and append to *GenSamplesArr*
  for *k = 1, ...,m* **do**
    $g \leftarrow$ *GenSamplesArr[k]*
    L2DistanceArr ⟵ L_2 Distance (g and d)
  end for
  $index_{l2min} \leftarrow$ *index(min(L2DistanceArr))*
  $l2minCurrent \leftarrow$ *L2DistanceArr*$[index_{l2min}]$
  $g_{l2min} \leftarrow$ *GenSamplesArr*$[index_{l2min}]$
  DTWDistanceArr ⟵DTWDistance (g_{l2min} and d)
  CnbrDistanceArr ⟵CanberraDistance (g_{l2min} and d)
  if ($l2minCurrent < l2min$) **do**
    $l2min = l2minCurrent$
    $g_{l2minglobal} \leftarrow g_{l2min}$
    $d_{l2minglobal} \leftarrow d$
  end if
end for
Compute statistics for L_2, DTW and Canberra Distances

Figure 8.3 : GAN/WGAN-GP: Sample Generation Algorithm.

8.4 Sample Generation for RBM and CRBM Models

Once trained, the generation process in the CRBM model involves iterating over conditional units and generating the next signal data point at the visible unit by applying one cycle of Gibbs sampling. Only conditional units are populated with data from the test dataset. Unlike GAN or WGAN-GP, the CRBM model generates only one signal at a time and the distance measure function is applied between the generated

signal and the true signal drawn from the test dataset. For RBM and CRBM, the following distance function is used to evaluate the performance.

Let $d: \mathcal{X} \rightarrow \mathbb{R}$ be a distance function from points in $\mathcal{X}$ drawn from generative set $\mathbb{Q}$ to the true sample $t \in \mathbb{T}$.

$$d(x) = \|x - t\|_2 \tag{8.2}$$

Then $d(x)$ is the Euclidean distance between the generated sample $x \in \mathbb{Q}$ and the true sample $t \in \mathbb{T}$. The algorithm in Figure 8.4 below shows the steps involved in generating samples from the CRBM model.

CRBM: Sample Generation Algorithm
# of time delayed units: 3, # of visible units: 1, # of hidden units: 200

n*: total sample size (value = 100)*
ConditionArr*: Array for storing Time Delayed Units*
VisibleArr*: Array for storing Visible Units Values*
L2DistanceArr*: Array for storing Euclidean Distances*
DTWDistanceArr*: Array for storing DTW Distances*
CnbrDistanceArr*: Array for storing Canberra Distances*
l2min*: The global min L2 distance (initial value = 1000)*

1: Compute *ConditionArr* and *VisibleArr* from Test Data
2: **for** *i = 1, ...,n* **do**
3: $\mathcal{d} \leftarrow$ Sample from test data $\mathcal{d} \sim \mathcal{P}_t$
4: $\mathcal{g} \leftarrow$ Sample CRBM with *ConditionArr & VisibleArr*
5: *L2DistanceArr* $\leftarrow L_2$ Distance ($\mathcal{g}$ and $\mathcal{d}$)
6: *DTWDistanceArr* $\leftarrow$DTWDistance ($\mathcal{g}$ and $\mathcal{d}$)
7: *CnbrDistanceArr* $\leftarrow$ *CanberraDistance($\mathcal{g}$ and $\mathcal{d}$)*
8: $index_{l2min} \leftarrow$ *index(min(L2DistanceArr))*
9: $l2minCurrent \leftarrow$ *L2DistanceArr*$[index_{l2min}]$
10: **if** (*l2minCurrent* < $l2min$) **do**
11: $l2min$ = *l2minCurrent*
12: $\mathcal{g}_{l2minglobal} \leftarrow \mathcal{g}$
13: $\mathcal{d}_{l2minglobal} \leftarrow \mathcal{d}$
14: **end if**
15: **end for**
16: Compute statistics for L_2, DTW and Canberra Distances

Figure 8.4 : CRBM: Sample Generation Algorithm.

Unlike CRBM, RBM visible layer dimension is equal to the length of signal data points. Once trained, RBM model generates a signal by applying one cycle of Gibbs sampling. After Gibbs sampling cycle, the generated signal is obtained at the visible layer. The pseudocode in Figure 8.5 shows the sample generation algorithm.

RBM: Sample Generation Algorithm
visible units: 128, hidden units: 200

n*: total sample size (value = 100)*
L2DistanceArr*: Array for storing Euclidean Distances*
DTWDistanceArr*: Array for storing DTW Distances*
CnbrDistanceArr*: Array for storing Canberra Distances*
l2min*: The global min L2 distance (initial value = 1000)*

1: **for** *i = 1, ...,n* **do**
2: $d \leftarrow$ Sample from test data $d \sim \mathcal{P}_t$
3: $g \leftarrow$ Sample RBM by applying d on visible units
4: *L2DistanceArr* $\leftarrow L_2$ Distance (g and d)
5: *DTWDistanceArr* $\leftarrow$DTWDistance (g and d)
6: *CnbrDistanceArr* $\leftarrow$ *CanberraDistance(g and d)*
7: $index_{l2min} \leftarrow$ *index(min(L2DistanceArr))*
8: $l2minCurrent \leftarrow$ *L2DistanceArr*$[index_{l2min}]$
9: **if** (*l2minCurrent* < $l2min$) **do**
10: $l2min$ = *l2minCurrent*
11: $g_{l2minglobal} \leftarrow g$
12: $d_{l2minglobal} \leftarrow d$
13: **end if**
14: **end for**
15: Compute statistics for L_2, DTW and Canberra Distances

Figure 8.5 : RBM: Sample Generation Algorithm**.**

8.5 LSTM Model for Prediction

The algorithm in Figure 8.6 shows the steps involved in the LSTM prediction model. LSTM model is configured to predict the next time step of the sensor signal by using the previous three signal data.

