PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY MOHAMED BOUDIAF - M'SILA

FACULTY : Mathematics And Computer Science

DOMAIN : Mathematics And Computer Science

DEPARTMENT : Computer Science

BRANCH : Computer Science

N°:.....….....

OPTION: RTIC

A Thesis Submitted To Obtain The Master Degree

Prepared by: DJADJA SARRA & DJIAB DJIHAD

SUBJECT

Convolutional Neural Networks-Based Model Architecture for Signal Processing Applications

Supervised by: Dr. MOHAMED BENOUIS

Academic year: 2019/2020

ACKNOWLEDGEMENT

In the name of Allah, the Most Gracious and the Most Merciful,And prayers and peace be upon the holy Prophet Mohammed.

Firstly, we would like to thank God for giving us the power to complete this work, and because it illuminated our path until we achieved the goal that we aspired to.

We faced a lot of difficulties and pressures, and we were worried during the time we spent at home after the Corona crisis, But despite all that, with the help of God first, and with the help of our supervisor ***Dr. Mohamed Benouis****, who offered us the opportunity to work on this project and had enough patience to review and correct our work over and over. So, thank you very much for your guidance, understanding and patience, it was pleasure to work with you. It has been our great pleasure and honor to have a supervisor like you.*

Deepest gratitude is also due to ***Dr. Gamougi****, our second supervisor. Thank you for everything.*

Thank you to our parents who have been supportive during our educational career for many years. They always believed in us despite everything.

We are grateful and happy with our friendship and work together. We hope that God sustains our friendship and love. And make the duo ***Sarah*** *and* ***Jihad*** *together in this world and paradise.*

Last but not least, deepest thanks go to all people who took part in making this thesis real.

Djihad djiab Djadja sarra

DEDCATION

This thesis is dedicated to:

The sake of Allah, our Creator and our Master, our great teacher and messenger, Mohammed (MayAllah blessand grant him), who taught us the purpose of life.

Our deceased friend ***Assia Belkacem****, God have mercy on her.*

Our ***Mothers*** *and* ***Fathers*** *for their love, endless support and encouragement; to our beloved brothers****(«Djadja Raouf****,* ***Djadja Toufik****,* ***Djadja Tayab****,* ***Djadja Abdelghani****,* ***Djadja Amar» ; «Djiab Mohibeddine****,* ***Djiab Takieddine****,* ***Djiab Aalaeddine»),*** *and sisters****(«Djadja Samiha****,* ***Djadja Aicha****,* ***Djadja Marya»; « Dc.Djiab Malak****,* ***Djiab soulef»)****; and cousins****(«Djadja Khadidja, Chichi Amira, Chichi Douaa, Mechri Kenza»; «Djiab Romyessa****,* ***Djiab Aya****,* ***Mezaache Abderrahmen»)****; to our family****(«Djadja, Boukharouba»; «Djiab, Mezaache»)****; to our friends And in particular* ***Waffa Bouras****,* ***Ahlam Attar****,* ***Imane Belarbi****,* ***Rouaa Djadja****,* ***Bouchra Djriou****,* ***Mariem Amroune****,* ***Marya Bensalem****,* ***Chayema Slimankadi****,* ***Djalal Djemiat****,* ***Kamal Benyahya****,* ***Ilyes Berra****,* ***Karim Wadah*** *and to everyone who knows and loves* ***jihad*** *and* ***Sarah****.*

Our colleagues and profs in the faculty of Mathematics and Computer Science and the university ***MOHAMED BOUDIAF – M'SILA****.*

Table of Contents

N°	Contents	Page
1	Dedication	I
2	Acknowledgements	II
3	Table of Contents	IV
4	List of Tables	VI
5	List of Figures	VIII
6	List of Abbreviations	XIII
7	General introduction	1
	Chapter I **Study Background**	
1.1	Introduction	5
1.2	Deep Learning	6
1.2.1	Why deep learning	8
1.2.2	Different architectures of Deep Learning	8
1.2.3	Deep Learning Frameworks	8
1.3	Convolutional Neural Network	9
1.3.1	CNN architecture and layers	10
1.3.2	Different existing layers	10
1.3.2.1	Convolutional layer	10
1.3.2.2	Pooling Layer	15
1.3.2.3	Fully connected layer	16
1.3.3	The more popular architectures of CNNs	16
1.3.4	The related works	28

1.4	Conclusion	29
	Chapter II **1D Convolutional Neural Network**	
2.1	Introduction	31
2.2	CNN 1D architecture and layers	32
2.2.1	Forward and Back-Propagation in CNN layers	33
2.2.2	ECG And EMG 1D-CNNs applications	37
2.3	Comparison between 1D CNN and 2D CNN	40
2.4	The related works	41
2.5	Conclusion	42
	Chapter III **Local Binary Patterns**	
3.1	Introduction	45
3.2	Local Binary Patterns	46
3.2.1	LBP Operation (Example)	46
3.2.2	Domain of applications of LBP	47
3.3	Local binary convolution neural network	48
3.3.1	The related works	49
3.4	Material and Methods	51
3.4.1	ECG Database	51
3.5	Experimental Results and Discussion	54
3.6	Conclusion	56

	General Conclusion	59
	References	62
	Abstract in English, Arabic and French	71

List of Figures and Tables

List of figures:

N°	Subject	Page
Figure 1.1	The relation between AI, ML and deep learning**.**	6
Figure 1.2	The process of ML classic compared to that of Deep Learning**.**	7
Figure 1.3	The difference in performance between DL and most ML algorithms in depending on the amount of data.	7
Figure 1.4	Different DL frameworks	8
Figure 1.5	The “LeNet” architecture.	10
Figure 1.6	Convolutional layer operation.	11
Figure 1.7	Input and output of a convolutional layer.	12
Figure 1.8	Filters of a trained AlexNet	13
Figure 1.9	Summary of the main activation functions.	14
Figure 1.10	Max pooling example.	15
Figure 1.11	Model architecture of 2D-CNN build “Cifar-10” image classifier	17
Figure 1.12	“CIFAR-10” input image	18
Figure 1.13	Model Loss/ Accuracy of Convolution keras model	19
Figure 1.14	Confusion Matrix Convolution keras model	21
Figure 1.15	Architecture model of 2D-CNN for “MNIST” Digit recognition	21
Figure 1.16	The **“**MNIST” input image	22

Figure 1.17	Model Loss/ Accuracy of 2D CNN for "MNIST" Digit Recognition	23
Figure 1.18	Output image of 2D CNN Architecture for "MNIST" Digit Recognition model	24
Figure 1.19	ECG sequence with zero padding	25
Figure 1.20	Spectrogram Function	25
Figure 1.21	Architecture of the proposed network in example 03	26
Figure 1.22	Spectrogram without/with log transform	27
Figure 1.23	Classification Electrocardiogram	27
Figure 2.1	A sample 1D CNN configuration with 3 CNN and 2 MLP layers.	32
Figure 2.2	Three consecutive hidden CNN layers of a 1D-CNN	33
Figure 2.3	Forward and back-propagation in hidden CNN layers.	35
Figure 2.4	Forward and back-propagation between the last hidden CNN layer and the first hidden MLP layer.	36
Figure 2.5	ECG signal for every category	38
Figure 2.6	ECG signal	38
Figure 2.7	Proposed architecture model in example 01	39
Figure 2.8	Confusion matrix for heartbeat classification on the test set	40
Figure 3.1	Example of LBP calculation. (a) Gray-scale map, (b) binary value set, and (c) code set.	47
Figure 3.2	Basic module in CNN and LBCNN. Wl and Vl are the learnable weights for each module.	49
Figure 3.3	LBCNN/CNN architecture model	49
Figure 3.4	Graphical Representation of short segment of ECG as 40×40 spectrogram energy images	53

List of tables:

N°	Subject	Page
Table 1.1	Different architectures of Deep Learning	8
Table 1.2	Example of popular CNNs architectures	16
Table 1.3	Parameters number of Convolution keras model	19
Table 1.4	2D CNN architecture for "MNIST" Digit Recognition	22
Table 1.5	Parameters number of 2D CNN for "MNIST" Digit Recognition	23
Table 1.6	Arrhythmia classes table	27
Table 2.1	Arrhythmia classes table	37
Table 2.2	Comparatif table between 1D-CNN and 2D-CNN	41
Table 3.1	LBCNN architecture used in the ECG biometric identification system	53
Table 3.2	The number of ECG spectrogram images for each database used to validate our approach.	55
Table 3.3	Performance of LBCNN model	55
Table 3.4	Comparison of the performance evaluation of the proposed approach to other existing biometric method based on deep learning approach.	56

Table of equations:

N°	Subject	Page
Equation 1.1	Calculating stride	13
Equation 2.1	One- Dimension forward propagation formula	33
Equation 2.2	Calculating the intermediate output	33
Equation 2.3	Back propagation formula	34
Equation 2.4	Calculating the chain-rule of derivatives bias and weight	34
Equation 2.5	Calculating the regular (scalar) BP	34
Equation 2.6	Calculating the delta error	34
Equation 2.7	Calculating the BP of the delta error	35
Equation2.8	Calculating the weight and bias sensitivities	35
Equation 2.9	Calculating the updates biases and weights with the learning factor, ε.	35
Equation 3.1	Calculating Local binary pattern	46
Equation 3.2	Calculating Local neighborhood binary code	46
Equation 3.3	Sparsity formula	48
Equation 3.4	STFT formula	52
Equation 3.5	Spectrogram formula	52
Equation 3.6	HTER formula	54

Abbreviation table:

Abbreviations	Full Form
AF	Atrial Fibrillation
ANN	Artificial Neural Network
BNN	Binarized Neural Network
BP	Back-Propagation
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CWT	Continuous Wavelet Transformation
DBN	Deep Belief Network
DL	Deep Learning
DSNs	Deep Stacking Networks
ECG	Electrocardiogram
EER	Equal Error Rate
EMG	Electromyography
FAR	False Acceptance Rate
FC	Fully Connected
FN	
FP	Forward Propagation
FRR	False Rejection Rate
GA	Genetic Algorithm
GCN	Gabor Convolutional Networks
GPU	Graphics Processing Unit
GRU	Grated Recurrent Unit
IA	Intelligence Artificial
KNN	K-Nearest Neighbor

LBC	Local Binary Convolutional
LBCNN	Local Binary Convolutional Neural Network
LBP	Local Binary Patterns
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean-Squared Error
PP	Post-Process
QNN	Quantized Neural Network
RBF	Radial Basis Function
RNNs	Recurrent Neural Networks
SGD	Stochastic gradient descent
SPEC	Spectrogram
STFT	Short-Time Fourier Transform
SVHN	Street View House Numbers

General introduction

GENERAL INTRODUCTION

A neural network considered a powerful tool inspired by the brain of humans, which is made to simulate the human brain task. Thus, is performed by designing a suitable function activation and training data to extract a useful pattern for a specific task.

The neural network imitates the mechanism of the brain, which is composed of connections of nodes, however, they mimics the neurons association, which is the most important mechanism of the brain, neural network use connection weights of neural the information of the neural network is stored in the form of weights and bias and the input signals are multiplied by the weight before entering the node, the output of the node is the weights sum.

Many researchers have been well successful develop a new deep learning algorithm which are mostly deployed in many disciplines to solve a complex task such machine visions, robotic, mobile network medicine agriculture and so on. One of the most well-known algorithms is CNN which firstly designed for image vision tasks, however, for signal application, we have found a few works that adopted CNN model. Here, we have studied the CNN model for ECG application by investigating on some exiting CNN architecture and testing a recent CNN model, called LBCNN to show it performance on the human identification task as case study.

Motivation

Most of the current state-of-the-art ECG analyses techniques suffer from problems which are mainly related to the ECG measurement condition setting. For example, some of the existing methods do not perform well on ECG high variants, which can be originated by time-varying nature of the ECG signals, irregular conditions of cardiac18 disorders, and the long waiting period to collect the ECG data. In order to overcome such limitations, in this dissertation we propose to perform ECG feature extraction by exploiting CNN techniques. These have been widely used in image processing and many applications related to machine vision, where they were applied to an image instead of signal. Recently, a few studies have proposed 1D CNN (i.e, Unlike of 2DCNN structure, 1D-CNN having low complexity and parameter's' network) architecture for ECG processing and proposed 1D-CNN. The CNN is automatic powerful feature extraction and is promising as it has the ability to extract discriminative information

from the ECG waveform without requiring some handcraft features techniques or preprocessing step. Recently, may works have criticized the standard CNN for many application tasks where needs a huge parameter and hard tuning experimental to achieve a good performance. To deal with these challenges, a new architecture are raised in many applications, in particular, for signal applications such as EEG, EMG and ECG. Motivated by those works, we are firstly willing to carry out the CNNs architecture for ECG application to study its outcome and the weakness on the well-known ECG database. At last, we designed a specific CNN architecture (2DCNN, 1DCNN, 2DLBCNN) for ECG anomaly detection and human identification.

Statement of the problem

Although CNNs architecture have achieved outstanding results. There are still applications couldn't achieve a high performance regarding their data nature. On other hand, the CNN serves only the data with two dimensions, in this way, we have to find a good representation to fed into the CNN network to reach a high performance, especially in bio signals modeling, personal biomedical data classification, early diagnosis, structural health monitoring, etc.

On other hand, our aim is to find an alternative CNN structure may handle the ECG signal complex structure and extract the relevant features to achieve high accuracy and lower error rate in the ECG classification.

Contributions

To fulfill our mission, we made the following contributions:

- We have tested a different CNNs architecture including 1D and 2D for ECG application.

- We have designed a CNN architecture by training the 2D-CNN network with ECG spectrum energy.
- We have studied and designed an alternative CNN model called, two-dimensional LBCNN network as end to end ECG human identification.

Report outline

Our work was divided into three chapters:

- **The First chapter:** revolved around explaining deep learning, different architecture of deep learning, we chose two-dimension convolutional neural network, reinforced with three application examples to “MINIST”, “CIFAR-10”, “MIT-BIH” data bases.
- **The Second chapter:** contained the implementation and explaining 1-Dimension convolutional neural network architecture, we applied it to “MIT-BIH” database and comparison between 2D-CNN and 1D-CNN.
- **The Third chapter**: in the last chapter, we explain LBCNN architecture, and we explained as an approach for ECG human identification. finally, a general conclusion was concluded.

Chapter 1

Study Background

1.1 Introduction

The organs in the human body ensure the healthcare of human being through the information circulates in the whole body. This information can be recorded through physiological medical sensors that may measure the heart rate, blood pressure, oxygen saturation levels, blood glucose, nerve conduction, brain activity, etc. Traditionally, these measurements are taken at specific points in time and recorded on the patient chart. So, the field of biomedical engineering faces many challenges because it combines product development and innovative technology [1]. Signal analysis is important sub field in biomedical engineering, because it provides an efficient way to extract the relevant information from 'raw' data. These systems exhibit many favorable characteristics for biomedical engineering such as extreme ability to extract the relevant information from the signal noisy, invariant to time and frequency scaling and spatial [2]. The signal analysis strength of the data extraction depends upon the environment system and algorithm scheme. Signal analysis divided into two classes, spatial and frequency domain. In literature [3], the most used signal extractions for biomedical application are Fourier Transform, wavelet, fractal space [4]. Many researchers have also pointed weakness in behavior of conventional signal extractions like non-linearity system, complex algorithm in huge space, finite precision effect, and computational complexity [5]. While the ultimate goal of signal modeling is to abstract data, it explicitly aims to make the system high accurate, smart, automatic task. Thanks to engineering applications, deep learning is making it possible to model data extremely well, without using complex purposes about the modeled system. Deep learning can usually better describe data than biomedical models and thus provides both engineering solutions and an essential benchmark. It can also be a tool to advance understanding. The vast field of Deep learning is a radically different way of approaching modeling that relies on minimal human insight. We focus here on the most popular sub discipline, CNN, which assumes that the original data can be processed through several layers to extract implicitly the most important information without any handcraft engineering tool as they used in machine learning applications [6].

1.2 Deep Learning

Deep Learning (DL) is the new subfield generation of IA domain (see **figure1.1**) where it has achieved a remarkable success in various applications over recent years. Indeed, in Google, Microsoft, Twitter, Facebook, Mercedes, Huawei have been launched many program researches where they have investigated more billions of deep learning application [7]. Unlike to the artificial network, deep learning is made to handle large amounts of data by adding layers to the network. A deep learning model has the capacity extracting characteristics from raw data through multiple layers of processing composed of multiple linear and nonlinear transformations and learns about these small features to small through each layer with minimal human intervention.