LSTM: Sample Generation Algorithm
# of inputs: 3, # of hidden units: 3, # of predicted units: 1

n*: total sample size (value = 100)*
ConditionArr*: Array for storing Time Delayed Units*
VisibleArr*: Array for storing Predicted Units Values*
L2DistanceArr*: Array for storing Euclidean Distances*
DTWDistanceArr*: Array for storing DTW Distances*
CnbrDistanceArr*: Array for storing Canberra Distances*
l2min*: The global min L2 distance (initial value = 1000)*

1: Compute *ConditionArr* and *VisibleArr* from Test Data
2: **for** *i = 1, ...,n* **do**
3: $\mathcal{d} \longleftarrow$ Sample from test data $\mathcal{d} \sim \mathcal{P}_t$
4: $\mathcal{g} \longleftarrow$ Sample LSTM with *test data*
5: *L2DistanceArr* $\longleftarrow L_2$ Distance ($\mathcal{g}$ and $\mathcal{d}$)
6: *DTWDistanceArr* $\longleftarrow$DTWDistance ($\mathcal{g}$ and $\mathcal{d}$)
7: *CnbrDistanceArr* $\longleftarrow$ *CanberraDistance($\mathcal{g}$ and $\mathcal{d}$)*
8: $index_{l2min} \longleftarrow$ *index(min(L2DistanceArr))*
9: $l2minCurrent \longleftarrow$ *L2DistanceArr[*$index_{l2min}$*]*
10: **if** (*l2minCurrent* < $l2min$) **do**
11: $l2min$ = *l2minCurrent*
12: $\mathcal{g}_{l2minglobal} \longleftarrow \mathcal{g}$
13: $\mathcal{d}_{l2minglobal} \longleftarrow \mathcal{d}$
14: **end if**
15: **end for**
16: Compute statistics for L_2, DTW and Canberra Distances

Figure 8.6 : LSTM: Sample Generation Algorithm.

Figure 8.7, Figure 8.8, and Figure 8.9 show the true and generated x-axis body acceleration signals for walking upstairs, walking downstairs, and walking activities. The generated signal from GAN and WGAN-GP is chosen by selecting the generated signal from the generation set having the closest distance to the true signal.

Figure 8.7 shows the true and generated x-axis accelerometer signals for walking upstairs activity for GAN, WGAN-GP, RBM, LSTM, and CRM models.

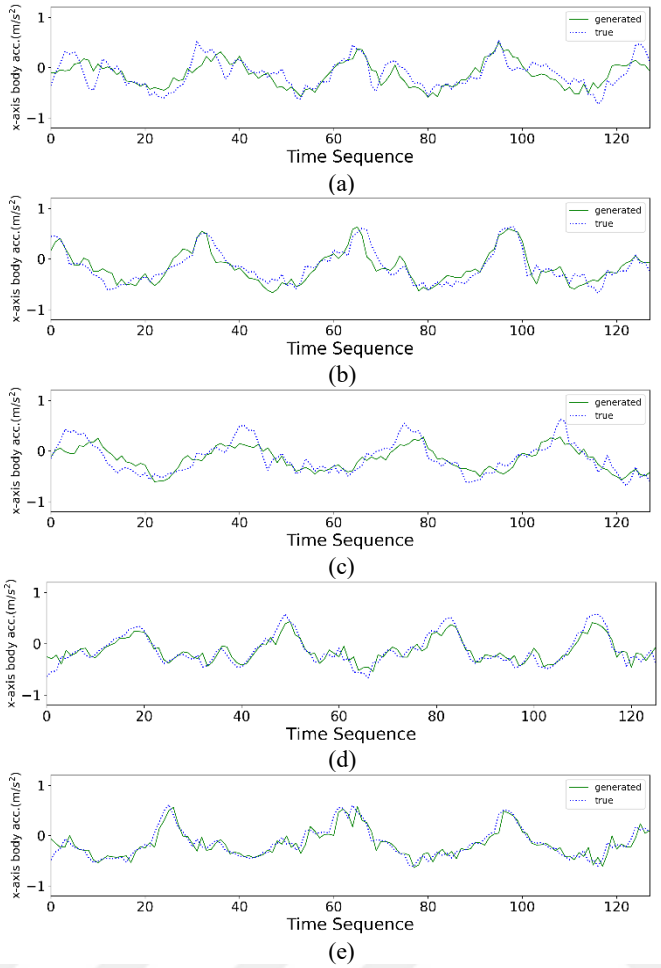

Figure 8.7 : True and Generated x-axis Body Acceleration Signals while walking upstairs for the models (a)GAN (b) WGAN-GP (c)RBM (d)LSTM (e)CRBM.

Table 8.7, Table 8.8, and Table 8.9 show generative models' similarity metric performance and R2 score statistics for walking upstairs, walking downstairs, and walking activities, respectively. Average and standard deviation of L2 norm, DTW, Canberra Distance, and R2 score distributions are indicated as mean and std in the tables below. The computation of the similarity metric statistics is presented in the pseudocode for sample generation algorithms.

Table 8.7 shows the L_2 Norm, DTW, Canberra Distance, and R2 score statistics between the true and the generated x-axis accelerometer signals for walking upstairs activity, plotted in Figure 8.7. CRBM outperforms the other generative models in almost all statistics. WGAN-GP shows better performance than GAN and RBM generative models. The worst performer is the GAN model among all. LSTM performs the closest to CRBM.

Table 8.7 : L_2 Norm, DTW, Canberra Distance, and R2 statistics between the generated and true samples for x-axis Body Acceleration while walking upstairs (sample size = 100).

	L2 Norm Distance				DTW Distance		Canberra Distance		R2 Score	
	mean	**std**	**min**	**max**	**mean**	**std**	**mean**	**std**	**mean**	**std**
GAN	3.42	0.47	2.43	4.84	23.01	3.3	62.16	7.6	-0.33	0.64
WGAN-GP	2.69	0.53	1.85	4.25	18.47	3.42	50.4	8.51	0.39	0.38
RBM	2.51	0.41	1.65	3.78	19.96	3.62	50.79	6.66	0.13	0.41
LSTM	2.01	0.32	1.54	3.41	14.39	2.73	42.12	4.43	0.54	0.07
CRBM	**1.91**	**0.29**	**1.42**	**2.82**	**14.37**	**2.42**	**39.61**	**4.09**	**0.76**	**0.06**

Figure 8.8 shows the true and the generated x-axis accelerometer signals for walking downstairs activity for GAN, WGAN-GP, RBM, LSTM, and CRM models, respectively.