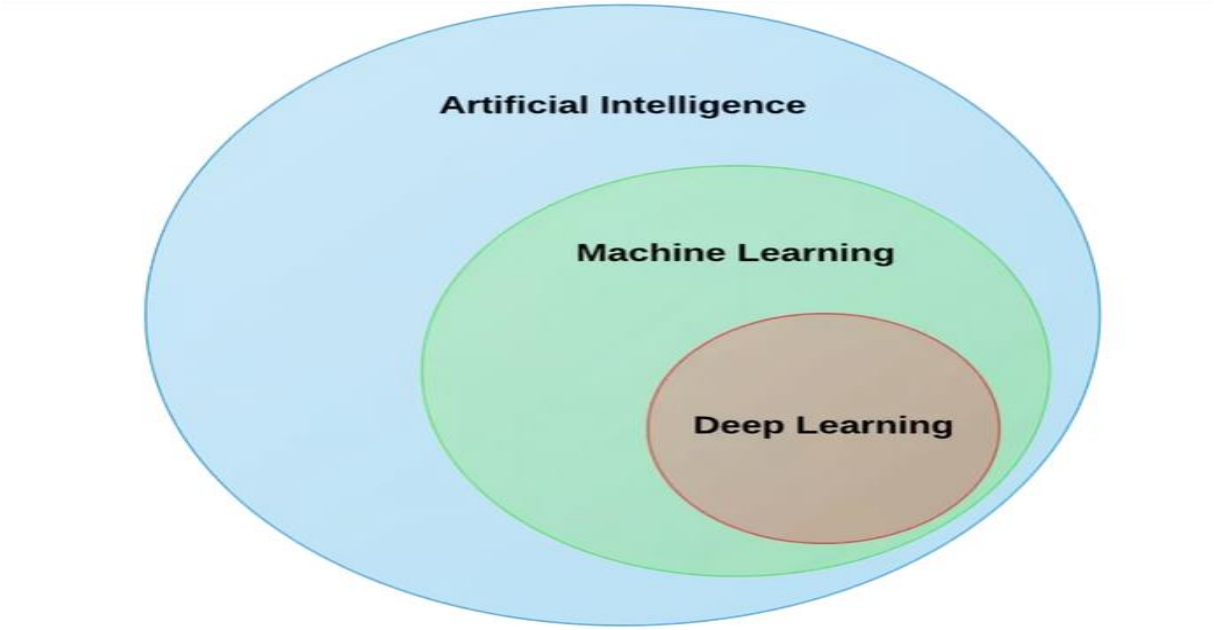


Figure 1.1: The relation between AI, ML and deep learning [8].

1.2.1 Why deep learning

The development of deep learning was designed to perform automatic feature extraction from raw data without any handcart features tool and also can be performed in huge data space, as it depicted in figure 1.2.

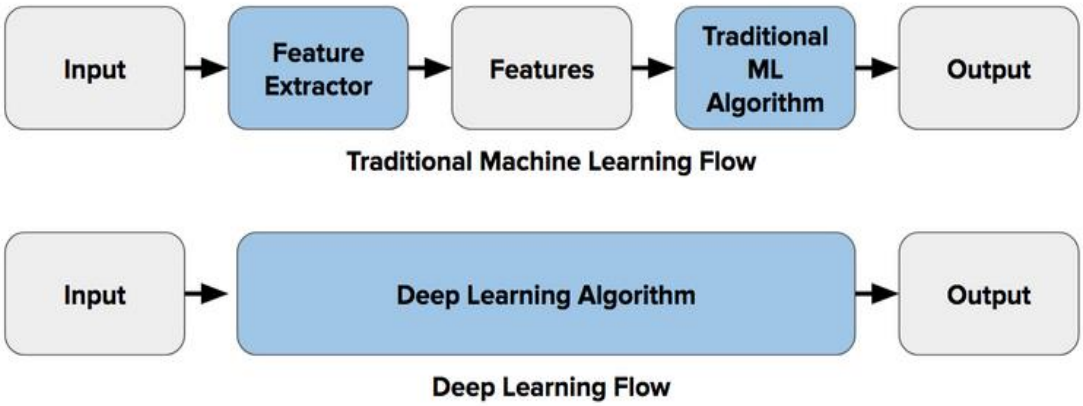


Figure 1.2: The process of ML classic compared to that of Deep Learning [9].

Deep learning models are trained by using large amount of data and a combination of many layer sets that learn features directly from the data without the need for prior feature extraction, which took too much time-consuming step.

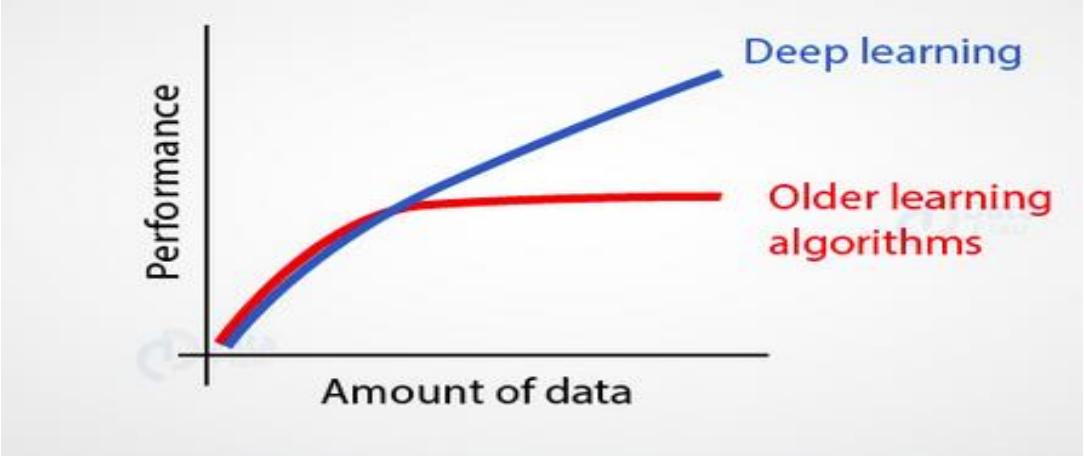


Figure 1.3: The difference in performance between DL and most ML algorithms in depending on the amount of data [10].

The main differences between DL and traditional ML is its performance as the scale of data increases when the data provided increases (figure 1.3), deep learning algorithms become more convenient, this is because DL algorithms need a large amount of data.

Contrary to many conventional ML algorithms, which can receive a specific amount of data, sometimes called “Performance Tray”.

1.2.2 Different architectures of Deep Learning:

There are a wide variety of buildings and algorithms used for deep learning [11].Over the past 20 years there have been five deep learning structures: recurrent neural networks (RNNs), long short-term memory (LSTM)/gated recurrent unit (GRU), convolutional neural networks (CNNs), deep belief networks (DBN), and deep stacking networks (DSNs).

LSTM and CNN are two of the oldest methods on this list, but also one of the most widely used in various applications.

Table 1.1: Different architectures of Deep Learning

Architecture	**Application**
RNN	Speech recognition, handwriting recognition
LSTM/GRU network	Natural language text compression, handwriting Recognition, speech recognition, image captioning
CNN	Image recognition, video analysis, natural language processing
DBN	Image recognition, information retrieval, natural language understanding, failure prediction
DSN	Information retrieval, continuous speech recognition

1.2.3 Deep Learning Frameworks:

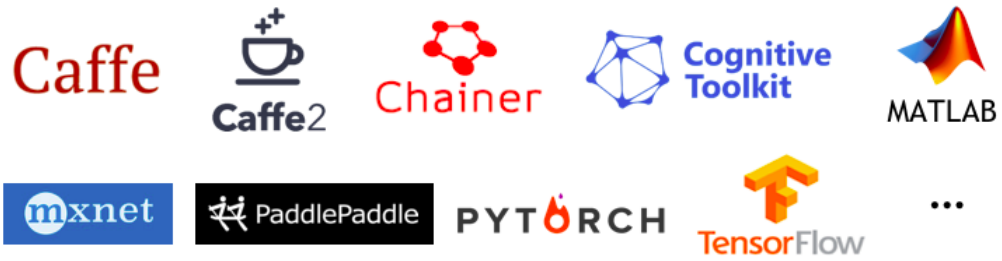


Figure 1.4: Different DL frameworks [12].

1.3 Convolutional Neural Network:

Human organ makes a suitable decision based on the information provided by the neural cell, in this way, the secret success of CNN is to well have simulated the brain cell mechanism to abstract the data based on its information, for example, an image can be classified based on pixel values, shape geometry, edge and so on.

On other hand, Convolutional neural network has a different architecture than standard neural network (Multi-Layer Perceptron),thus is consisting of three layers: input layer, hidden layer and output layer, each layer is made up of a set of neurons, where each layer is fully connected to all neurons in the layer before, finally there is a last fully connected layer (the output layer) that represent the prediction.

CNNs consist of multiple layers of small neuron collections that process portions of the input image, called receptive field. The main applications of convolutional neural networks are related with image recognition such as image classification, object detection, facial recognition [7]. However, CNNs application aren't limited to computer vision but also has been adopted in universal fields like biological analysis, electric vehicle, smart grid, robotic, mechanic (fault detection) and so on [7].

1.3.1 CNN architecture and layers:

The convolutional neural network consists primarily of a group of consecutive convolutional layers. So, each input neuron binds to all output neurons in the next layer, this is called fully connected layers or affine layer and in CNN architectures, their use is reserved for the final layers and in CNN structures, its use is allocated to the final layers.

For example, we take the ancestor of CNNs "LeNet"[7]. There are two interleaved convolutional and pooling layers following by three (two hidden and one output) fully_connected layers. The output layer is composed of 10 Radial Basis Function (RBF) neurons each of which computes the Euclidean distance between the network output and ground truth label for 10 classes.

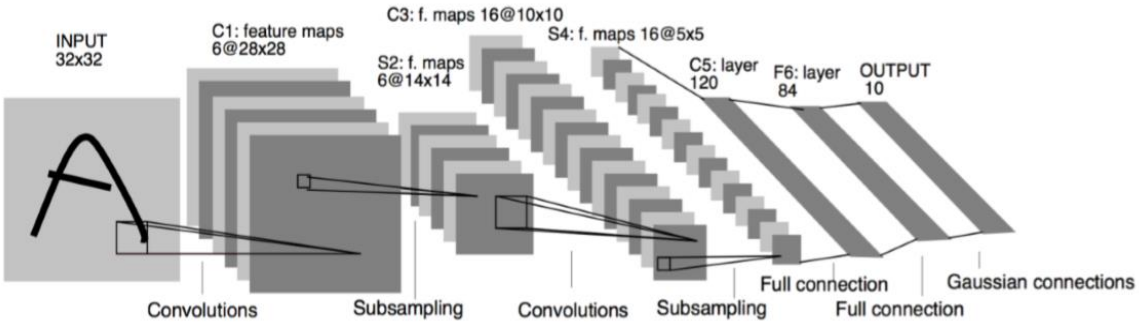


Figure 1.5: The "LeNet" architecture [7].

CNN structures can vary in size and complexity but share the same underlying infrastructure. Many convolutional layers are linked and the nonlinear activation function is applied to the results of each layer. Subsampling or pooling mechanism is putted between each of these convolutional layers. Thus, sequential structures are usually composed of three layers: convolutional, activating function, and pooling. After then, some fully connected layers are often added to the end, just before the final layer which depends on the problem the CNN is trying to solve. This final layer can be a classifier/regress or with respect the application task.

1.3.2 CNN layers:

1.3.2.1 Convolutional layer:

The convolutional layer is the most important and the core block of a CNN. First of all, notice that in conventional neural networks or in fully connected layers, the inputs are depicted as a vertical line of neurons. Instead of doing this, in a CNN we structure the inputs as a square of neurons whose values correspond to the pixel intensities of the input image. Behind the two dimensions space (width and height), these squares or matrix also have a depth. If the input image is grey, the depth is equal to one, whereas an RGB image has a depth of three. However, all three dimensions vary in size as they advance through the network and the different layers are applied and the fundamental two properties of the convolutional layers, "weight sharing" and "limited connectivity" exist even in the most-recent CNN architectures. In fact, these are the main features, which separate CNNs from the conventional MLPs. otherwise; both networks are homogenous and share the common linear neuron model.

As depicted in the figure 1.6, convolutional layers are composed by several filters of the same width and height. The depth of these filters must be equal to the depth of the input volume. Meanwhile, the width and height of the filters are smaller so they can be shifted during the convolution.

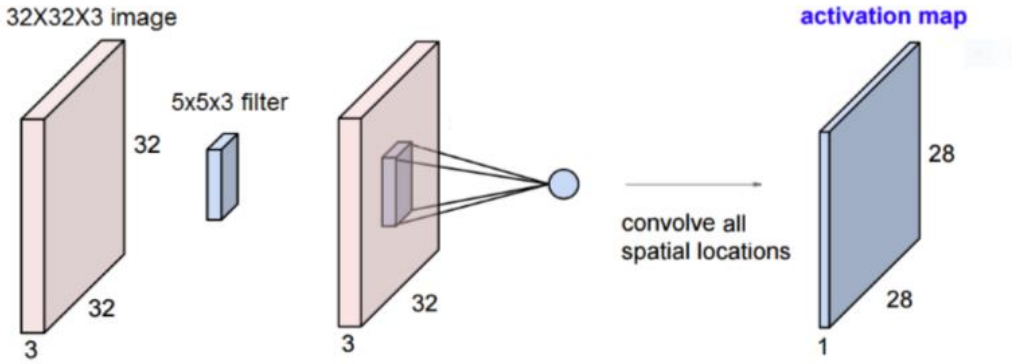


Figure 1.6: Convolutional layer operation. [13]

As it can be seen in **figure 1.6**, when doing the computation each filter only is applied to a small number of pixels at each step. Thus, each neuron at the output is only connected to a small region of the input neurons, which is called the receptive field. During the forward process, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input, and producing a 2-dimensional activation map of that filter. Each connection or neuron learns a weight and at each layer an overall bias is learnt as well. Notice that the output width and height are smaller than the input ones as a result of the convolution applied [13].

The full output volume of the convolutional layer is formed by stacking the activation maps of all filters along the depth dimension. In particular, there is one activation map for each filter and it is important to note that the depth of the output is determined by the number of layer filters. Instead, the depth of each filter is determined by the input depth. As shown in figure 1.7, we can see the input and output of the convolutional layer. It can be seen that six activation maps have been produced, which means that six filters have been applied to the inputs. For depth, all these filters must have three depths to match the input dimensions.

If another convolutional layer is applied to the output, the number of filters can be applied but each one must have a depth of six.

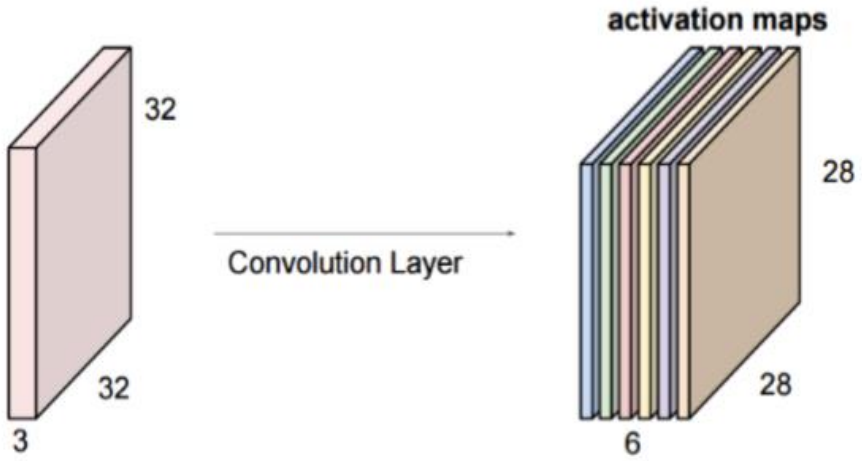


Figure 1.7: Input and output of a convolutional layer. [13]

When calculating the activation map, we go back to the concept of local receptive fields, and remember that each resulting neuron is connected to a small area of input. Connections are local in space (length and width), but always extend to the depth of the entire volume. This structure ensures that the learned filters produce the strongest response to the local spatial input pattern. As a result, the grid learns which filters are active when you see a certain type of feature in some spatial positions in the entry.

We can reinterpret the meaning of the output size. For each depth value, based on the structure of the activation maps where we have a different activation map calculated by a filter designed to identify specific shapes or patterns. At the same time, to obtain a certain position of width and height, we have a number of values along the depth that corresponds to the same input area. Then, each of these value corresponds to the application of different filters.

In figure 1.8, these diagrams indicate the shapes and colors targeted by each candidate and are attempted to detect these typical shape filters on the first convolutional layer of trained "AlexNet", the well-known CNN. For example, there are filters running

on green dots or vertical frequencies. In the deeper layers, this interpretation is complicated because filters are less form and definition.

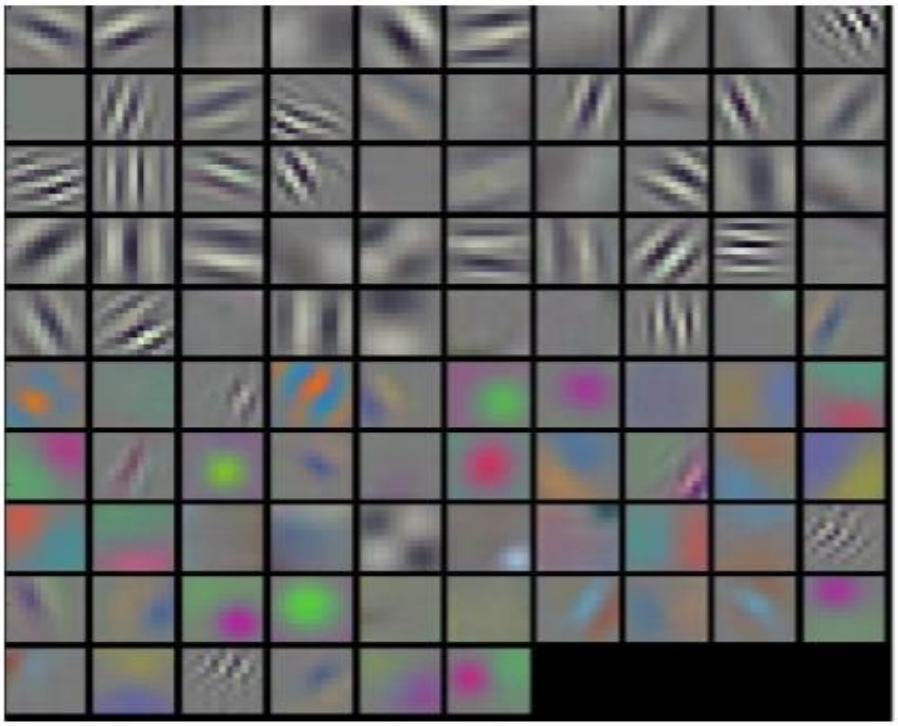


Figure 1.8: Filters of a trained AlexNet [13].