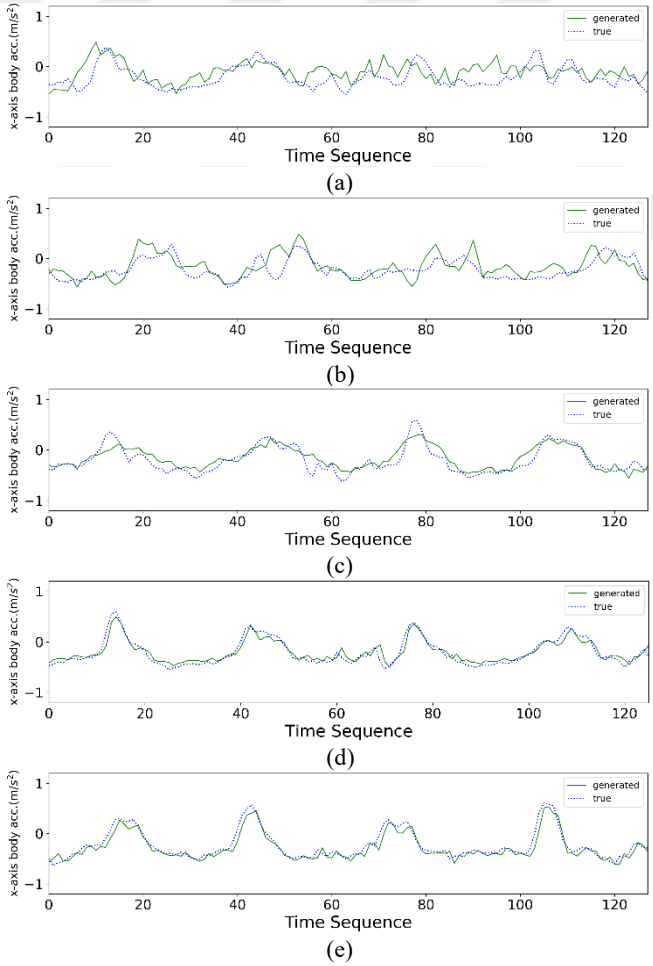

Figure 8.8 : True and Generated x-axis Body Acceleration Signals while walking downstairs for the models (a)GAN (b) WGAN-GP (c)RBM (d)LSTM (e)CRBM.

Table 8.8 shows the L_2 Norm, DTW, Canberra Distance, and R2 statistics between the true and generated x-axis accelerometer signals for walking downstairs activity, plotted in Figure 8.8. CRBM again clearly outperforms the other generative models in all distance metric statistics. RBM performs better than GAN and WGAN-GP models. LSTM again performs the closest to the CRBM model.

Table 8.8 : L_2 Norm, DTW, Canberra Distance, and R2 statistics between the generated and true samples for x-axis Body Acceleration while walking downstairs (sample size = 100).

	L2 Norm Distance				DTW Distance		Canberra Distance		R2 Score	
	mean	**std**	**min**	**max**	**mean**	**std**	**mean**	**std**	**mean**	**std**
GAN	3.48	0.57	2.38	5.03	22.29	4.08	58.28	7.22	-0.38	0.71
WGAN-GP	2.81	0.61	1.7	5.04	18.57	4.35	44.33	7.05	0.48	0.34
RBM	2.41	0.42	1.36	3.4	18.74	4.31	42.73	4.83	0.28	0.39
LSTM	2.04	0.38	**1.12**	2.72	17.08	3.76	32.93	3.54	0.62	0.07
CRBM	**1.88**	**0.34**	1.3	**2.61**	**13.63**	**2.95**	**30.11**	**3.12**	**0.77**	**0.06**

Figure 8.9 shows the true and the generated x-axis accelerometer signals for the walking activity for GAN, WGAN-GP, RBM, LSTM, and CRM models, respectively.

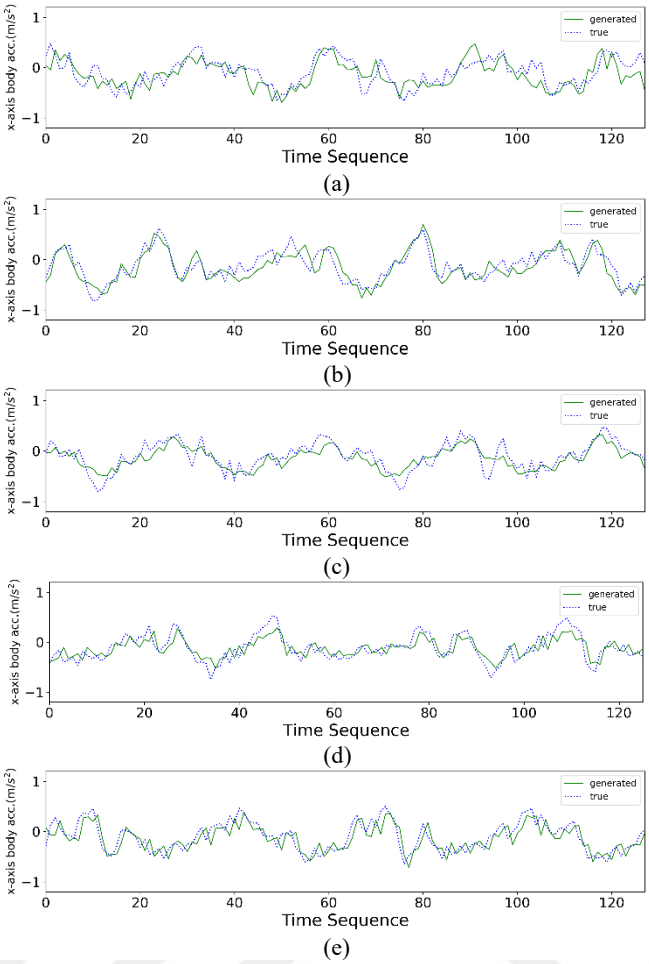

Figure 8.9 : True and Generated x-axis Body Acceleration Signals while walking for the models (a)GAN (b) WGAN-GP (c)RBM (d)LSTM (e)CRBM.

Table 8.9 shows the L2 Norm, DTW, Canberra Distance, and R2 statistics between the true and generated x-axis accelerometer signals for walking activity, plotted in Figure 8.9. CRBM again clearly outperforms the RBM, GAN, and WGAN-GP generative models in all distance metric statistics. GAN performs the worst among all the generative models. CRBM and LSTM models perform similarly to each other.

Table 8.9 :.L_2 Norm, DTW, Canberra Distance, and R2 statistics between the generated and true samples for x-axis Body Acceleration while walking (sample size = 100).