Finally, we highlight two hyper parameters in convolutional layers, although they exist in other layers:

- **Stride:** This parameter can take any value that makes the next expression produce an integer. This parameter specifies the number of rows or columns that the filter is removed during the convolution.

$$\frac{N + F}{stride} + 1 \qquad (1.1)$$

Equation 1.1: Calculating stride

N is the input size and **F** corresponds to the filter size. Thus, not all the values for the stride are possible given an input and a filter. If the previous condition is not fulfilled, when shifting the filter, at the last step, the filter edges do not match the input edges.

- **Zero-padding:** The mechanism of adding zeros to the input size limits, allowing to control the spatial size of the output volumes. In particular, it is sometimes desirable

to maintain the spatial size of the input size. This method is also useful for applying a specific step value. If the stated condition is not met for the current values, the zero padding may be a solution because it increases the value of **N**.

- **Activation Functions:**

Activation functions are mathematical equations that determine the output of a neural network. The function is attached to each neuron in the network, and determine whether it should be activated or not, Activation functions also help normalize the output of each neuron to a range between 1 and 0 or between -1 and 1.

These non-linear functions which applied by neurons are added at the output of each convolutional layer and also after fully connected layers. Their goal is to introduce non-linear properties in the CNN. And in order to enable gradient-based optimization methods they must also have some desirable characteristics such as being continuously differentiable. Each of them shows better performance in certain cases and has their own benefits; For example, the results of ReLU in the neural network training several times faster, without making a significant difference to generalization accuracy. Anyway, the design of activation functions and the selection of the most appropriate one in each case are complex issues that are out of the scope of this thesis, the most relevant activation functions are shown in **figure 1.9**.

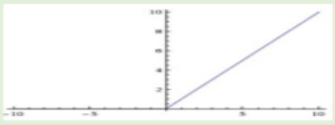
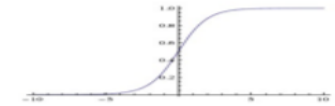
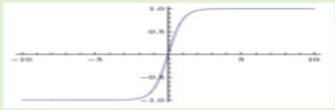
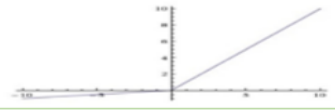
Name	Equation	Plot
Rectified Linear Unit (ReLU)	$f(x) = \begin{cases} 0 & for \quad x < 0 \\ x & for \quad x \geq 0 \end{cases}$ $f(x) = \max(0, x)$	
Sigmoid	$f(x) = \frac{1}{1 + e^{-x}}$	
Hyperbolic tangent (tanh)	$f(x) = \tanh(x)$	
Leaky ReLU	$f(x) = \max(\alpha x, x)$	

Figure 1.9: Summary of the main activation functions [13].

1.3.2.2 Pooling layer:

It is a form of nonlinear reduction. There are many non-linear functions to perform grouping, including maximum grouping is the most common. It divides the input image into a set of non-overlapping rectangles, producing the maximum for each sub-region. The exact location and approximate location of the feature are determined in relation to other features, allowing for a gradual reduction of the spatial size of the representation and sampling down the results in some layers. It also aims to reduce the amount of parameters and calculation in the network, thus also controlling over processing. It is common to periodically insert pooling layer in-between successive convolutional layers in CNN architecture.

The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2. This leads to a reduction of 75% of the inputs, as only one out of four elements is selected. Although max pooling is the most common and used, the pooling layers can also perform other functions, such as average pooling and even L2-norm pooling.

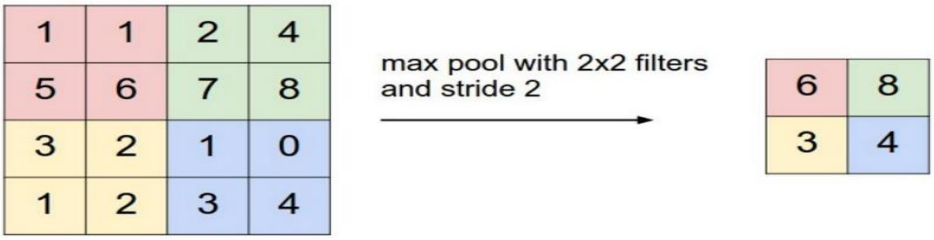


Figure 1.10: Max pooling example. [13]

In this figure, an example of max pooling is represented, showing the operation on a single activation map. A 2x2 filter is being used which is convolved using a stride of 2. The different colors shown correspond to each position of the filter in the input activation map. Then, the maximum value at each location is taken as the output.

1.3.2.3 Fully connected layer:

As explained previously, neurons are connected in a layer fully connected to all activations in the previous layer, the same way in convolutional neural networks. In CNN designs, these layers are usually at the end of the grid, after several convolutional and aggregate layers.

In particular, fully connected layers can be viewed as a subset of convolutional layers where filters have exactly the same size as the input size. Next, the filter can be interpreted as a set of coefficients that multiply all input elements followed by offset bias. In this case, the output size is 1 × 1 as deep as the number of filters, and is thus represented as a column.

1.3.3 The related works:

The 2D-CNN is an area of great research interest due to its wide range applications, and among deep learning algorithms, the 2D-CNN became prominent based on its use in handwriting recognition in the late 1990s[20], and in October 2016 "Ahmed El-Sawy", "Hazem EL-Bakry", and "Mohamed Loey" [21] propose a paper to using CNN to create deep learning recognition system for Arabic handwritten digits recognition.

In particular, since the development of CNNs that achieve good performances on image recognition tasks, great progress has been achieved in learning image data, and when applied to facial recognition, CNNs significantly reduced error recognition rate, that is, with a success rate of 97.6% on 5,600 still images for more than 10 subjects [22].

In addition, CNN uses some features of the visual cortex. One of the most popular uses of this architecture is image classification. For example, Facebook CNN uses auto-tagging algorithms, Amazon to create product recommendations and Google (to search by user images) [23]

By "S.McCloskeyetal" the spectrum was fed to the nasal air flow signal, which was calculated using continuous wavelet transformation (CWT) with Morlet analysis waves [24],to CNN2D where the network had two side layers with subsequent RLU activation layers and a two-dimensional maximum aggregation layer followed by the fully connected layer and the Softmax layer With three external nodes representing the three layers (normal, apnea). The model achieved an average accuracy of 79.8%.

"Chen" et "al". [25] used 2D CNN with leave one out cross validation, which has three input signals (blood oxygen saturation, or onasal airflow, and ribcage and abdomen movements) with one second annotation. A two-dimensional matrix with zero padding was created as input to the network that consisted of two convolution layers, two subsampling layers and a fully-connected layer connected to the output layer with three nodes. The multiclass classification overall accuracy was 79.61%.

1.3.4: The popular CNN architectures

Table 1.2: example of popular CNNs architectures

	LeNet	**AlexNet**	**ZFNet**	**GoogLe-Net**	**VGGNet**	**ResNet**
Input size	32 x 32	128 x 128	128X128	224x224	128 x 128	224 X 224
Kernel size	5 x 5	11 x11	7,11	1, 3,5,7	3 ,5,11	1,3,7
Stride	1	1	1	1,2	1	1,2
Developer	Yann LeCun et al [15], [14]	Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever [16]	Matthew Zeiler and Rob Fergus [16], [14]	Google [16]	Simonyan, Zisserman [16], [14]	Kaiming He [16]
Data set	MNIST	ImageNet	ImageNet	ImageNet	ImageNet	ImageNet CIFAR-10

Nb of parameters	0.060 M	60 M	60 M	4 M	138 M	6.8 M 1.7 M
Nb. Of Layers	7	8	8	22	19	152 110
Error rate	0.8	16.4	11.7	6.7	7.3	3.6 6.43
References	[15], [14]	[16], [14]	[16]	[16], [14]	[16]	[16], [14]

Example 01: Two-Dimension CNN Build "Cifar-10" Image Classifier.

In this employ, they used "CIFAR-10" database, which contains from 80 million tiny image dataset, is a collection of size 32 x 32 pixels [17].

- **Architecture Model :**

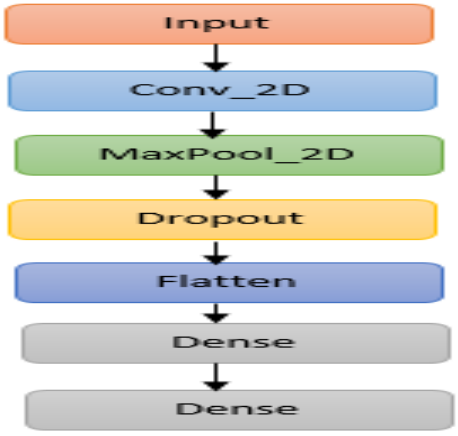


Figure 1.11: Architecture model of 2D-CNN Build "Cifar-10" Image Classifier

- **Input Image:**

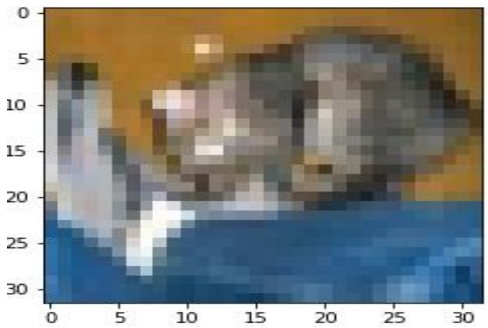


Figure 1.12: "CIFAR-10" input image

- **Convolution keras model:**

Firstly, they used in the conventional network three types of layers contains the following:

- **Input Layer:**
  - Input image size: 32 x 32
  - Filter size: 3 x 3
- **Hidden layer:**

Convolutional layer:

  - Filters: 32.
  - Filter size: 3 x 3
  - Padding: 'same'
  - Activation function: ReLu

MaxPool layer:

  - Pool Size: 2 x2
  - Strides: 2
  - Padding:' valid'
  - Dropout: Dropout the overfitting with 0.5

- **Output layer:**

Flatten layer:

Transform a two-dimensional matrix of features to vector, that fed fully connected neural network classifier.

Dense layer:

- Activation: ReLu
- Softmax activation function to get the output

Table 1.3: Parameters number of Convolution keras model

Layer type	Nb. Of parameters
2D-Conv _1	896
2D-Conv_2	9243
2D-Max_pooling	0
Dropout	0
Flatten	0
Dense	1048704
Dense_1	1290

- Total parameters: 1060138.
- Trainable parameters: 1060138.
- Non trainable parameters: 0.
- In this model, we train on 50000 samples and validate on 10000 samples.

- **Training and Validation accuracy values:**

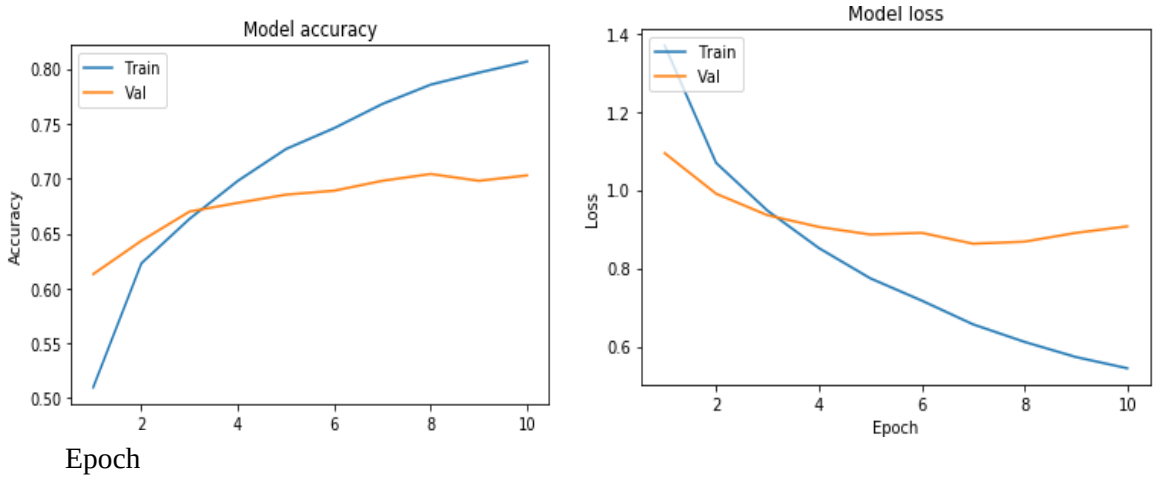


Figure 1.13: Model Loss/ Accuracy of Convolution keras model

- Validation accuracy equal: 0.70

- Validation loss values equal: 0.90

- **Explaining the graphs:**

  . **Model accuracy:** the validation accuracy less than the training set, that's been the model is overfitting.

  . **With the loss :** you can verify that validation loss has saturated after three epochs that's mean the model is not learning, but we kept training our model, and training error has, you know the training loss kept decreasing, but the validation loss decreasing, this says that our model is overfitting: that's mean we had increased the complexity of our model ..

- **Visualization**:

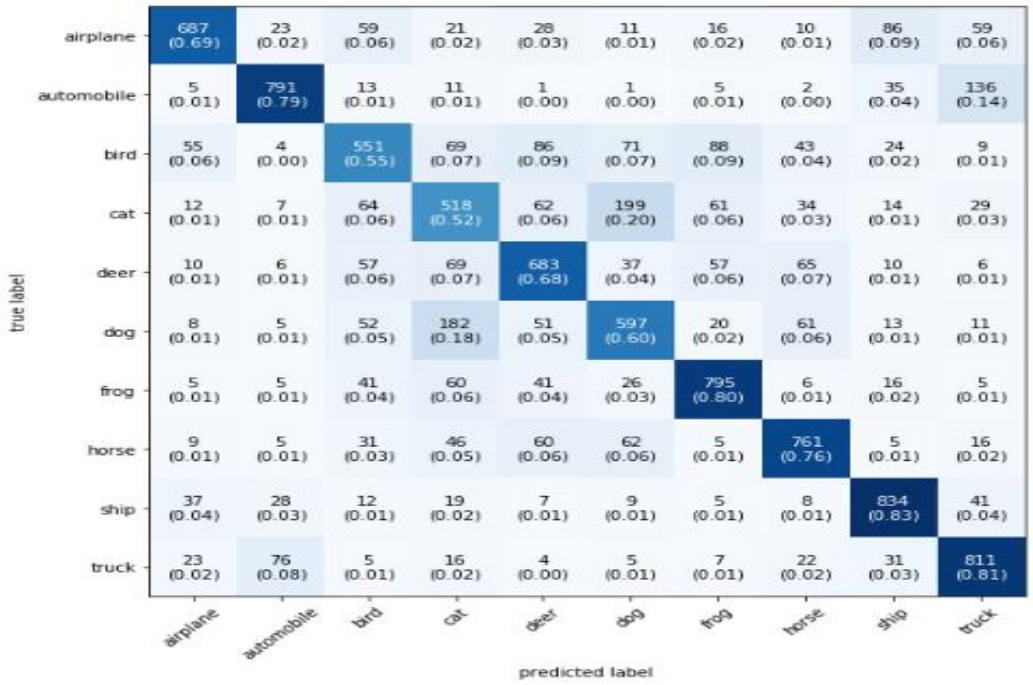


Figure 1.14: Confusion Matrix of Convolution keras model

For these problems we proposed this example which solve the most problems and given a high accuracy.

Example 02: Two-Dimension CNN for “MNIST” Digit Recognition.

In this work, they used "MNIST" database contains about 60000 training images and 10000 test images of the size 28 x 28 pixels [18].

- **Model Architecture:**

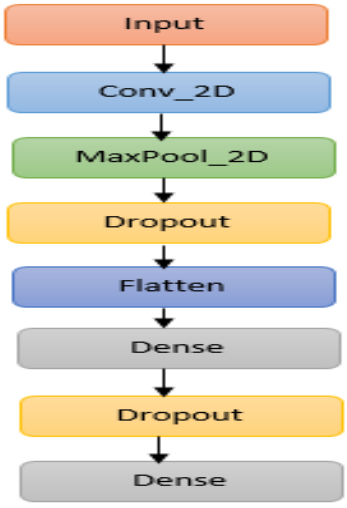


Figure 1.15: Architecture Model of two-Dimension CNN for “MNIST” Digit Recognition.

- **Input Image:**

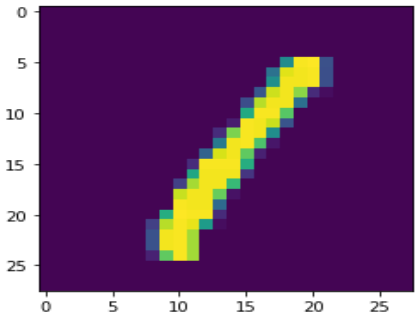


Figure 1.16: The **"**MNIST" input image

Table 1.4:2D CNN Architecture for "MNIST" Digit Recognition

Layer type	Kernel size	Stride	Input size	Activation function
2D-Conv	3x 3	1	26 x 26	ReLu
2D-Conv_1	3 x 3	1	24 x 24	ReLu
2D-Max_pooling	2 x 2	1	12 x 12	
Dropout		1	12 X 12	
Flatten		1	9216	
Dense		1	128	ReLu
Dropout_1		1	128	
Dense		1	10	Softmax

Table 1.5: Parameters number of 2D CNN for "MNIST" Digit Recognition

Layer type	Nb. Of parameters
2D-Conv	320
2D-Conv_1	18496
2D-Max_pooling	0
Dropout	0
Flatten	0
Dense	1179776
Dropout_1	0

- Total parameters: 1199882
- Trainable parameters: 1199882
- Non trainable parameters: 0
- In this model, we train on 60000 samples and validate on 10000 samples

- **Training and Validation accuracy values:**

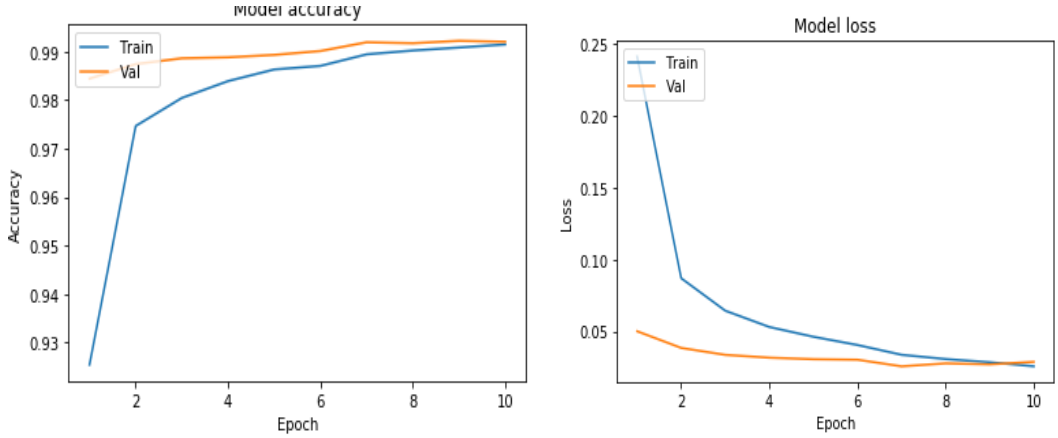


Figure 1.17: Model Loss/ Accuracy of 2D CNN for "MNIST" Digit Recognition

- The validation accuracy equal 0.99.

- **Explaining the graphs:**

In model accuracy: the validation and training is almost equal
In the model loss: training and validation is almost equal.

Therefore, the model is almost stable and not having overfitting, we can say that's a perfect model.

- **Visualization:**

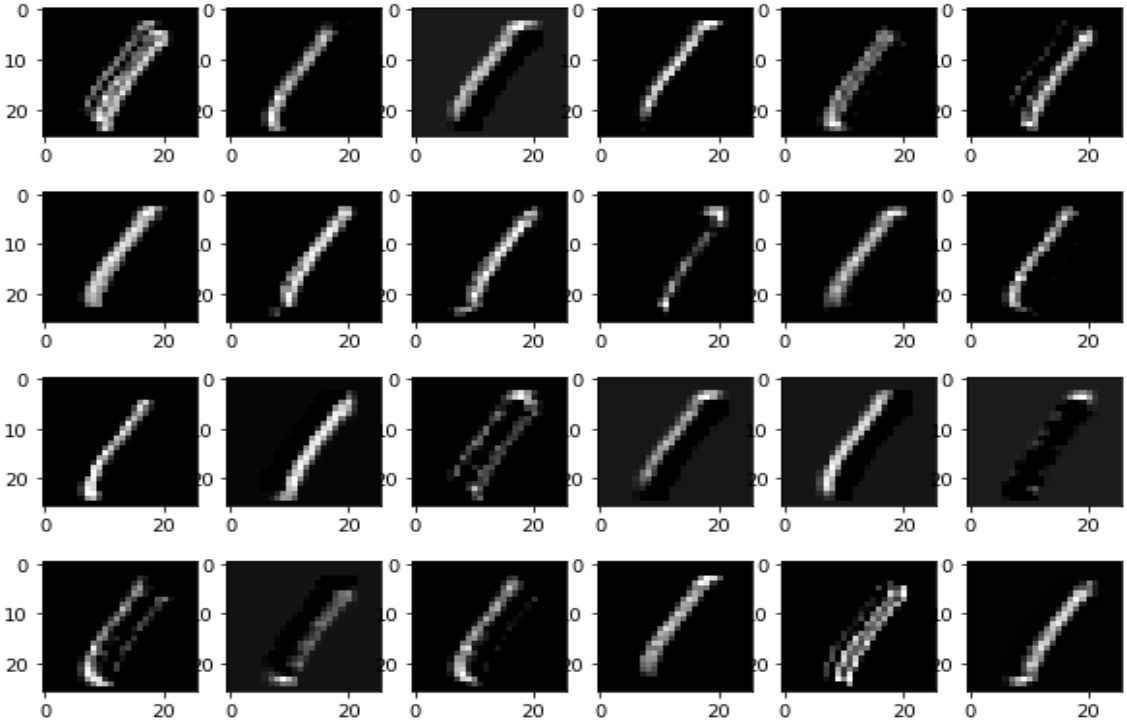


Figure 1.18: Output image of 2D CNN for "MNIST" Digit Recognition model

In the previous example, we used 2D-CNN on minist and CIFAR-10 dataset, now we explain 2D-CNN on ECG classification model.

Example 3: Two-dimensional CNN on ECG classification model

In this work they used the 2017 PhysioNet/CinC Challenge which aims to encourage the development of classification algorithms , from a single short ECG lead recording (between 30 s and 60 s in length), whether the recording shows a normal sinus rhythm, atrial fibrillation (AF), an alternative rhythm, or it is too noisy to be classified [19].

- **Preprocessing:**

In the beginning, they converted ECG signal to spectrogram and used it as an input, and they calculated the image dimensions using spectrogram function, which converts ECGs signal into spectrogram.

Electrocardiograms are representations of the periodic cardiac cycle. Which contained useful information in both the time and the frequency domain. We can extract

frequency information by applying a Fourier transform and normalizing the frequency contributions to obtain the power spectrum of the ECG shown below:

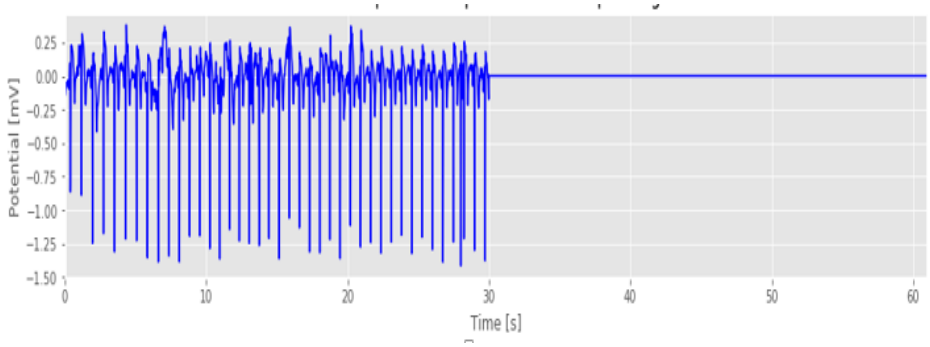


Figure 1.19: ECG sequence with zero padding

- **Spectrogram function**

```
def spectrogram(data, nperseg=64, noverlap=32, log_spectrogram = True):
    fs = 300
    f, t, Sxx = signal.spectrogram(data, fs=fs, nperseg=nperseg, noverlap=noverlap)
    Sxx = np.transpose(Sxx,[0,2,1])
    if log_spectrogram:
        Sxx = abs(Sxx) # Make sure, all values are positive before taking log
        mask = Sxx > 0 # We dont want to take the log of zero
        Sxx[mask] = np.log(Sxx[mask])
    return f, t, Sxx
```

Figure 1.20: Spectrogram Function

- **Model Architecture:**

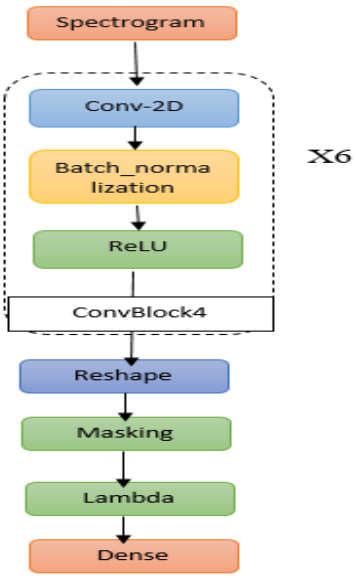


Figure 1.21: Architecture of the proposed network in example 03

Now, we present a simple explanation on how to convert time series into spectrogram to preserve both time and frequeny content of the signal, we can combine between the PDS and the time series. A spectrogram returns the time-dependent Fourier transform for a sequence computed using a sliding window. This form of the Fourier transform, also known as the short-time Fourier transform (STFT), has numerous applications in speech, sonar, and radar processing. The spectrogram of a sequence is the magnitude of the time-dependent Fourier transform versus time. To optimize the dynamic range of frequency, it is also useful to apply a logarithmic transform. In fact, other reports showed that the logarithmic transform considerably increases the classification accuracy. Here is an illustration of the effect of the logarithmic transform.

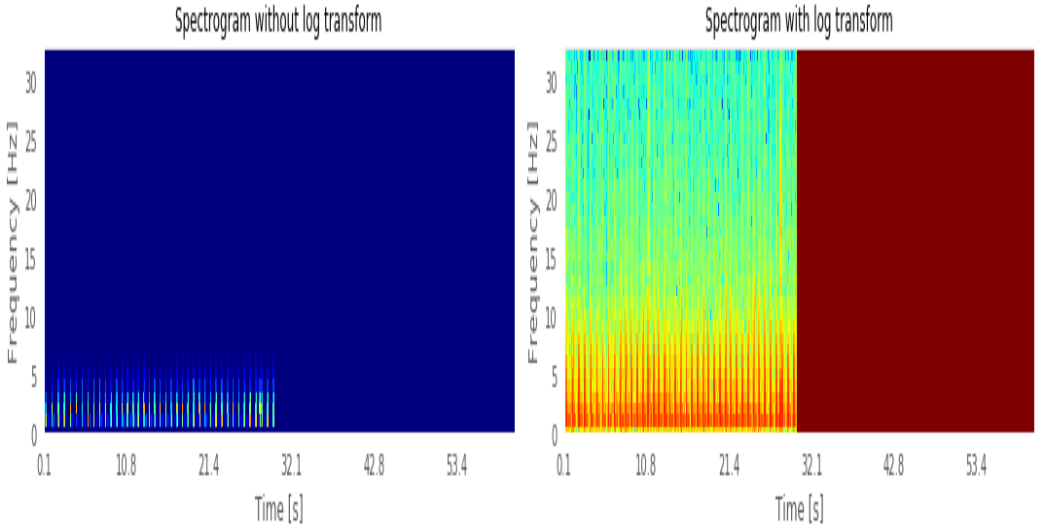


Figure 1.22: Spectrogram without/with log transform

In this exampel, they used spectrogram with long transform shown under, and training a Two-Dimension convolution neural network for a classification of ECG

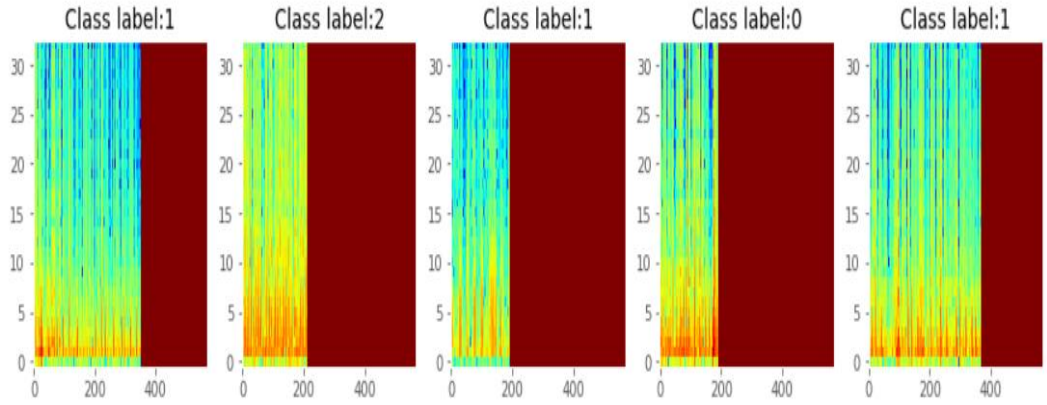


Figure 1.23: Classification Electrocardiogram

Shown in the table under arrhythmia classes**:**

Table 1.6: Table of arrhythmia classes

N°	Category	Annotations
0	N	Normal sinus rythma
1	A	Atrial fibrillation
2	O	Other rythma
3	~	Too noisy to be classified

- **Convolution keras model:**

  After the spectrogram conversion, they used six groups of four layers, in all the convolution layers, they are applying Two-Dimension convolution in the same number of filters of size 3 x 3.

  The number of filters is initially a set of 32 for the first three convolution layers, but increases by 32 in the last layer of each convolution block, the first three convolution layers applies size 1, but in the last layer, it keeps a stride of two.

  Using the default setting for the spectrogram conversion (sequence length = 18286, hanning window =64, overlap=32), are input dimensions of the network proposed are 570 x 33 x 1.

- **Training:**

  Illustrates the network architecture proposed in this work they used cross-entropy loss and employed the Adam optimizer. The batch size was set to 32.

  - Total parameters: 3554532
  - Trainable parameters: 3548772
  - Non-trainable parameters: 5760

In the last, the accuracies were save in the file "history.csv".

1.4 Conclusion

Although the significant power of 2D-CNNs that was successfully contributed it the main in different engineering applications. However, the data representation as 2D map for signal application remains still a big challenge to ensure its performance. To address this issue, 1D-CNNs have recently been proposed and immediately achieved the state-of-the-art performance levels in several applications such as personalized biomedical data classification and early diagnosis, structural health monitoring, anomaly detection and identification in power electronics and motor-fault detection. Another major advantage is that a real-time and low-cost hardware implementation is feasible due to the simple and compact configuration of 1D-CNNs that perform only 1D convolutions (scalar multiplications and additions). In the next section, we provide a comprehensive review of the general engineering principles and 1D-CNN along with major engineering applications.

CHAPTER 02

1D CNNs

Chapter 2: One-dimensional Convolution Neural Networks

2.1 Introduction

The Deep CNNs presented in the previous section have the ability to learn complex objects and patterns provided that were trained in a large visual database with truth labels. Thanks to proper training, this has made it the essential tool for various engineering applications of 2D signals such as images and video frames. However, this does not seem to be a decent option for application in many applications via 1D signal, especially when training data are scarce or application-specific. To address this problem, 1D-CNNs were recently proposed and immediately achieved a good performance levels in many applications such as personal biomedical data classification, early diagnosis, structural health monitoring, anomalies detection and identification in power electronics and motors - fault detection. The advantage of the latter is that real-time, low-cost hardware implementation is possible due to the simple and compact configuration of 1D-CNN networks that perform only 1D correlation (multiplication and numerical additions).

1D-CNNs useful, so its 2D counterparts prefer to handle 1D signals for many reasons, especially since the computational complexity of 1D-CNN networks is much lower than 2D-CNN. Instead of matrix operations, FN and BP in 1D require simple matrix operations. Recent studies have indicated that 1D-CNN networks have relatively shallow structures (i.e., a few hidden and neuronal layers) capable of learning difficult tasks involving 1D signals. On the other hand, 2D-CNN usually requires deeper architectures to handle these tasks. Obviously, shallow structures are much easier to train and implement; In addition, any CPU implementation on a standard PC is relatively quick and possible to train 1D-CNN compressed with a few hidden layers (such as 2 or less) and neurons (eg<50). Similar to the training of deep CNN 2 networks that require special hardware setup (such as cloud computing or GPU farms). Due to their low computational requirements, 1D embedded CNNs are well suited for real-time, low-cost applications especially on mobile or handheld devices.

Compact 1D-CNNs have shown superior performance on applications that have data with limited markers and high signal changes acquired from different sources (for example, patient ECGs, civil, mechanical and aerospace structures, high-power circuits, power motors or motors, etc.).

2.3 The related works

Some authors have adapted the concept by using a one-dimensional CNN (1D-CNN) network in several application areas, for example "Urtnasan et al."[29], analyzing the signal from one single lead electrocardiogram using 1D-CNN with a suspension method (a 63-topic training group, a 19-topic test group). The mesh was composed of different sizes of wrapping, activation and assembly layers, followed by leakage. The highest accuracy (96%) was achieved using a six-layer CNN network using the F1 score as the highlight parameter.

“Urtnasan et al.”[30] also used the 1D-CNN for multiclass classification (normal, apnea and hyperpnea). The input of the network was 10 s long contained and 2000 samples. The six-layer CNN achieved 90.8% mean accuracy among the classes.

“Dey et al.”[31] also employed a1D-CNNto analyze one minute segments of a single lead ECG signal, each with 6000 samples. Unlike other implementations, it used only convolution and fully connected layers. The pooling was performed using convolutional pooling. Authors tested the model with different training: test dataset ratios from 50:50 to 20:80 where 50:50 had the best average accuracy of 98.91%.

In addition, binary classification (either apnea or normal) was performed based on nasal airflow analysis by “Haider et al” [32]. With 1D-CNN class and balanced data set. The model achieved an accuracy rate of 75%.