	L2 Norm Distance				DTW Distance		Canberra Distance		R2 Score	
	mean	**std**	**min**	**max**	**mean**	**std**	**mean**	**std**	**mean**	**std**
GAN	3.39	0.64	2.39	5.32	24.31	4.49	67.97	5.23	-0.75	0.7
WGAN-GP	2.66	0.46	1.96	3.92	20.34	3.23	56.93	6.15	0.23	0.37
RBM	2.71	0.52	1.91	4.29	23.2	4.86	59.77	3.78	-0.28	0.38
LSTM	2.51	0.37	**1.91**	3.67	18.39	3.67	57.13	**4.5**	0.02	0.11
CRBM	**2.49**	**0.27**	2.01	**3.26**	**15.8**	**2.46**	**53.84**	4.74	**0.47**	**0.11**

8.6 Statistical Analysis of the Results

Distance functions mentioned before will be used in the statistical analysis of the results. $d_1(x)$ is used as the distance function used for GAN and WGAN-GP models. The Euclidean distance $d_1(x)$ is the L_2 norm distance between the true sample $t \in \mathbb{T}$ and the generated sample $x \in \mathbb{Q}$ satisfying the min distance condition.

$$d_1(x) = argmin_{x \in \mathbb{Q}} \|x - t\|_2 \tag{8.3}$$

Equation 61 indicates the distance function used for RBM and CRBM models. The Euclidean distance, $d_2(x)$ below is the L_2 norm distance between the true sample $t \in \mathbb{T}$ and the generated sample $x \in \mathbb{Q}$.

$$d_2(x) = \|x - t\|_2 \tag{8.4}$$

When comparing the performance of the generative models, as a significance test, we would be formulating the null hypothesis by considering the L_2 distance between true and generated signals. We would like to compare the signal generation performance of the models in terms of their ability to generate the signal closest to the original signal. This proposition can be formulated by the following.

$$\mathbb{E}_{x \sim \mathbb{Q}_{CRBM}}[d_2(x)] \ncong \mathbb{E}_{x \sim \mathbb{Q}_{RBM,GAN,WGAN-GP,LSTM}}[d_{1/2}(x)] \tag{8.5}$$

The null hypotheses can then be given by the following statements.

$$H_0^1: \mathbb{E}_{x\sim\mathbb{Q}_{CRBM}}[d_2(x)] = \mathbb{E}_{x\sim\mathbb{Q}_{GAN}}[d_1(x)] \tag{8.6}$$

$$H_0^2: \mathbb{E}_{x\sim\mathbb{Q}_{CRBM}}[d_2(x)] = \mathbb{E}_{x\sim\mathbb{Q}_{WGAN-GP}}[d_1(x)] \tag{8.7}$$

$$H_0^3: \mathbb{E}_{x\sim\mathbb{Q}_{CRBM}}[d_2(x)] = \mathbb{E}_{x\sim\mathbb{Q}_{RBM}}[d_2(x)] \tag{8.8}$$

$$H_0^4: \mathbb{E}_{x\sim\mathbb{Q}_{CRBM}}[d_2(x)] = \mathbb{E}_{x\sim\mathbb{Q}_{LSTM}}[d_2(x)] \tag{8.9}$$

The above null hypotheses indicate that CRBM performance is no significant than RBM, GAN, WGAN-GP, and LSTM models in terms of the expected value of Euclidean distance between true and generated signals.

For the significance test, Student's t-test appears to be more reliable for typical hypotheses related to mean systems effectiveness (Urbano et al., 2019). Two-tail Student's t-test with an alpha level of 0.05 is chosen as the significance test.

Table 8.10 shows the t and p-values of the significance test for the three activities. T-values were computed by using the L_2 norm distance statistics presented in Table 8.7, Table 8.8, and Table 8.9. As can be seen from Table 8.10, all p-values except LSTM/CRBM significance for walking activity are less than the predefined alpha level of 0.05. With this finding, we can easily state that there is a significant difference between the performance of CRBM and GAN, WGAN-GP, and RBM models mutually with respect to L2 norm distance. Thus, we can immediately reject all three null hypotheses H_0^1, H_0^2 and H_0^3. On the other hand, LSTM and CRBM perform close to each other for walking activity, and thus we can only reject null hypothesis H_0^4 for walking upstairs and walking downstairs activities.

Table 8.10 :. t and p-values for significance test (sample size =100).

	WALKING		**WALKING UPSTAIRS**		**WALKING DOWNSTAIRS**	
	t-value	**p-value**	**t-value**	**p-value**	**t-value**	**p-value**
GAN/CRBM	12.96	4.82E-23	27.34	6.22E-48	24.11	3.21E-43
WGAN-GP/CRBM	3.19	0.0019	12.91	6.02E-23	13.32	8.50E-24
RBM/CRBM	3.75	0.0003	11.95	6.64E-21	9.81	2.87E-16
LSTM/CRBM	0.44	0.6633	2.32	0.0226	3.14	0.0022

9. DISCUSSION

Though we are comparing our generative models' performance in terms of some distance metric, the other performance concern is the training and generation times of the models. Table 9.1 shows for the three activities the generative models' mean and standard deviation of the training times, in seconds. GAN and WGAN-GP models were trained on a Google Colab environment because of their need for more CPU resources to train. On the other hand, RBM and CRBM models required less CPU power, and both models were trained on a laptop. Among the models, LSTM has the longest mean training time, while RBM has the least mean training time.

Table 9.1 : Training Times of each model for the three activities.

		Training Time Statistics in seconds				
		GAN	**WGAN-GP**	**RBM**	**LSTM**	**CRBM**
WALKING	mean	230.6	425.7	9.3	672.0	369.4
	std	46.1	21.8	0.4	23.0	27.0
WALKING UPSTAIRS	mean	212.7	409.8	8.3	596.0	455.0
	std	43.3	8.5	1.5	30.6	98.1
WALKING DOWNSTAIRS	mean	206.9	396.8	7.1	556.0	289.5
	std	41.4	11.1	0.3	45.6	49.4

Figure 9.1 shows the CRBM Model training time in seconds versus the number of training samples. The dots on the figure indicates the average training time in seconds per training sample. With a regression model, it can be easily seen that the relationship between CRBM training time and the number of training samples is linear. This leads to a time complexity as $O(n)$ where n is the number of training samples.