“Haider et al” tested one-dimensional CNN with three input signals [33], analyzing the nasal flow, the abdominal and thoracic plethysmography signals using hold-out methods with 75% training and 25% test datasets. It was verified that the performance of the model with three channels was better than any single or double channels model, with an average accuracy of 83.5%.

“Mc Closkey et al.”[34] have also performed a multilayer classification (normal, apnea), by analyzing the nasal airflow signal, which was normalized with epochs of 30 seconds, with an input size of 960 samples. The output was three nodes representing three chapters. 1D-CNN achieved an average accuracy of 77.6%.

2.2 1D-CNN architecture

In 1D-CNN networks, two different types of layers are proposed. The first is the so-called "CNN layers" where there are two-dimensional bonds and a sub-sample (aggregation), and second, fully connected layers that correspond to typical MLcept Perceptron layers (hence called "MLP layers"). The 1D-CNN configuration is configured by hyperlink parameters.

In the 1D-CNN proposed sample there are three and 2 hidden CNN and MLP layers, respectively. Additionally, the size of the filter (nucleus) in each CNN layer (as shown in **figure 2.1**), is 41 in all hidden CNN layers; The sub-factor of the sample in each layer of CNN networks shown in **figure 2.1** is 4, with careful selection of collection and activation functions.

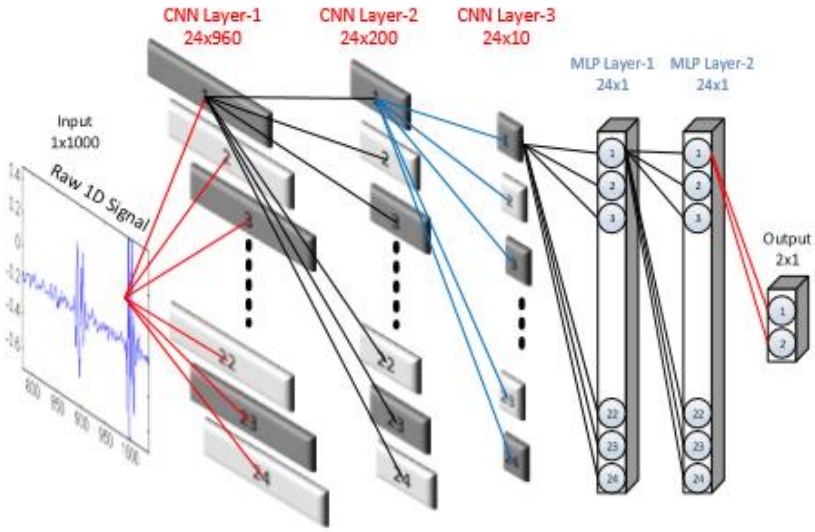


Figure 2.1: A sample 1D CNN configuration with 3 CNN and 2 MLP layers [26].

Shown in **figure 2.2**. Three consecutive CNN layers of 1D-CNN. In this sample diagram, the 1D filter kernels have size 3 and the sub-sampling factor is two where the kth neuron in the hidden CNN layer, which is passed through the activation function, followed by the sub-sampling process. In the end, CNN layers process raw 1D data and "learn to extract" such features that can be used in the classification task performed by MLP layers. Therefore, both extraction and feature classification operations are combined into a single process that can be optimized to maximize classification performance. This is the main advantage of 1D-CNN

networks which can also provide low computational complexity because the only costly process is a series of 1D correlations that are only linear weighted amounts of two 1D arrays. Since the only operation with a significant cost is a sequence of 1D convolutions which are simply linear weighted sums of two 1D arrays. Such a linear operation during the Forward and Back-Propagation operations can effectively be executed in parallel.

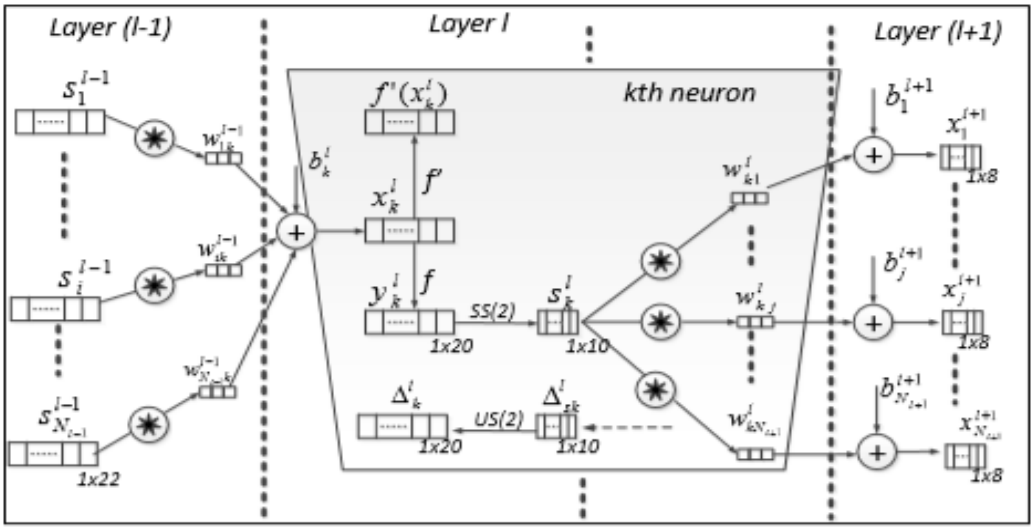


Figure 2.2: Three consecutive hidden CNN layers of a 1D-CNN [26].

2.2.1 Forward- and Back-Propagation in CNN-layers:

1D forward propagation (1D-FP) in each CNN-layer is expressed as follows:

$$X_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} conv1D\left(w_{ik}^{l-1}, s_i^{l-1}\right) \qquad (2.1)$$

Equation 2.1: One- Dimension forward propagation formula

Where x_k^l is defined as the input, b_k^l is defined as the bias of the k^{th}neuron at layer l, $S_i^{l-1}$is the output of the i^{th}neuron at layer $l-1$, $w_{ik}^{l-1}$is the kernel from the i^{th} neuron at layer $l-1$ to the k^{th} neuron at layer l. $conv1$ (.,.) is used to perform 'in-valid' 1D convolution without zero-padding. Therefore, the dimension of the input array,x_k^l is less than the dimension of the output arraysS_i^{l-1}. The intermediate output, y_k^l, can be expressed by passing the input x_k^l through the activation function, (.), as:

$$y_k^l = f\left(x_k^l\right) \wedge s_k^l = y_k^l \downarrow ss \qquad (2.2)$$

Equation 2.2: Calculating the intermediate output

Where s_k^l stands for the output of the k^{th} neuron of the layer, l, and "$\downarrow ss$" represents the down-sampling operation with a scalar factor, ss. The back-propagation (BP) algorithm can be summarized as follows. Back propagating the error starts from the output MLP-layer. Assume l = 1 for the input layer and $l = L$ for the output layer. Let $N_L$be the number of classes in the database; then, for an input vector p, and its target and output vectors, $t^p$and $[y_1^L, \dots, y_{N_L}^L]'$, respectively. With that, in the output layer for the input p; the mean-squared error (MSE), E_p, can be expressed as follows:

$$E_p = MSE\left(t^p, \left[y_1^L, \dots, y_{N_L}^L\right]'\right) = \sum_{i=1}^{N_L}(y_i^L - t_i^P)^2 \qquad (2.3)$$

Equation 2.3: Back propagation formula

To find the derivative of E_p by each network parameter, the delta error $\Delta_k^l = \frac{\partial E}{\partial x_k^l}$ should be computed. Specifically, for updating the bias of that neuron and all weights of the neurons in the preceding layer, one can use the chain-rule of derivatives as:

$$\frac{\partial E}{\partial w_{ik}^{l-1}} = \Delta_k^l y_i^{l-1} \quad \text{and} \quad \frac{\partial E}{\partial b_k^l} = \Delta_k^l \qquad (2.4)$$

Equation 2.4: Calculating the chain-rule of derivatives bias and weight

So, from the first MLP layer to the last CNN layer, the regular (scalar) BP is simply performed as:

$$\frac{\partial E}{\partial s_k^l} = \Delta s_k^l = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x_i^{l+1}} \frac{\partial x_i^{l+1}}{\partial s_k^l} = \sum_{i=1}^{N_{l+1}} \Delta_i^{l+1} w_{ki}^l \qquad (2.5)$$

Equation 2.5: Calculating the regular (scalar) BP

Once the first BP is performed from the next layer, l+1, to the current layer, l, then one can carry on the BP to the input delta of the CNN layer l,Δ_k^l. Let zero order up-sampled map be: $s_k^l = up\left(s_k^l\right)$, then the delta error can be expressed as follows:

$$\Delta_k^l = \frac{\partial E}{\partial y_k^l}\frac{\partial y_k^l}{\partial x_k^l} = \frac{\partial E}{\partial us_k^l}\frac{\partial us_k^l}{\partial y_k^l} f'\left(x_k^l\right) = up\left(\Delta s_k^l\right)\beta f'\left(x_k^l\right)$$

Equation 2.6: Calculating the delta error

Where$\beta = (ss)^{-1}$. Then, the BP of the delta error ($\Delta s_k^l \leftarrow \Delta_i^{l+1}$) can be expressed as:

(2.7)

$$\Delta s_k^l = \sum_{i=1}^{N_{l+1}} conv1Dz\left(\Delta_i^{l+1}, rev\left(w_{ki}^l\right)\right)$$

Equation 2.7: Calculating the BP of the delta error

Where (.) is used to reverse the array and $conv1(.,.)$ is used to perform full 1D convolution with zero-padding. The weight and bias sensitivities can be expressed as follows:

$$\frac{\partial E}{\partial w_{ik}^l} = conv1D\frac{\left(s_k^l, \Delta_i^{l+1}\right) \wedge \partial E}{\partial b_K^l} = \sum_n \Delta_k^l(n) \qquad (2.8)$$

Equation 2.8: Calculating the weight and bias sensitivities

When the weight and bias sensitivities are computed, they can be used to update biases and weights with the learning factor, ε as:

$$w_{ik}^{l-1}(t+1) = w_{ik}^{l-1}(t) - \varepsilon\frac{\partial E}{\partial w_{ik}^{l-1}} \wedge b_k^l(t+1) = b_k^l(t) - \varepsilon\frac{\partial E}{\partial b_k^l} \qquad (2.9)$$

Equation 2.9: calculating the Updates biases and weights with the learning factor,

The forward and back-propagation in hidden CNN layers are illustrated in **figure 7**. The output sensitivity of the k^{th} neuron at the CNN layer l, Δs_k^l, is formed by back-propagating all the delta errors, Δ_i^{l+1}, at the next layer, $\boldsymbol{l}$ +1ala , by using Eq. (8), while the forward and back-propagation between the last hidden CNN layer and the first hidden MLP layer are summarized in **figure 2.3**.

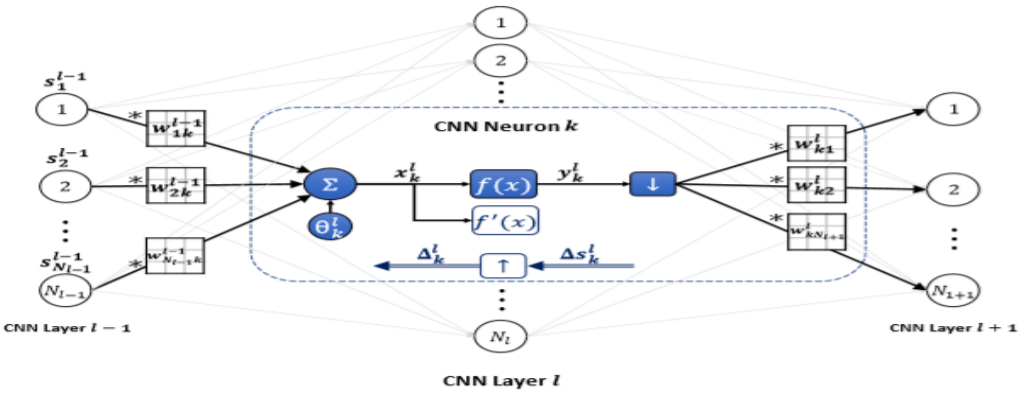


Figure 2.3: Forward and back-propagation in hidden CNN layers [26].

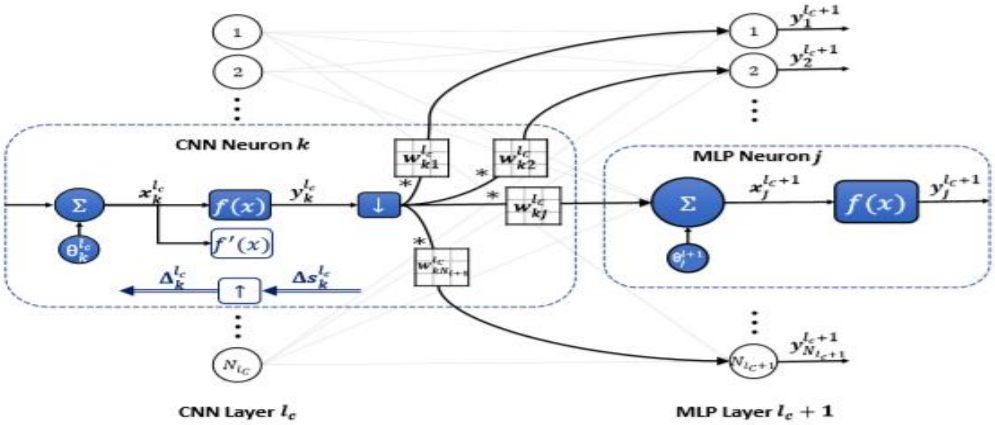


Figure 2.4: Forward and back-propagation between the last hidden CNN layer and the first hidden MLP layer [26]

Consequently, the iterative flow of the BP for the 1D raw signals in the training set can be stated as follows:

1) Initialize weights and biases (e.g., randomly, ~U (-0.1, 0.1)) of the network.
2) For each BP iteration **DO**:

   For each PCG beat in the dataset, DO:

- **FP**: Forward propagate from the input layer to the output layer to find outputs of each neuron at each layer, sil, $\forall i \in$ [1,], and $\forall l \in$ [1,].
- **BP**: Compute delta error at the output layer and back-propagate it to first hidden layer to compute the delta errors, Δ, $\forall k \in$ [1,] and $\forall l \in$ [1,].
- **PP**: Post-process to compute the weight and bias sensitivities.
- **Update**: Update the weights and biases by the (accumulation of) sensitivities scaled with the learning factor, ε.

2.2.2 ECG analysis based 1D-CNNs

Exampel 01:

In this work, they used PhysioNet MIT-BIH databases. The MIT-BIH dataset consist of ECG from 47 different subjects records at the sampling rate of 360 Hz [34].

Table 2.1: Table of arrhythmia classes

Category	Annotations
N	• Normal • Left/Right bundle branch block • Atrial escape • Nodal escape
S	• Atrial premature • Aberrant atrial premature • Nodal premature • Supra-ventricular premature
V	• Premature ventricular contraction • Ventricular escape
F	• Fusion of ventricular and normal
Q	• Paced • Fusion of paced and normal • Unclassifiable

- **Preprocessing:**

  In this exampel, the Input is ECG signal

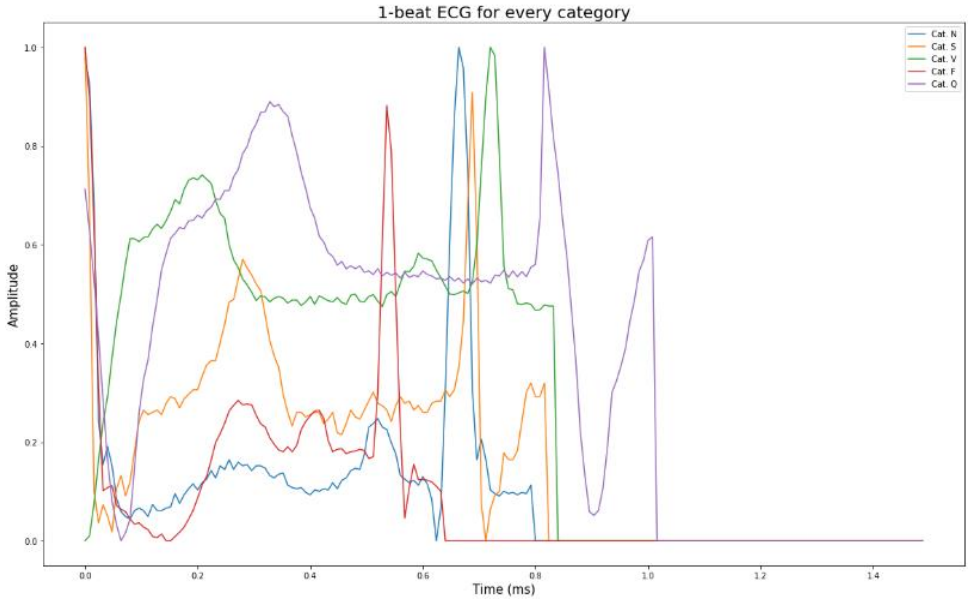


Figure 2.5 ECG signal for every category

To train properly the model, we should augment all data to the same level. Nevertheless, for a first try, we will just augment the smallest class to the same level as class 1. With that, we will be able to have a test set of around 5x800 observations.

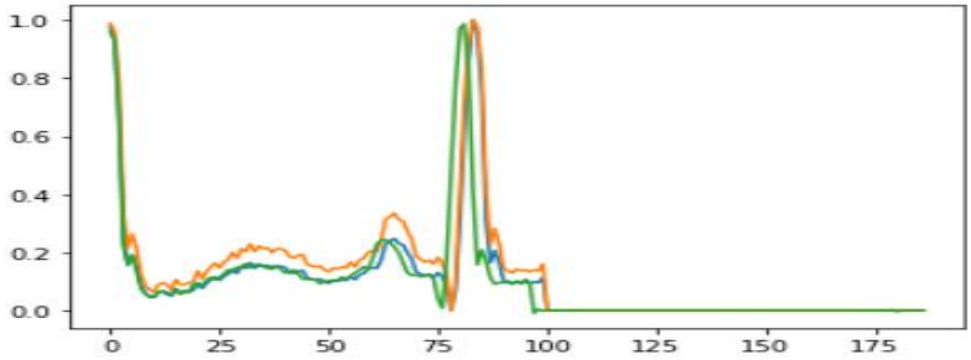


Figure 2.6: ECG signal

- **The Architecture of the proposed network :**

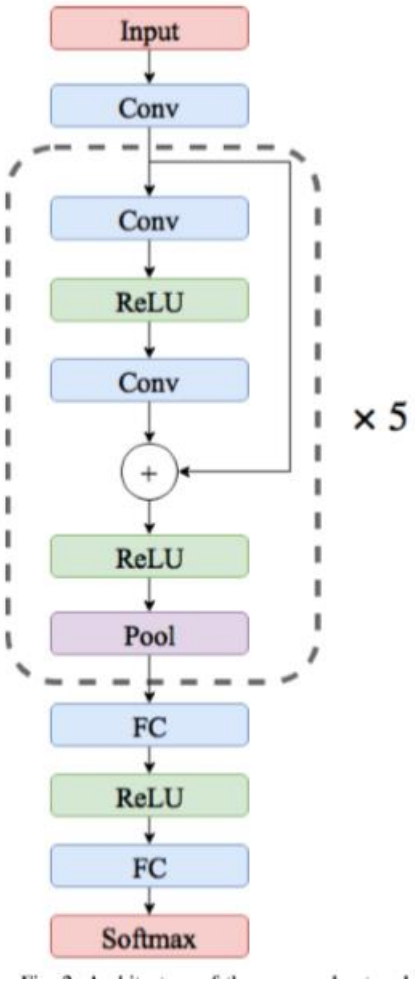


Figure 2.7: Proposed architecture model in example 01

- Total parameters: 50853
- Trainable parameters: 50853
- Non-trainable parameters: 0

- **Training the Arrhythmia classifier:**

In this exampel training a 1D convolutional neural network for classification of ECG beat types on the MIT-BIH dataset.

In the architecture of the proposed network. Extracted beats, are used as input all convolution layers are applying 1D convolution and each have 32 kernels of size 5.we also use max pooling of size 5 and stride 2 in all pooling layers, the network consist of

five blocks followed by two fully-connected layers with 32 neurons each and a Softmax layer to predict output class.

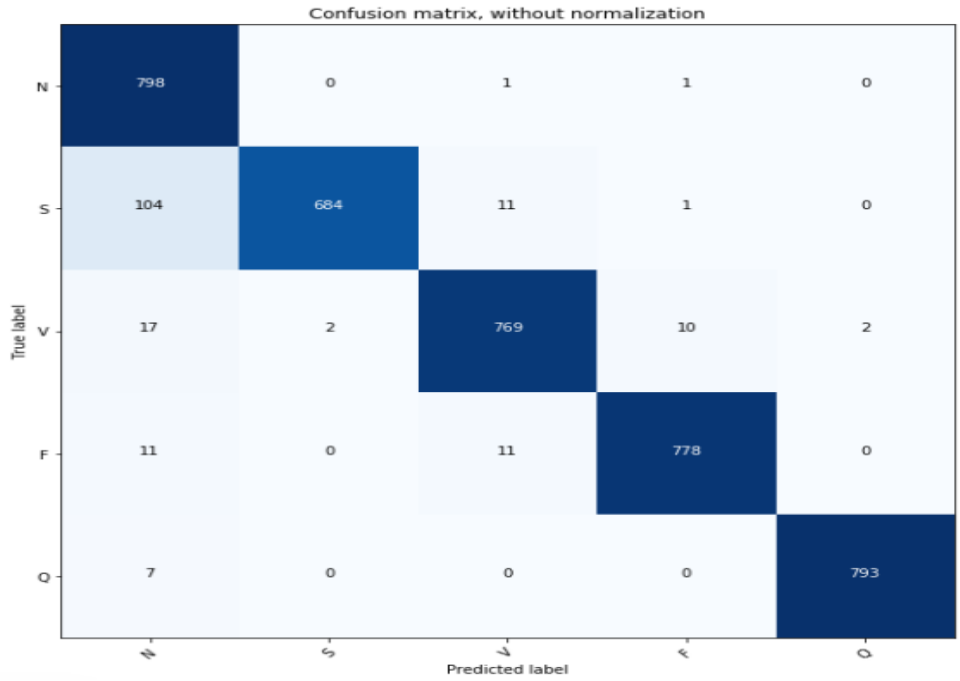


Figure 2.8: Confusion matrix for heartbeat classification on the test set

2.4 The Comparison between 1D-CNN and 2D-CNN:

Table 2.2: Comparative table between 1D-CNN and 2D-CNN

	2D-CNN	**1D-CNN**	**Reference**
Architectures (Nb Layer & neurons)	More Nb. Of hidden layers and neurons	Small number of hidden layers and neurons	[7]
Nb. Of Conv Layers	5	3	[7]
MLP	3	2	[7]
Input size	256 x 256	820	[28]
Kernel size	4	3	[7]
Sub-Sampling	3	2	[7]
Complexity	More	Low	[28]
Cost	High-cost	Low-cost	[7]
Form of kernel and feature map	Matrices	Arrays	[7]
Classification performance	Lower accuracy	Higher accuracy	[28]

2.5 Conclusion

Despite the impressive results of 1D CNNs, the main limitation is actually common for conventional CNN networks and ANN networks in general are homogeneous (the same type of neurons in the entire network) and are based solely on the linear neuron model of 1950. Therefore, many models have been proposed to tackling this issue, where made to omit the activation function such as: Gabor function, wavelet function, optimization function and local binary pattern. This latter is a straightforward function but powerful descriptor for spatial features, which can lessen the workload of CNNs and improve the classification accuracy. In

order to make full use of the feature extraction capability of CNNs and the discrimination of LBP features, a novel classification method combining dual-channel CNNs and LBP is proposed.

Specifically, a one-dimensional CNN (1D-CNN) is adopted to process original multivariate data to extract hierarchical spectral features and the LBCNN is applied to process signal through LBP filter instead of convolution filter of CNN. Then, the concatenation of two fully connected layers from the both CNNs, which fused features, is fed into a Softmax classifier to complete the classification, and we will show more details in the next section.

CHAPTER 03

LBCNN

3.1 Introduction

Since the training of large CNN networks requires a lot of time and energy, and because the application of energy saving is very important, some researchers have reduced the complexities of training to the CNN by developing modern improvement techniques that are heuristic or metaheuristic. These optimization techniques have been applied to solve any problems in research in science, engineering and even industry. However, metaheuristic research is rarely done to improve the DL method. One work is the combination of genetic algorithm (GA) and CNN, proposed by You and Pu [35]. Their model selects the CNN characteristic by the process of recombination and mutation on GA, in which the model of CNN exists as individual in the algorithm of GA.

Research continued until the researchers came up with new technologies. In 2007, Peijin Ji et al [36] presented the partial Gabor feature-based method approach which based on Gabor filter banks were replaced with Gabor filters specific CNN granules. to reduce the dimension, computation and redundancy of the feature, therefore, much fewer memory and computation was involved. And in 2016 Felix Juefei-Xu et al presented an alternative approach to reducing the computational complexity of CNNs while performing as well as standard CNNs. Propose local binary convolution (LBC), an efficient alternative to convolutional layers in standard convolutional neural networks (CNN). and the design principles of LBC are motivated by local binary patterns (LBP). Empirically, Juefei-Xu et al. [37] proposed a hybrid combination of fixed and learned weights introducing a local binary convolutional layer (LBC). The authors state that CNNs with LBC layers, called local binary convolutional networks (LBCNN) have lower model complexity and are less prone to over-fitting. The methodology achieved best performance than regular CNNs in five datasets: MNIST, SVHN, Cifar-10 and ImageNet, face recognition but the authors have not tested on signal datasets.

This chapter contains the implementation and experimentation of our approach which is explained in the previous chapter, and the main contributions of this work is summarized as follows:

• We use an alternative CNN architecture towards ECG based biometric that can be trained end-to-end on ECG signal without using either fiducial or non-fiducial points.

• Our network uses significantly fewer parameters and has a lower computational cost when compared to the other standard CNN networks.

3.2 Local binary patterns (LBP)

The local binary pattern LBP was proposed by Ojala et al [39], who was firstly defined as a descriptor texture for image processing task. Mathematically, LBP is a simple operator, but is a powerful descriptor which can detect a local and global feature that may be used to identify a specific pattern from image such as edge, color intensity, face, shadow and so on.

LBP code only depends on the difference values between the center pixel and neighboring pixels, which possesses 256 patterns to reflect various texture information. LBP code maps the gray value of any pixel in the image to LBP code, which can be used to obtain an LBP histogram of an image through histogram statistics to describe the texture features of the image.

The LBP is a powerful non parametric operator for the description of local image features and has been proved to be rotation invariant and translation invariant. Given a center pixel (xc, yc), an ordered binary set defined as LBP is obtained by comparing the gray value of the center pixel (xc, yc) with the pixels of its eight neighbors. Thus, the LBP code is expressed as the decimalized form of an octet binary number:

$$\mathrm{LBP}(x_c, y_C) = \sum_{n=0}^{7} S(i_n - i_C)2^n \qquad (3.1)$$

Equation 3.1: Calculating Local binary pattern [38]

Where i_C represents the gray value of the center pixel (x_c, y_C), and *in* is the gray value of the pixels of its eight neighbors. LBP code has been proved to be invariant to any monotonous gray level transformation, and the local neighborhood binary code remains unchanged after transformation.

$$S(i_n - i_C) = f(x) = \begin{cases} 1 i_n - i_C \geq 0 \\ 0 i_n - i_C < 0 \end{cases} \qquad (3.2)$$

Equation 3.2: Calculating Local neighborhood binary code [38]

Example 01:

The example illustrates an LBP calculation with a process of reducing eight neighbors:

- **Figure 3.1(a):** shows a 3 × 3 window; difference values are calculated between the gray value of the center pixel and the gray values of neighboring pixels. As shown in **Equation 3.2***, the transformation function *S* resets the difference value to "1" when the difference value is greater than 0, and "0" otherwise.
- **Figure 3.1(b):** the binary value obtained with the corresponding binary code (LBP)2 = 10,011,100.
- **Figure 3.1(c):** shows the code set of binary codes, that is, a weighted coefficient set of binary codes. Then, the corresponding code (LBP) = 156.

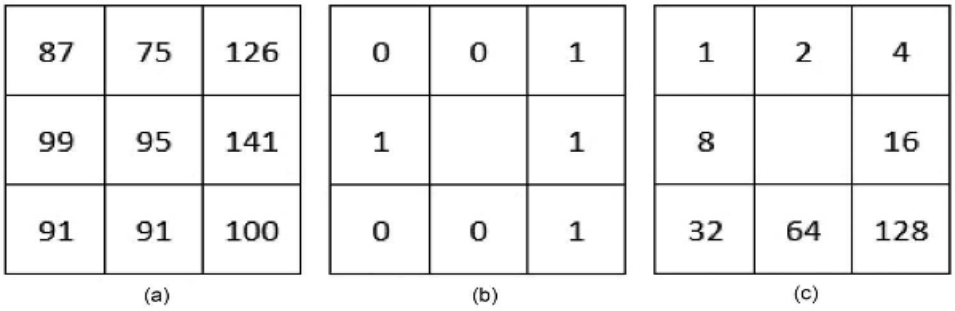


Figure 3.1: Example of LBP calculation. (a) Gray-scale map, (b) binary value set, and (c) code set [40].

3.2.2 Applications domain of LBP

LBP was originally proposed for texture analysis by T,Ojala et al in 1996 [39], and it yielded impressive results in image processing and computer vision. As a non-parametric method, LBP efficiently summarizes local image structures by comparing each pixel with its neighboring pixels, This is presented by T,Ojala et al in 2002 [41]. It was also used by T.Ahonen, A. Hadid, D.P. Huijsmans et al [42] , [43] in many applications, most notably facial image analysis, image and video retrieval [44] , [45].

W.Ali, L. Nanni and others used it in environmental modeling [46],[47], LBP was used in visual examination [48] - [49], movement analysis by M.Heikkilä, V. Kellokumpu, et al [50],[51], biomedical and atmospheric image analysis by A.Oliver, S.Kluckner et al [52] ,[53] and Remote sensing [54], In addition LBP has been exploited for facial representation in different tasks containing face detection [42], [55]-[56], face recognition [57]-[58], facial expression analysis [59]-[60], demographic (gender, race, age, etc.).....

3.3 Local binary convolution neural network (LBCNN)

Although the outstanding results that achieved in many applications real world, many authors have been arguing their complexity architecture such as the training network time consuming, heavily learning parameter network and so on [61, 67 , 68, 69, 70, 71, 72, 73 , 74,75]. Recent works has suggested a new alternative representation of CNNs to reduce the parameter's' network and more suitable for real time application. Among of these works, LBCNN have been recently introduced by Juefei-Xu et [61]. they made a new convolution filter to get a good approximation of a conventional CNN layer. Thus, is omitted by a fixed sparse initialized layer which is generated through local binary operation and then fed to the ReLu activation layer. The output layer is considered as learnable 1×1 convolution filter. So, the LBCCN is specified by two common features:

1) The rate of sparsity is referred to the percentage of non-zero value weights of binary convolutional layer and computed as the following formula:

$$sparsity = \frac{numberofnon-zeroweights}{\sum ofweights} \quad (3.3)$$

Equation 3.3: Sparsity formula [91].

2) The pre-defined LBC layer, it initializes through Bernoulli distribution with 0, 1, and −1 randomly based on the sparsity.

In the input image, each LBConv layer firstly applies p pre-defined non-learnable binary filters to generate difference maps which will be set of binary maps through ReLU function activation. Secondly, the output previous layer is linearly combined using q learnable 1×1 convolution filter to approximate the standard convolutional layers. While the training (i.e. back propagation algorithm) the LBCNN network, only the weight of second layer must be updated.

LBCNN is first proposed to handle the image classification task (i.e, MNIST, SVHN, Cifar-10) [61]. The LBCNN algorithm aim is to achieve two goals: First, obtain an effective deep learning network; and second, provide an alternative CNN network for ECG human identification where simpler and end to end learning can be accomplished. Unlike the conventional CNN, LBCNN has two important features, LBP concept and sparsity coding are also adopted to overcome the challenges of standard CNN mostly used in image vision task. Generally, LBCNN almost contains the layers of standard CNN expect of the convolution layer

is omitted by LBP layer (differential layer). In Fig. 3.2 and 3.3, we have shown the block-diagram of the LBCNN algorithm studied in this work.

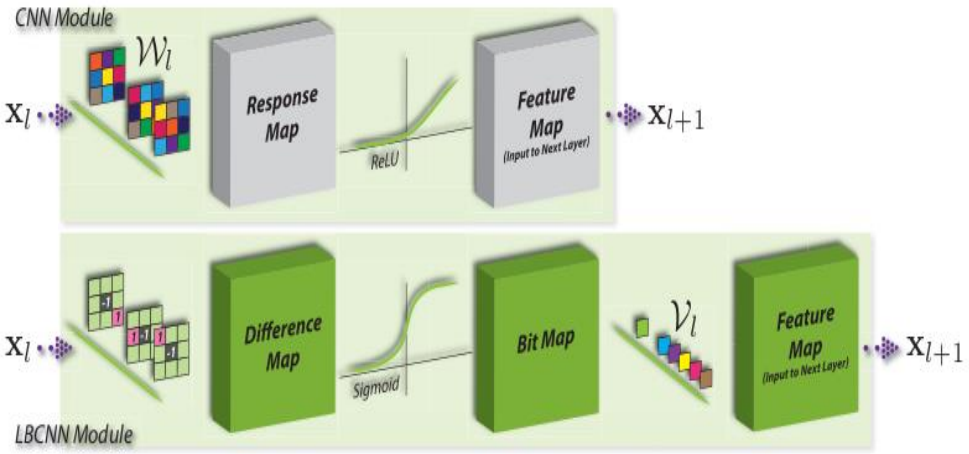


Figure 33.2: Basic module in CNN and LBCNN. Wl and Vl are the learnable weights for each module [61].