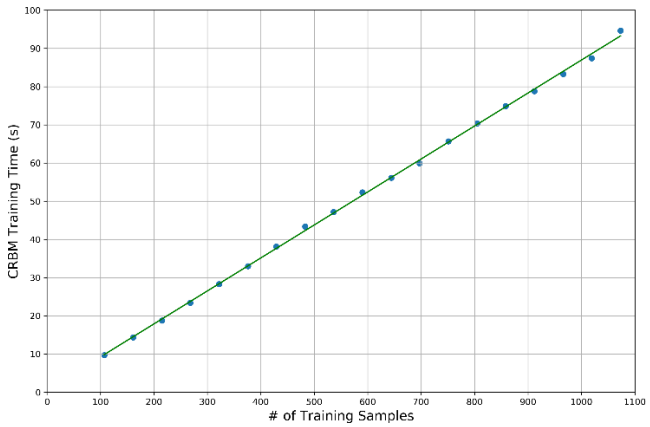

Figure 9.1 : CRBM Model Training Time vs Number of Training Examples.

10. CONCLUSION

For machine learning algorithms, raw data in the form of images, video, and sensor data is typically high dimensional, noisy, and redundant. Machine learning tasks, especially classification models require informative, representative, or discriminative features to be identified.

The raw data set can be noisy and redundant at first. To facilitate learning and improve generalization and interpretability, a key method in many machine learning applications is to pick a subset of features or generate a new and informative feature set. Feature learning involves a collection of techniques for converting raw data into features.

Feature learning is the process of converting raw data into features by combining domain knowledge and particular techniques. In machine learning and pattern recognition, the term "feature" refers to a broad idea. Any property can be a feature as long as it is valuable to the underlying algorithm. Apart from being a property, the function of a feature would be much clearer in the context of a problem. A feature is a trait that may assist in the solution of a problem.

Feature learning or representation learning in machine learning enables the underlying machine learning model to automatically discover the features from raw data. Feature learning replaces the manual feature engineering in which the representation of data is discovered in form of attributes of the underlying raw data by using the domain knowledge.

RBMs are energy-based two-layered stochastic neural networks that can automatically find intrinsic patterns in data by reconstructing input. The visible layer has input nodes, while the hidden layer is made up of nodes that extract feature information from the data. CRBMs are a special form of RBMs. In CRBM, the visible layer consists of time-delayed units which enable the model to capture temporal information inherent in raw data. CRBM’s ability to capture the time dependency in sensor data makes it suitable to be used for time-dependent data like signals etc.

In this study, it is shown that CRBM is capable of learning features and can generate samples closer to the original signals recorded from embedded sensors and outperform RBM, GAN, and WGAN-GP generative models. LSTM is also a strong model for prediction purposes in time series. Though it is a useful model for prediction in time series, LSTM is not suitable for feature learning. The ability of the CRBM model for generation, prediction, and feature learning in time series makes it distinguishable from other models.

CRBM excels in extracting temporal information from time-series data. Instead of using time and frequency domain feature extraction approaches, the model output on the hidden unit layer can be used as a feature for classification. In Wang et. al, CRBM is used in the speech domain for spectral envelope modeling and manages to obtain high-order features successfully in an unsupervised manner (Wang et al., 2016).

The following list highlights the findings of this study:

- A benchmark framework has been presented to compare and evaluate the capability of generative and predictive models in generating the sensor signals obtained from sensors embedded in wearable devices.
- It is shown that CRBM can generate sensor signals more accurately than GAN, WGAN-GP, and RBM models. LSTM shows similar performance for only WALKING activity.
- The ability to learn signal features accurately by using the CRBM model allows us to employ it as the building block for Deep Belief Networks which can be used as a classifier to identify and classify human activities. Such an architecture eliminates the need for pre-processing steps to extract features from the signals and also the need for using a separate classifier to classify human activities.

With this thesis study, the representation learning performance for sensor signals from a probabilistic perspective has been investigated and the results were compared with other generative models. The CRBM model as a probabilistic undirected graphical model can learn features from time-series data considering the temporal dependency information encoded in time-series data. Besides probabilistic approaches, as a future work, other representation learning approaches such as reconstruction-based algorithms can be investigated to evaluate the performance in generating the time-

series data. The family of autoencoder-based models falls into the category of reconstruction-based algorithms.

REFERENCES

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A Learning Algorithm for Boltzmann Machines. *Cognitive Science*, *9*(1), 147–169.

Ahmadi, A., Mitchell, E., Destelle, F., Gowing, M., O'Connor, N. E., Richter, C., & Moran, K. (2014). Automatic activity classification and movement assessment during a sports training session using wearable inertial sensors. *Proceedings - 11th International Conference on Wearable and Implantable Body Sensor Networks, BSN 2014*, 98–103. https://doi.org/10.1109/BSN.2014.29

Alpaydin, E. (2014). *Introduction to Machine Learning*. The MIT Press.

Alshurafa, N., Xu, W., Liu, J. J., Huang, M. C., Mortazavi, B., Roberts, C. K., & Sarrafzadeh, M. (2014). Designing a Robust Activity Recognition Framework for Health and Exergaming Using Wearable Sensors. *IEEE Journal of Biomedical and Health Informatics*, *18*(5), 1636–1646. https://doi.org/10.1109/JBHI.2013.2287504

Arjovsky Martin et al. (2016). Wasserstein GAN. *ArXiv*, *50*(06), 35–42.

Avci, A., Bosch, S., Marin-perianu, M., Marin-perianu, R., & Havinga, P. (2010). Activity Recognition Using Inertial Sensing for Healthcare , Wellbeing and Sports Applications : A Survey. *23th International Conference on Architecture of Computing Systems, January*, 167–176.

Ayrulu-Erdem, B., & Barshan, B. (2011). Leg motion classification with artificial neural networks using wavelet-based features of gyroscope signals. *Sensors*, *11*(2), 1721–1743. https://doi.org/10.3390/s110201721

Baarnhielm, A. (2017). *Multiple time-series forecasting on mobile network data using an RNN-RBM model* [Uppsala University]. http://www.teknat.uu.se/student

Bidargaddi, N., Sarela, A., Klingbeil, L., & Karunanithi, M. (2007). Detecting walking activity in cardiac rehabilitation by using accelerometer. *Proceedings of the 2007 International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP*, 555–560. https://doi.org/10.1109/ISSNIP.2007.4496903

Boyle, M., Edwards, C., & Greenberg, S. (2000). The effects of filtered video on awareness and privacy. *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, 1–10.