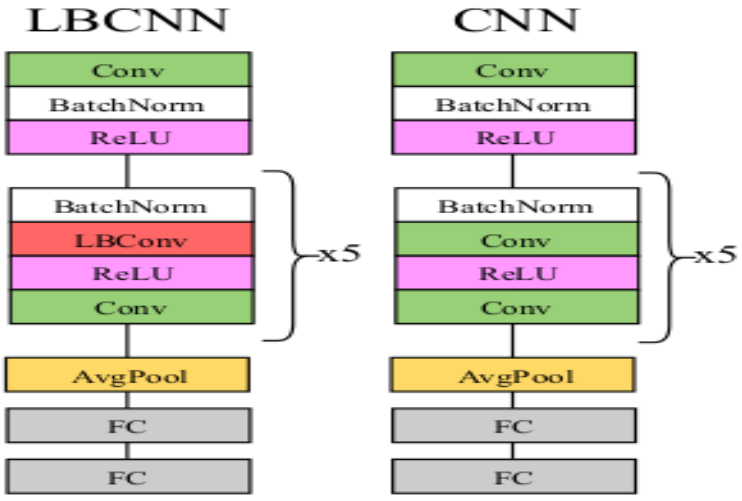


Figure 3.3: LBCNN/CNN architecture model [62]

3.3.1 The related works

Processing ECG signals using the time-frequency distributions has proven its effectiveness in finding the irregularities in ECG signals [76]. In fact, although the ECG features are highly variable, their detection using the Short-Time Fourier Transform (STFT)[77]. Can be easily performed with low complexity. In this work, techniques are also being introduced to capture both time and frequency domain characteristics of the signals as better inputs for machine learning models. This latter plays an important role

in biometric and its applications. Some of these methods train the being features such as color and texture, shapes, spectrogram energy, duration time, amplitude etc. and find the appropriate task. The main drawback of these methods is that they require hand crafted feature extraction from the image and, if not properly selected, the classifier result will not be accurate and would not provide a good accuracy.

The handcraft engineering approach was the dominant approach till recently when deep learning techniques started demonstrating recognition performance better than the carefully crafted feature extractors [78]. Indeed, as the developing of many architectures, the end to end classification task becomes much easier, and such high dimensionality data can be abstracted to low dimensionality where keep the important feature and then make a specific task. In 1995, Lucan, et al. [79] proposed a deep learning method for the first time, and it set off a wave of interest in deep learning, in the academic and industrial fields. Presently, deep learning shows a clear advantage in processing large data volumes of images and speech.

To achieve a good interpretation of ECG signal, several recent methods which are made to be robust to the heart rate variability, baseline draft, noise etc. The most recent approaches for ECG tasks based on CNN generally involve computing a QRS shape by extracting the spectral information related to ECG signal energy while heart beat activity. However, these approaches use heavily CNN network which are not suitable for real application time.

To deal with this issue, the binary filters for convolutional layers are considered a compressive CNN architecture [61], the BinaryConnect has been proposed by Courbariaux et al [63] who suggested the binary weight may achieve a good approximation for any object trained through the CNN network.

In [64,65] they proposed a new approach to deal with time consuming CNN network by omitting the 32-bit floating point multiply-accumulations by 1-bit XNOR-count operations. Their results show a good performance on MNIST, CIFAR-10 and SVHN dataset.

In addition to network binarization, model compression and network quantization technique Another category of technology [67, 68, 69, 70, 71, 72, 73 , 74] has been proposed that seeks to address the computational limitations of CNN networks.

However, the performance of such methods are usually upper bounded by the uncompressed and unquantized models.

Recently, a new CNN model is proposed [75], called Gabor Conventional Networks (GCNs or Gabor CNNs) [75]. It borrows Gabor filters in CNNs to enhance the sensibility of deep learned features to the orientation and scale changes, with much fewer learnable network parameters. it's improved CNNs on the generalization ability of rotation and scale variations by introducing extra functional modules on the basic element of CNNs, i.e. the convolution filters, and the extensive experiments has shown that these networks have greatly improved baselines, resulting in the state-of-the-art performance over several benchmarks.

In the end, the current CNN architectures can be divided into three categories. The first categories that are focused on the design of number layers, the pretrained data finetuning hyperparameter either by defining new training function or by applying bioinspired methods to solve the overfitting and overfitting. The second categories are almost using hybrid architecture with other kind of deep learning methods. The last categories are made to design new architecture by the standard CNN convolution layers by an alternative convolution layers such as local binary filter concept, Gabor filter, Lego filter [61, 75]. In case of ECG biometric, although there are several works achieved a significant result [80,81], the data nature including either the signal or image implies to develop a new architecture with low parameter's network and low time consuming that serve the existing applications in real world.

In this work, we have studied the ECG biometric human identification by proposing LBCNN to train the ECG spectrogram and identify the people based on their ECG shape.

3.4 Material and Methods

3.4 .1 ECG Database

Recently, many open access ECG databases are available for biometric purposes. They are categorized according to its software and acquisition environment. In order to well asses our proposed approach, we have selected 4 databases: ECG-ID, MIT-BIH, PTB, CYBHI [82].

The LBCNN presented in [83] was designed to exclusively operate in 2D data. This is why we have to covert the raw ECG signal to convert it the 2D representation. In the literature, the signal representation in frequency domain is considered the most important space to well illustrate the energy information as well to extract the important information in real condition settings. The Short-Time Fourier Transform (STFT) [77] (or short-term Fourier transform) is a powerful general-purpose tool for biomedical signal processing [84]. This technique decomposes signal after having projected upon well-defined time frequency distributions a set of high and low frequency components. The short-time Fourier transform (STFT) was defined as follows:

$$STFT\{x(n)\}(m,\omega) \equiv X(m,\omega) = \sum_{n=\infty}^{\infty} x(n) \times \omega\,(n-m) \times e^{-jx\omega xn} \qquad (3.4)$$

Equation 3.4: STFT formula [89]

Where x(n) is the discrete signal and ω(n) is a window. The sequence x(n) × ω(n − m) is a short-time section of the signal at time n. The square modulus of the short-time Fourier transform is referred to as the spectrogram (SPEC). The spectrogram was defined as follows:

$$SPEC\{x(n)\}(\tau,\omega) \equiv |X(\tau,\omega)|^2 \qquad (3.5)$$

Equation 3.5: Spectrogram formula [90]

Thus, the spectrogram of ECG signals was calculated using the following parameters: A Hamming window length of 250 samples, an overlap between segments of 96% (240 samples), and number of discrete Fourier transform points of 1,000. As result, the ECG spectrogram is depicted in the figure below:

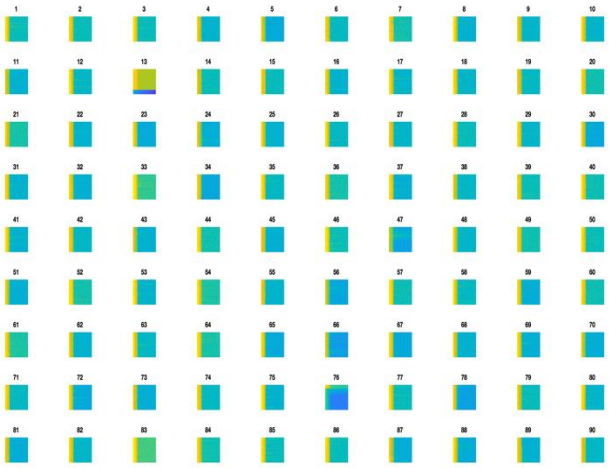


Figure 3.4: Graphical Representation of short segment of ECG as 40×40 spectrogram energy images

To reduce the amount of time consuming of CNN, the map is resized into a 40*40 square, as shown in **figure 3.4**. After converting to gray scale map, the time-frequency map can be used to train the neural network (In the CNN model, as shown in **Table 3.1**, C1, C2, C3, C4 and C5 are differential convolutional layers. F1 and F2 are the fully connected layers and SoftMax is the classification layer.) The proposed LBCNN architecture is given in **Table 3.1** .

Table 3.1:LBCNN architecture used in the ECG biometric identification system

Layer type	Activation unit	Kernel Size	Output Size	Number of parameters
Diff (C1)	ReLU	3 × 3	40 ×40 ×1	200
Diff (C2)	ReLU	3 ×3	40 ×40 ×1	250
Diff (C3)	ReLU	3 ×3	40 ×40 ×1	550
Diff (C4)	ReLU	3 ×3	40 ×40 ×1	300
Diff (C5)	ReLU	3 ×3	40 ×40 ×1	5000
Flaten (F1)			512	
Fully-connected (F2)	ReLU		512	262656
Output	SoftMax		18	9234

In the end, the LBCNN-ECG model can be divided into four stages:

- Data preprocessing: The number of raw signals were converted by using Equation **(** 3.4).
- Data preparation: We have divided our data two data sets, training and test data
- LBCNN-ECG model: Input the converted 2D ECG image to CNN, which is 5-layer differential layer and 2-layer fully- connected layer. Set the parameters including kernel size, padding number, and stride number. The updated parameter used in this study is (Stochastic gradient descent) SGD.
- Fine Tuning LBCNN model: we have made a series experiences to select the best parameters that allow our model to well train our model and to avoid the overfitting and underfitting phenomena.

- Model evaluation: The performance of system is assessed through many experiences by computing the accuracy metric.

3.5 Experimental Results and Discussion

In our experiment, L ECG signals for P different people are used, each with one ECG signals [82]. After transformation process, 2D ECG images with a dimension of 40x40 for each person are used. First, each 2D-CNN model, using N images is trained. Then, each model using M images is tested in the same condition. Afterward, the performance of the proposed method is evaluated by using rate accuracy as well-known metric evaluation.

Equal Error Rate (EER) and This error rate equates to the point at which the FAR and FRR cross (the compromise between False Acceptance Rate(FAR) and False Rejection Rate(FRR)). The smaller EER indicates a better performance.

$$HTER = (FAR + FRR)/2 \qquad (3.6)$$

Equation 3.6: HTER formula

Where, false acceptance rate is the number of imposter acceptance reject rate is the number of legitimate rejections.

Table 3.2: The number of ECG spectrogram images for each database used to validate our approach.

Data base	Raw signal	Data train N	Data test M
MIT-BIH	10000	240	240
ECG-ID	10000	540	540
PTB	10000	300	300
CYBHI long term	10000	372	37

As shown in **Figure 3.4** and **Table 3.1**, the proposed architecture consists of 7 layers: input layer of size 40×40, the five LBC layers, flatten layer, fully connected layer and at last the SoftMax layer of size N (i.e., the number subjects of each database are used in this work). Additionally, the performance of our architecture trained by LBCNN classifier is assessed in terms of the EER which is a very interesting metric for biometric systems evaluation. All datasets have achieved an EER ranging from 0.5 to 12 %, except for the CYBHI that achieved 15% EER. The next table summarizes a comparison between our proposed method for ECG based biometric and some existing works in terms of EER metric.

The performance for the proposed method is shown in **Table 3.3**. As it can be seen from **Table 3.3**, the CNN-LBCNN model which trains spectrogram energy of ECG as input gives a lower EER of 0.5% and while the second high ERR with 15%.

Table 3.3: Performance of LBCNN model

Method	EER
ECG-ID	0.05
ECG-sinus	0.06
PTB	0.12
CYBHI	0.15

The proposed method was compared to both other CNN based methods and conventional methods using spectrum energy feature. For a fair comparison, the methods evaluated with the fifth database were selected. The comparison results according to the EER were summarized in **Table 3.4**. As seen in **Table 3.4**, although the proposed method performed without any preprocessing step, it achieves a performance close to that given in [86, 87, 88]. It should be noted that the proposed method has been trained with a different database. In addition, the use of a database which consists of non-healthy individuals affected the performance of the proposed method.

Table 3.4: Comparison of the performance evaluation of the proposed approach to other existing biometric method based on deep learning approach.

Author	Extraction method	Database	EER%
R. Salloum et al [86]	RNN network	ECG-ID	20
Luz et al [87]	2D-CNN 1D-CNN	CYBHI CYBHI	20.48 15.60
Pinto et al [88]	1D-CNN	CYBHI PTB	15.37 9.06
Our approach	LBCNN	ECG-ID MITI-BIH CYBHI PTB	**10.08** **0.5** **12.00** **15.00**

3.6 Conclusion

In this section, ECG human identification has introduced to identify the human identity based on their ECG representation. The proposed ECG LBCNN system immediately recognizes the 2D ECG pattern without any handcraft features step. The proposed architecture reached a lower EER of 0.5%, for 2D ECG spectrogram within the ability of LBCNN to well train the ECG images with low parameter's network. The proposed CNN architecture, is also compared

to the related studies and we have achieved a significant result. LBCNN can be further improved in the following directions: 1) 1D LBCNN architecture to train directly the raw ECG signal instead of 2D ECG shape; 2) Reduction of the computational cost; 3) Application of LBCNN to other application tasks including EEG and EMG.

General Conclusion

General Conclusion

This master thesis is made to study the CNN performance on the ECG applications, the first study was focused on the prediction arrhythmia from ECG signal by performing the both CNN architectures such as 1D and 2D model. With the open-source deep learning platform TensorFlow, it was possible to implement and test a 2D and 1D-CNN architectures trained with different ECG heartbeats. Having a good tuning of hyperparameters, the both models achieves acceptable results, however, the scope of this thesis is not to develop a new architecture, but to explore most exiting CNN models, and perform it to classify ECG class based on these features.

Different hyperparameters were tuned by trial and error in order to improve the results. During experimenting it was quickly noted that the missing of data stability had major effect on the training process. On the other hand, heavily hyperparameters' network made the training too slow which is not suitable for real time application. Indeed, it was concluded that the conventional CNN architecture could not support the application of unbalanced data.

To deal with issues, in the second aim of these thesis is to test a new CNN architecture which utilize ECG spectrogram energy. The designed model was made for biometric task, and has tested on well-known databases. This work, which the framework is based on, proposed relatively good results for ECG human using spectrogram energy. However, unlike the one-dimensional ECG representation, the passage to 2D ECG image may miss some important temporal information which does appear noise within spectrogram energy. For ECG signals, which are complex sets of data, feature extraction is presumably a better solution. The purpose of this thesis is to look at the potentials of using CNN to classify ECG data. The framework only supports the 2D data. This thesis experiments with different types of network structures, with the aim to find out which parameters are important for the EER, and possibly performance.

The results of this thesis are relatively acceptable, and even though the data was not enough to well train the CNN network, the application still have great potentials. This thesis also deals with a more challenging ECG dataset, but since is bit small, the results comes with a little uncertainty. The framework is still a prototype-level system, and is far from practical system.

There are a number of continuations of this work which would allow for further investigated on the performance CNNs model on the ECG application. First, continuing work will concentrate on using LBCNN with new variants to predict the ECG label. By using the ECG that contains a high temporal pattern, it might also be possible to extended the 2D-LBCNN to 1D-LBCNN. Furthermore, through the use of Gabor filter and Lego filter in the CNN architecture instead of CNN filter may will be more practicable for signals analysis, and classification.

References

References

1. Articles :

[6] Gu, Jiuxiang, et al. "Recent advances in convolutional neural networks." *Pattern Recognition* 77 (2018): 354-377.

[7] Kiranyaz, Serkan, et al. "1D convolutional neural networks and applications: A survey." *arXiv preprint arXiv:1905.03554* (2019).

[13] Fernàndez García, Jaume. *Deep learning neural networks in malaria diagnosis*. MS thesis. Universitat Politècnica de Catalunya, 2014.

[14] Khan, Asifullah, et al. "A survey of the recent architectures of deep convolutional neural networks." *Artificial Intelligence Review* (2019): 1-62.

[15] LeCun, Yann, et al. "Learning algorithms for classification: A comparison on handwritten digit recognition." *Neural networks: the statistical mechanics perspective* 261 (1995): 276.

[16] Alom, Md Zahangir, et al. "The history began from alexnet: A comprehensive survey on deep learning approaches." *arXiv preprint arXiv:1803.01164* (2018).

[17] Krizhevsky, Alex, and Geoff Hinton. "Convolutional deep belief networks on cifar-10." *Unpublished manuscript* 40.7 (2010): 1-9.

[18] Deng, Li. "The mnist database of handwritten digit images for machine learning research [best of the web]." *IEEE Signal Processing Magazine* 29.6 (2012): 141-142.

[19] Clifford, Gari D., et al. "AF Classification from a short single lead ECG recording: the PhysioNet/Computing in Cardiology Challenge 2017." *2017 Computing in Cardiology (CinC)*. IEEE, 2017.

[20] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.

[24] McCloskey, Stephen, et al. "Detecting hypopnea and obstructive apnea events using convolutional neural networks on wavelet spectrograms of nasal airflow." *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, Cham, 2018.

[25] Cen, Ling, et al. "Automatic system for obstructive sleep apnea events detection using convolutional neural network." *2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*. IEEE, 2018.

[26] Wang, Wei, et al. "End-to-end encrypted traffic classification with one-dimensional convolution neural networks." *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 2017.

[27] Urtnasan, Erdenebayar, et al. "Automated detection of obstructive sleep apnea events from a single-lead electrocardiogram using a convolutional neural network." *Journal of medical systems* 42.6 (2018): 104.

[28] Urtnasan, Erdenebayar, Jong-Uk Park, and Kyoung-Joung Lee. "Multiclass classification of obstructive sleep apnea/hypopnea based on a convolutional neural network from a single-lead electrocardiogram." *Physiological measurement* 39.6 (2018): 065003.

[29] Dey, Debangshu, Sayanti Chaudhuri, and Sugata Munshi. "Obstructive sleep apnoea detection using convolutional neural network based deep learning framework." *Biomedical engineering letters* 8.1 (2018): 95-100.

[30] Haidar, Rim, Irena Koprinska, and Bryn Jeffries. "Sleep apnea event detection from nasal airflow using convolutional neural networks." *International Conference on Neural Information Processing*. Springer, Cham, 2017.

[31] Haidar, Rim, et al. "Convolutional neural networks on multiple respiratory channels to detect hypopnea and obstructive apnea events." *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018.