Caspersen CJ, Powell KE, C. G. (1985). Physical activity, exercise, and physical fitness: definitions and distinctions for health-related research. *Public Health Reports (Washington, D.C. : 1974)*, *100*(2), 126–131. https://doi.org/10.1093/nq/s9-IX.228.365-f

Charle K Chui. (1992). *An Introducrion to Wavelets*.

Clarke, M. R. B., Duda, R. O., & Hart, P. E. (1974). Pattern Classification and Scene Analysis. *Journal of the Royal Statistical Society. Series A (General)*, *137*(3), 442. https://doi.org/10.2307/2344977

D.M. Gavrila. (1999). The Visual Analysis of Human Movement : A Survey. *Computer Vision and Image Understanding*, *73*(1), 82–98.

Denton, Frank and Feaver, Christine and Spencer, B. (2005). Time series analysis and stochastic forecasting: An econometric study of mortality and life expectancy. *Journal of Population Economics*, *18*(2), 203–227.

Ermes, M., Pärkkä, J., Mäntyjärvi, J., & Korhonen, I. (2008). Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions. *IEEE Transactions on Information Technology in Biomedicine*, *12*(1), 20–26. https://doi.org/10.1109/TITB.2007.899496

Figo, D., Diniz, P. C., Ferreira, D. R., & Cardoso, J. M. P. (2010). Preprocessing techniques for context recognition from accelerometer data. *Personal and Ubiquitous Computing*, *14*(7), 645–662. https://doi.org/10.1007/s00779-010-0293-9

Fischer, A., & Igel, C. (2014). Training restricted Boltzmann machines: An introduction. *Pattern Recognition*, *47*(1), 25–39. https://doi.org/10.1016/j.patcog.2013.05.025

Fulcher, B. D., & Jones, N. S. (2014). Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering*, *26*(12), 3026–3037. https://doi.org/10.1109/TKDE.2014.2316504

Giorgino, T. (2009). Computing and visualizing dynamic time warping alignments in R: The dtw package. *Journal of Statistical Software*, *31*(7), 1–24. https://doi.org/10.18637/jss.v031.i07

Goodfellow, I. (2017). NIPS 2016 Tutorial: Generative Adversarial Networks. *ArXiv*, *1701.00160*. http://arxiv.org/abs/1701.00160

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ..., & Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, *3*(January), 2672–2680.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein GANs. *Advances in Neural Information Processing Systems*, *December*, 5768–5778.

He, Z., & Jin, L. (2009). Activity recognition from acceleration data based on discrete consine transform and SVM. *2009 IEEE International Conference on Systems, Man and Cybernetics*, *October*, 5041–5044. https://doi.org/10.1109/ICSMC.2009.5346042

He, Z., Jin, L., Zhen, L., & Huang, J. (2008). Gesture recognition based on 3D accelerometer for cell phones interaction. *IEEE Asia-Pacific Conference on Circuits and Systems, Proceedings, APCCAS*, 217–220. https://doi.org/10.1109/APCCAS.2008.4745999

Heinz, E. A., Kunze, K. S., Gruber, M., Bannach, D., & Lukowicz, P. (2007). Using wearable sensors for real-time recognition tasks in games of

martial arts - An initial experiment. *Proceedings of the 2006 IEEE Symposium on Computational Intelligence and Games, CIG'06*, *May*, 98–102. https://doi.org/10.1109/CIG.2006.311687

Hinton, G. E. (2012). A practical guide to training restricted boltzmann machines. In H. Springer, Berlin (Ed.), *(eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science: Vol. 7700 LECTU* (pp. 599–619). https://doi.org/10.1007/978-3-642-35289-8-32

Hinton, G. E., Sejnowski, T., & Ackley, D. (1984). *Boltzmann Machines: Constraint Satisfaction Networks That Learn*. Carnegie-Mellon University.

Hong, X., Nugent, C., Mulvenna, M., McClean, S., Scotney, B., & Devlin, S. (2009). Evidential fusion of sensor data for activity recognition in smart homes. *Pervasive and Mobile Computing*, *5*(3), 236–252. https://doi.org/10.1016/j.pmcj.2008.05.002

Hou, J. C., Wang, Q., Alshebli, B. K., Ball, L., Birge, S., Caccamo, M., ..., & Zeng, Z. (2007). PAS: A wireless-enabled, sensor-integrated personal assistance system for independent and assisted living. *Proceedings - 2007 Joint Workshop on High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability, HCMDSS/MDPnP 2007*, *2010*, 64–75. https://doi.org/10.1109/HCMDSS-MDPnP.2007.13

Hu, Y., Liu, J. N. K., You, J., & Chan, P. W. (2015). Continuous RBM based deep neural network for wind speed forecasting in Hong Kong. *Proceedings of the 2015 International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV 2015*, 368–374.

Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Snin, H. H., Zheng, Q., ..., & Liu, H. H. (1998). The empirical mode decomposition and the Hubert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *454*(1971), 903–995. https://doi.org/10.1098/rspa.1998.0193

Ishii, S., Luimula, M., Yokokubo, A., & Lopez, G. (2020). VR Dodge-ball: Application of Real-time Gesture Detection from Wearables to ExerGaming. *2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, 81–82. https://doi.org/10.1109/CogInfoCom50765.2020.9237824

Jakšić, Z., Cadenelli, N., Prats, D. B., Polo, J., Berral Garcia, J. L., & Perez, D. C. (2020). A highly parameterizable framework for Conditional Restricted Boltzmann Machine based workloads accelerated with FPGAs and OpenCL. *Future Generation Computer Systems*, *104*, 201–211. https://doi.org/10.1016/j.future.2019.10.025

Janidarmian, M., Fekr, A. R., Radecka, K., & Zilic, Z. (2017). A comprehensive analysis on wearable acceleration sensors in human activity recognition. *Sensors (Switzerland)*, *17*(3). https://doi.org/10.3390/s17030529

Jiang, W., & Yin, Z. (2015). Human activity recognition using wearable sensors by deep convolutional neural networks. *MM '15: Proceedings of the 23rd*

ACM International Conference on Multimedia, 1307–1310. https://doi.org/10.1145/2733373.2806333

Johnson, J., Williams, E., Swindon, M., Ricketts, K., Mottahed, B., Rashad, S., & Integlia, R. (2019). A Wearable Mobile Exergaming System for Activity Recognition and Relaxation Awareness. *2019 {IEEE} International Systems Conference, SysCon 2019, Orlando, FL, USA, April 8-11, 2019*, 1–5. https://doi.org/10.1109/SYSCON.2019.8836789

Karantonis, D. M., Narayanan, M. R., Mathie, M., Lovell, N. H., & Celler, B. G. (2006). Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring. *IEEE Transactions on Information Technology in Biomedicine*, *10*(1), 156–167. https://doi.org/10.1109/TITB.2005.856864

Keogh, E., Chu, S., Hart, D., & Pazzani, M. (2001). An online algorithm for segmenting time series. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 289–296.

Keogh, E., Chu, S., Hart, D., & Pazzani, M. (2004). Segmenting Time Series: A Survey and Novel Approach. *Machine Perception and Artificial Intelligence, Data Mining in Time Series Databases, Pp. 1-21 (2004)*, *57*, 1–21.

Krishnan, N. C., Colbry, D., & Juillard, C. (2008). Real time human activity recognition using tri-axial accelerometers. *Sensors Signals and Information Processing Workshop*.

Krishnan, N. C., Juillard, C., Colbry, D., & Panchanathan, S. (2009). Recognition of hand movements using wearable accelerometers. *Journal of Ambient Intelligence and Smart Environments*, *1*(2), 143–155. https://doi.org/10.3233/AIS-2009-0019

Krishnan, R., Jeena, K., & Prasad, K. P. (2020). Time series forecasting of Covid-19 using deep learning models: India-USA comparative case study. *Chaos, Solitons & Fractals*, *140*(November).

Kunze, K., Lukowicz, P., Junker, H., & Tröster, G. (2005). Where am I: Recognizing on-body positions of wearable sensors. *Location-and Context-Awareness,First International Workshop, LoCA*, 264–275.

Kuremoto, T., Kimura, S., Kobayashi, K., & Obayashi, M. (2014). Time series forecasting using a deep belief network with restricted Boltzmann machines. *Neurocomputing*, *137*(November 2019), 47–56. https://doi.org/10.1016/j.neucom.2013.03.047

Längkvist, M., Karlsson, L., & Loutfi, A. (2014). A review of unsupervised feature learning for time series classification. *Pattern Recognition Letters*, *42*, 11–24. https://www.m-culture.go.th/mculture_th/download/king9/Glossary_about_HM_King_Bhumibol_Adulyadej's_Funeral.pdf

Lara, Ó. D., & Labrador, M. A. (2013). A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys and Tutorials*, *15*(3), 1192–1209. https://doi.org/10.1109/SURV.2012.110112.00192

Lara, Ó. D., Prez, A. J., Labrador, M. A., & Posada, J. D. (2012). Centinela: A human activity recognition system based on acceleration and vital sign data. *Pervasive and Mobile Computing*, *8*(5), 717–729. https://doi.org/10.1016/j.pmcj.2011.06.004

Lei, Y., & Wu, Z. (2020). Time series classification based on statistical features. *Eurasip Journal on Wireless Communications and Networking*, *2020*(1). https://doi.org/10.1186/s13638-020-1661-4

Li, X., Kang, Y., & Li, F. (2020). Forecasting with time series imaging. *Expert Systems with Applications*, *160*(April). https://doi.org/10.1016/j.eswa.2020.113680

Li, X., Shen, H., Li, H., & Zhang, L. (2016). Patch Matching-Based Multitemporal Group Sparse Representation for the Missing Information Reconstruction of Remote-Sensing Images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *9*(8), 3629–3641. https://doi.org/10.1109/JSTARS.2016.2533547

Liu, H., & Fulcher, B. D. (2018). Feature-Based Time-Series Analysis. *Feature Engineering for Machine Learning and Data Analytics*, 87–116. https://doi.org/10.1201/9781315181080-4

Mannini, A., & Sabatini, A. M. (2010). Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, *10*(2), 1154–1175. https://doi.org/10.3390/s100201154

Meehan, C., Chaudhuri, K., & Dasgupta, S. (2020). A Non-Parametric Test to Detect Data-Copying in Generative Models. *ArXiv*, *2004.05675*. http://arxiv.org/abs/2004.05675

Mishra, M., Nayak, J., Naik, B., & Abraham, A. (2020). Deep learning in electrical utility industry: A comprehensive review of a decade of research. *Engineering Applications of Artificial Intelligence*, *96*(August), 104000. https://doi.org/10.1016/j.engappai.2020.104000

Mitchell, E., Monaghan, D., & O'Connor, N. E. (2013). Classification of sporting activities using smartphone accelerometers. *Sensors*, *13*(4), 5317–5337. https://doi.org/10.3390/s130405317

Mnih, V., Larochelle, H., & Hinton, G. E. (2011). Conditional Restricted Boltzmann Machines for structured output prediction. *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence, UAI 2011*, 514–522.

Mortazavi, B. J., Pourhomayoun, M., Lee, S. I., Nyamathi, S., Wu, B., & Sarrafzadeh, M. (2016). User-optimized activity recognition for exergaming. *Pervasive and Mobile Computing*, *26*, 3–16. https://doi.org/10.1016/j.pmcj.2015.11.001

Nie, S., Wang, Z., & Ji, Q. (2015). A generative restricted Boltzmann machine based method for high-dimensional motion data modeling. *Computer Vision and Image Understanding*, *136*, 14–22. https://doi.org/10.1016/j.cviu.2014.12.005

O.w.f. Nike+. (2009). *Nike+*. http://www.nikeplus.com/

Olszewski, R. T. R., Maxion, R., Siewiorek, D., & Banks, D. (2001). Generalized feature extraction for structural pattern recognition in time-series data [Carnegie Mellon University]. In *PhD Thesis* (Vol. 35, Issue February). https://doi.org/10.1016/S0031-3203(01)00152-2

Osogami, T. (2017). Boltzmann machines for time-series. *Technical Report RT0980*. http://arxiv.org/abs/1708.06004

Pentland, A. (2000). Looking at people: Sensing for ubiquitous and wearable computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(1), 107–110. https://doi.org/10.1109/34.824823

Raymond, R., Osogami, T., & Dasgupta, S. (2017). Dynamic Boltzmann Machines for Second Order Moments and Generalized Gaussian Distributions. *NIPS Time Series Workshop 2017*, *December*. http://arxiv.org/abs/1712.06132

Reiss, A., & Stricker, D. (2011). Introducing a modular activity monitoring system. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, *August*, 5621–5624. https://doi.org/10.1109/IEMBS.2011.6091360