[32] McCloskey, Stephen, et al. "Detecting hypopnea and obstructive apnea events using convolutional neural networks on wavelet spectrograms of nasal airflow." *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, Cham, 2018.

[33] Wu, Yunan, et al. "A comparison of 1-D and 2-D deep convolutional neural networks in ECG classification." *arXiv preprint arXiv:1810.07088* (2018).

[34] Zihlmann, Martin, Dmytro Perekrestenko, and Michael Tschannen. "Convolutional recurrent neural networks for electrocardiogram classification." *2017 Computing in Cardiology (CinC)*. IEEE, 2017.

[37] Ferraz, Carolina Toledo, and José Hiroki Saito. "A comprehensive analysis of Local Binary Convolutional Neural Network for fast face recognition in surveillance video." *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*. 2018.

[39] Ojala, Timo, Matti Pietikäinen, and David Harwood. "A comparative study of texture measures with classification based on featured distributions." *Pattern recognition* 29.1 (1996): 51-59.

[40] Wei, Xiangpo, et al. "Convolutional neural networks and local binary patterns for hyperspectral image classification." *European Journal of Remote Sensing* 52.1 (2019): 448-462.

[41] Ojala, Timo, Matti Pietikainen, and Topi Maenpaa. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns." *IEEE Transactions on pattern analysis and machine intelligence* 24.7 (2002): 971-987.

[42] Ahonen, Timo, Abdenour Hadid, and Matti Pietikainen. "Face description with local binary patterns: Application to face recognition." *IEEE transactions on pattern analysis and machine intelligence* 28.12 (2006): 2037-2041.

[43] Hadid, Abdenour, et al. "Face analysis using local binary patterns." *Handbook of Texture Analysis*. 2008. 347-373.

[44] Huijsmans, Dionysius P., and Nicu Sebe. "Content-based indexing performance: size normalized precision, recall, generality evaluation." *Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429)*. Vol. 3. IEEE, 2003.

[45] Grangier, David, and Samy Bengio. "A discriminative kernel-based approach to rank images from text queries." *IEEE transactions on pattern analysis and machine intelligence* 30.8 (2008): 1371-1384.

[46] Ali, Wajid, Fredrik Georgsson, and Thomas Hellstrom. "Visual tree detection for autonomous navigation in forest environment." *2008 IEEE Intelligent Vehicles Symposium*. IEEE, 2008.

[47] Nanni, Loris, and Alessandra Lumini. "Ensemble of multiple pedestrian representations." *IEEE Transactions on Intelligent Transportation Systems* 9.2 (2008): 365-369.

[48] Mäenpää, Topi, Jaakko Viertola, and Matti Pietikäinen. "Optimising colour and texture features for real-time visual inspection." *Pattern Analysis & Applications* 6.3 (2003): 169-175.

[49] Turtinen, Markus, Matti Pietikainen, and Olli SilvÉn. "Visual characterization of paper using isomap and local binary patterns." *IEICE transactions on information and systems* 89.7 (2006): 2076-2083.

[50] Heikkila, Marko, and Matti Pietikainen. "A texture-based method for modeling the background and detecting moving objects." *IEEE transactions on pattern analysis and machine intelligence* 28.4 (2006): 657-662.

[51] Kellokumpu, Vili, Guoying Zhao, and Matti Pietikäinen. "Human activity recognition using a dynamic texture based method." *BMVC*. Vol. 1. 2008.

[52] Oliver, Arnau, et al. "False positive reduction in mammographic mass detection using local binary patterns." *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, Berlin, Heidelberg, 2007.

[53] Kluckner, Stefan, et al. "A 3D teacher for car detection in aerial images." *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007.

[55] Jin, Hongliang, et al. "Face detection using improved LBP under Bayesian framework." *Third International Conference on Image and Graphics (ICIG'04)*. IEEE, 2004.

[56] Wang, Haijing, Peihua Li, and Tianwen Zhang. "Histogram feature-based Fisher linear discriminant for face detection." *Neural Computing and Applications* 17.1 (2008): 49-58.

[57] Ahonen, Timo, Abdenour Hadid, and Matti Pietikainen. "Face description with local binary patterns: Application to face recognition." *IEEE transactions on pattern analysis and machine intelligence* 28.12 (2006): 2037-2041.

[58] Zhao, Jiali, et al. "LBP discriminant analysis for face verification." *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*. IEEE, 2005.

[59] Shan, Caifeng, Shaogang Gong, and Peter W. McOwan. "Facial expression recognition based on local binary patterns: A comprehensive study." *Image and vision Computing* 27.6 (2009): 803-816.

[60] Tan, Xiaoyang, and Bill Triggs. "Enhanced local texture feature sets for face recognition under difficult lighting conditions." *International workshop on analysis and modeling of faces and gestures*. Springer, Berlin, Heidelberg, 2007.

[61] Ferraz, Carolina Toledo, and José Hiroki Saito. "A comprehensive analysis of Local Binary Convolutional Neural Network for fast face recognition in surveillance video." *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*. 2018.

[62] Juefei-Xu, Felix, Vishnu Naresh Boddeti, and Marios Savvides. "Local binary convolutional neural networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[63] Courbariaux, Matthieu, Yoshua Bengio, and Jean-Pierre David. "Binaryconnect: Training deep neural networks with binary weights during propagations." *Advances in neural information processing systems*. 2015.

[64] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1." *arXiv preprint arXiv:1602.02830* (2016).

[65] Hubara, Itay, et al. "Quantized neural networks: Training neural networks with low precision weights and activations." *The Journal of Machine Learning Research* 18.1 (2017): 6869-6898.

[66] Rastegari, Mohammad, et al. "Xnor-net: Imagenet classification using binary convolutional neural networks." *European conference on computer vision*. Springer, Cham, 2016.

[67] Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." *arXiv preprint arXiv:1602.07360* (2016).

[68] Wu, Jiaxiang, et al. "Quantized convolutional neural networks for mobile devices." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[69] Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." *arXiv preprint arXiv:1510.00149* (2015).

[70] Chen, Wenlin, et al. "Compressing neural networks with the hashing trick." *International conference on machine learning*. 2015.

[71] Denil, Misha, et al. "Predicting parameters in deep learning." *Advances in neural information processing systems*. 2013.

[72] Han, Song, et al. "Learning both weights and connections for efficient neural network." *Advances in neural information processing systems*. 2015.

[73] Soudry, Daniel, Itay Hubara, and Ron Meir. "Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights." *Advances in neural information processing systems*. 2014.

[74] Esser, Steve K., et al. "Backpropagation for energy-efficient neuromorphic computing." *Advances in neural information processing systems*. 2015.

[75] Luan, Shangzhen, et al. "Gabor convolutional networks." *IEEE Transactions on Image Processing* 27.9 (2018): 4357-4366.

[76]Reddy, D. C.Biomedical signal processing: principles and techniques. McGraw-Hill, 2005.

[77] Griffin, Daniel, and Jae Lim. "Signal estimation from modified short-time Fourier transform." *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32.2 (1984): 236-243.

[78] Yu, Dong, and Li Deng. "Deep learning and its applications to signal and information processing [exploratory dsp]." *IEEE Signal Processing Magazine* 28.1 (2010): 145-154.

[79] LeCun, Yann, and Yoshua Bengio. "Convolutional networks for images, speech, and time series." *The handbook of brain theory and neural networks* 3361.10 (1995): 1995.

[80] Lei, Xiafei, Yue Zhang, and Zongqing Lu. "Deep learning feature representation for electrocardiogram identification." *2016 IEEE International Conference on Digital Signal Processing (DSP)*. IEEE, 2016.

[81] Abdeldayem, Sara S., and Thirimachos Bourlai. "ECG-based human authentication using high-level spectro-temporal signal features." *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018.

[82] Merone, Mario, et al. "ECG databases for biometric systems: A systematic review." *Expert Systems with Applications* 67 (2017): 189-202.

[83]JUEFEI-XU, Felix, NARESH BODDETI, Vishnu, et SAVVIDES, Marios. Local binary convolutional neural networks. In : Proceedings of the IEEE conference on computer vision and pattern recognition.2017. p. 19-28.

[84] Dripps, J. H. "An introduction to time-frequency methods applied to biomedical signals." (1997): 1-1.

[85] LeCun, Yann, and Yoshua Bengio. "Convolutional networks for images, speech, and time series." *The handbook of brain theory and neural networks* 3361.10 (1995): 1995.

[86] Salloum, Ronald, and C-C. Jay Kuo. "ECG-based biometrics using recurrent neural networks." *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017.

[87] da Silva Luz, Eduardo Jose, et al. "Learning deep off-the-person heart biometrics representations." *IEEE Transactions on Information Forensics and Security* 13.5 (2017): 1258-1270.

[88] Pinto, Joao Ribeiro, and Jaime S. Cardoso. "A end-to-end convolutional neural network for ECG based biometric authentication." *10th IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2019)*. 2019.

[89] Diab, Mohamad O., et al. "A Review on ECG-Based Biometric Authentication Systems." *Hidden Biometrics*. Springer, Singapore, 2020. 17-44.

[90] Hampton, John. *The ECG in practice*. Elsevier Health Sciences, 2013.

[91] Sebastien, Marcel, Mark Nixon, and Stan Li. "Handbook of biometric anti-spoofing: trusted biometrics under spoofing attacks." (2014).

2.Web sites:

[1] Biomedical Signal Processing, https://www.embs.org/about-biomedical-engineering/our-areas-of-research/biomedical-signal-processing/

[2] Powerline noise elimination in biomedical signals via blind source separation and wavelet analysis, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4493666/

[3]Frequencydomain,https://books.google.dz/books?id=fb02lEByjoC&pg=PA189&lpg=PA189&dq=signal+analysis+is+divided+into+two+classes;+spatial+and+frequency+domain&source=bl&ots=5ZXUr_I_NZ&sig=ACfU3U0d3Ia6a4AXSNgnrTNnWNcDARzIXw&hl=en&sa=X&ved=2ahUKEwiUxs_j5qnpAhUux4UKHZHxBuEQ6AEwAHoECAUQAQ#v=onepage&q=signal%20analysis%20is%20divided%20into%20two%20classes%3B%20spatial%20and%20frequency%20domain&f=false

[4] Biomedical Signal Analysis, http://unhas.ac.id/tahir/BAHAN-KULIAH/BIO-MEDICAL/NEW/HANBOOK/18_Biomedical_Signal_Analysis.pdf

[5] Pattern Recognition System,https://www.sciencedirect.com/topics/engineering/pattern-recognition-systems

[8] What is the key difference between AI machine learning and deep learning What are some real life,https://www.quora.com/What-is-the-key-difference-between-AI-machine-learning-and-deep-learning-What-are-some-real-life-examples

[9] Activation Functions and Optimizers for Deep Learning Models, https://becominghuman.ai/activation-functions-and-optimizers-for-deep-learning-models-5a1181649d6b

[10]_Machine Learning vs Deep Learning, https://dzone.com/articles/comparison-between-deep-learning-vs-machine-learni

[11] Deep learning architectures,___https://developer.ibm.com/technologies/artificial-intelligence/articles/cc-machine-learning-deep-learning-architectures/

[12] Deep Learning, https://developer.nvidia.com/deep-learning

[21] CNN for Handwritten Arabic Digits Recognition Based on LeNet-5, https://www.researchgate.net/publication/309493055_CNN_for_Handwritten_Arabic_Digits_Recognition_Based_on_LeNet-5/citation/download

[22] Subject independent facial expression recognition with robust face detection using a convolutional neural network,
https://www.sciencedirect.com/science/article/abs/pii/S0893608003001151?via%3Dihub

[23] Image Classification with Convolutional Neural Networks,
https://medium.com/@ksusorokina/image-classification-with-convolutional-neural-networks-496815db12a8

[35] Neural network and genetic algorithm-based hybrid approach to expanded job-shop scheduling,https://www.sciencedirect.com/science/article/abs/pii/S0360835201000109

[36] Vision-based Vehicle Type Classification Using Partial Gabor Filter Bank,
https://ieeexplore.ieee.org/document/4338720

[38] Convolutional neural networks and local binary patterns for hyperspectral image classification,https://doi.org/10.1080/22797254.2019.1634980

[54] Center for Machine Vision and Signal Analysis,

http://www.ee.oulu.fi/mvg/page/lbp_bibliography. The availability of the link was last checked on 15 Nov., 2010

Abstract

Abstract

With the rapid development of biometric identification in healthcare, the bio signal of human being (ECG, EEG, EMG) has attracted much attention from the research community. However, the raw signal acquired by different sensors mostly is noised and consequently affects the biometric system accuracy. Unlike the handcraft features algorithms, deep learning algorithms have proved their ability to learn automatically the original data without mostly any preprocessing. Among of those, CNN is widely used ECG application and achieved a remarkable result. However, the exiting CNN architectures are mostly conducted for 2D data. To deal with this challenge. Deep learning methods such as algorithms based on Convolution Neural Networks (CNN) can be used to avoid conventional extraction crafting of features from ECG signals. We have designed two CNN architectures including 1D and 2D ECG arrhythmia classification. On this experiences, various fine tuning hyperparameter training strategies based on various database were implemented and tested. All results have achieved satisfactory result. In case of ECG based biometric, we have proposed a new variant of CNN called LBCNN, which is carried out on well-known databases and without any pre-processing step. An equal error rate of 0.5 % via LBCNN was achieved using ECG spectrogram energy. Lower EER with LBCNN proves that 1D LBCNN can be applied for ECG detection and classification.

Keywords: ECG, biometric, CNN, LBCNN, Deep learning**.**

Résumé

Avec le développement rapide de l'identification biométrique dans les soins de santé, le bio signal de l'être humain (ECG, EEG, EMG) a attiré beaucoup d'attention de la communauté des chercheurs. Cependant, le signal brut acquis par les différents capteurs est principalement sonore et affecte par conséquent la précision du système biométrique. Contrairement aux algorithmes de l'artisanat, les algorithmes d'apprentissage profond ont prouvé leur capacité à apprendre automatiquement les données d'origine sans la plupart du temps tout prétraitement. Parmi ceux-ci, CNN est largement utilisé application ECG et a obtenu un résultat remarquable. Cependant, les architectures CNN sortantes sont principalement conduites pour les données 2D. Pour faire face à ce défi. Les méthodes d'apprentissage profond telles que les algorithmes basés sur les réseaux neuronaux de convolution (CNN) peuvent être utilisées pour éviter l'extraction conventionnelle de caractéristiques des signaux ECG. Nous avons conçu deux architectures CNN, y compris la classification 1D et 2D de l'arythmie ECG. Sur cette expérience, diverses

stratégies de formation au réglage fin des hyperparamètres basées sur diverses bases de données ont été mises en œuvre et testées. Tous les résultats obtenus sont satisfaisants. Dans le cas de la biométrie basée sur l'ECG, nous avons proposé une nouvelle variante de CNN appelée LBCNN, qui est effectuée sur des bases de données bien connues et sans aucune étape de pré-traitement. Un taux d'erreur égal de 0,5 % via LBCNN a été obtenu en utilisant l'énergie du spectrogramme ECG. Un EER inférieur avec LBCNN prouve que 1D LBCNN peut être appliqué pour la détection et la classification ECG.

Mots-clés : ECG, CNN, LBCNN, biométrique.

ملخص:

ومع التطور السريع للتعرف على القياسات الحيوية في مجال الرعاية الصحية، اجتذبت الإشارة الحيوية للإنسان (ECG، EEG، EMG) الكثير من الاهتمام من جانب مجتمع البحوث. ومع ذلك، فإن الإشارة الأولية التي يتم الحصول عليها من أجهزة استشعار مختلفة تكون مسموعة بشكل أساسي وبالتالي تؤثر على دقة نظام القياسات الحيوية. على عكس الخوارزميات الحرفية، أثبتت خوارزميات التعلم العميق قدرتها على تعلم البيانات الأصلية تلقائيًا دون أي معالجة مسبقة في معظم الاحيان. ومن بين هذه التطبيقات، يستخدم CNN على نطاق واسع تطبيق ECG وحقق نتيجة ملحوظة. ومع ذلك، فإن بنى CNN الصادرة تعتمد بشمل أساسي على بيانات ثنائية الأبعاد. لمواجهة هذا التحدي يمكن استخدام طرق التعلم العميق مثل الخوارزميات المستندة إلى الشبكات التلافيفية العصبية (CNN) لتجنب الاستخراج التقليدي للميزات من إشارات مخطط القلب. لقد قمنا بتصميم هيكلين لCNN ، بما في ذلك التصنيف 2D و1D لاضطراب ضربات القلب ECG . بناءً على هذه التجربة ، تم تنفيذ واختبار استراتيجيات تدريب متنوعة لصقل المعلمات الفائقة بناءً على قواعد بيانات مختلفة. جميع النتائج التي تم الحصول عليها مرضية. في حالة القياسات الحيوية المستندة الى ECG، اقترحنا متغيرا جديداً من CNN يسمىLBCNN ، والذي يتم تنفيذه على قواعد بيانات معروفة وبدون أي خطوة سابقة للمعالجة. تم تحقيق معدل خطأ يساوي 0.5 % عبر LBCNN باستخدام طاقة مخطط طيف القلبECG. يُظهر انخفاض كفاءة الطاقة(EER)مع LBCNN انه يمكن تطبيق ويثبت جهاز EER السفلي مع 1DLBCNN لاكتشاف و تصنيف مخطوط كهربية القلب.

الكلمات المفتاحية: مخطط القلب الكهربائي, الشبكات العصبية التلافيفية العصبية.