Reyes-Ortiz, J. L., Oneto, L., Samà, A., Parra, X., & Anguita, D. (2016). Transition-Aware Human Activity Recognition Using Smartphones. *Neurocomputing*, *171*, 754–767. https://doi.org/10.1016/j.neucom.2015.07.085

Roberto Shinmoto Torres, Damith Ranasinghe, Q. S. (2014). Mobile and Ubiquitous Systems: Computing, Networking, and Services: 10th International Conference, MOBIQUITOUS 2013 Tokyo, Japan, December 2–4, 2013 Revised Selected Papers. *Mobile and Ubiquitous Systems: Computing, Networking, and Services: 10th International Conference, MOBIQUITOUS 2013 Tokyo, Japan, December 2–4, 2013*, *131*(September), 384–395. https://doi.org/10.1007/978-3-319-11569-6

Sajjad Hossain, H. M., Al Hafiz Khan, M. A., & Roy, N. (2017). SoccerMate: A personal soccer attribute profiler using wearables. *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 164–169. https://doi.org/10.1109/PERCOMW.2017.7917551

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, 2234–2242.

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied Soft Computing Journal*, *90*, 106181. https://doi.org/10.1016/j.asoc.2020.106181

Song, K., & Wang, Y. (2005). Remote Activity Monitoring of the Elderly Using a Two-Axis Accelerometer. *Proceedings of the CACS Automatic Control Conference*, 18–19.

Stikic, M., Larlus, D., & Schiele, B. (2009). Multi-graph based semi-supervised learning for activity recognition. *Proceedings - International Symposium on Wearable Computers, ISWC*, *January*, 85–92. https://doi.org/10.1109/ISWC.2009.24

Taylor, G. W., & Hinton, G. E. (2009). Factored conditional restricted Boltzmann machines for modeling motion style. *Proceedings of the 26th International Conference On Machine Learning, ICML 2009*, 1025–1032.

Taylor, G. W., Hinton, G. E., & Roweis, S. (2007). Modeling human motion using binary latent variables. *Advances in Neural Information Processing Systems*, 1345–1352.

Thomas Plötz, N. Y. H. (2011). Feature Learning for Activity Recognition in Ubiquitous Computing. *IJCAI 2011 - 22nd International Joint Conference on Artificial Intelligence*, 1729–1734. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-290

Urbano, J., Lima, H., & Hanjalic, A. (2019). Statistical significance testing in information retrieval: An empirical analysis of type I, type II and type III errors. *SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 505–514. https://doi.org/10.1145/3331184.3331259

URL-1 <https://www.youtube.com/watch?v=XOEN9W05_4A>. date retrieved 06.05.2019.

Vengertsev, D. (2014). Deep Learning Architecture for Univariate Time Series Forecasting. *Cs229*, 3–7.

Wang, Y., Zhao, S., Qu, D., & Kuang, J. (2016). Using conditional restricted Boltzmann machines for spectral envelope modeling in speech bandwidth extension. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, *1*(1), 5930–5934.

Weng, L. (2019). From GAN to WGAN. *CoRR*, *abs/1904.0*. http://arxiv.org/abs/1904.08994

Wu, P., Peng, H. K., Zhu, J., & Zhang, Y. (2012). SensCare: Semi-automatic activity summarization system for elderly care. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, *95 LNICST*, 1–19. https://doi.org/10.1007/978-3-642-32320-1_1

Xu, H., Liu, J., Hu, H., & Zhang, Y. (2016). Wearable sensor-based human activity recognition method with multi-features extracted from Hilbert-Huang transform. *Sensors (Basel, Switzerland)*, *16*(12), 2048. https://doi.org/10.3390/s16122048

Yang, J. Y., Wang, J. S., & Chen, Y. P. (2008). Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers. *Pattern Recognition Letters*, *29*(16), 2213–2220. https://doi.org/10.1016/j.patrec.2008.08.002

Yu, H. (2017). *A gentle tutorial on Restricted Boltzmann Machine and Contrastive Divergence*. *March*. https://doi.org/10.13140/RG.2.2.26119.60326

Zeng, M., Nguyen, L. T., Yu, B., Mengshoel, O. J., Zhu, J., Wu, P., & Zhang, J. (2015). Convolutional Neural Networks for human activity recognition using mobile sensors. *6th International Conference on Mobile Computing, Applications and Services*, *November*, 197–205. https://doi.org/10.4108/icst.mobicase.2014.257786

CURRICULUM VITAE

Name Surname : Erkan Karakuş

EDUCATION :

- **B.Sc. :**1995, İTÜ, Electrical&Electronics Faculty,Control and Computer Eng.
- **M.Sc.:**2000, Boğaziçi University, Electrical and Electronics Engineering
- **MBA:**2011, Boğaziçi University, MBA

PROFESSIONAL EXPERIENCE AND REWARDS:

Company	Location	Position	Dates
Freelance	Istanbul	IT Consultant	Nov 2020-Jan 2022
IBM	Istanbul	IT Contractor	Nov 2019-Nov 2020
ZURIX	Istanbul	Project Manager	March-Sept 2019
REDINGTON	Istanbul	Sr. DevOps Engineer & Sr. Middleware Consultant	Mar 2009-Dec 2018
TIBCO	Istanbul	IT Project Manager	Oct 2008-Jan 2009
BEA-CA Tech.	Istanbul	Sr. Technical Consultant	Mar 2006-Oct 2008
Havelsan Teknoloji Radar AŞ	Istanbul/Ankara	Team Leader /Senior Software Engineer	Feb. 2002-July 2004
Thales AirSys ATM	Istanbul Paris, FRANCE Malmoe, SWEDEN	Sr. Software Engineer	Oct. 1997– Feb. 2002

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Karakus, E.** and Kose, H**.** (2020), Conditional restricted Boltzmann machine as a generative model for body-worn sensor signals. IET Signal Process., 14: 725-736. https://doi.org/10.1049/iet-spr.2020.0154
- **Karakus, E.**, Kose, H., Ozen F. (2022), Statistical Feature Learning and Signal Generation for Sensor Signals, International Graduate Research Symposium-IGRS June 2022

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Karakus, Erkan** & Ogrenci, Selcuk & Dundar, G.. (2001). Parameter Quantization Effects in Gaussian Potential Function Neural Networks. 247-252. International Conference on Neural Networks and Applications, 2001, Spain