Al-Quds University

Faculty of Graduate Studies

Arabic Alphabet Deaf Sign Gestures Recognition Based on Deep Machine Learning Methods

Ohood Adel Salameh Darabee

M.Sc. Thesis

Jerusalem-Palestine

1440-2019

Arabic Alphabet Deaf Sign Gestures Recognition Based on Deep Machine Learning Methods

Prepared by:

Ohood Adel Salameh Darabee

B.Sc. Computer Science

Hebron University/Palestine

Supervisor: Dr. Abdullah Kamal

A thesis submitted in fulfillment of requirements for the degree of Master of Computer Science of Al-Quds University.

1440/2019

Al-Quds University

Deanship of Graduate Studies

Computer Science

Thesis Approval

Arabic Alphabet Deaf Sign Gestures Recognition Based on Deep Machine Learning Methods

Student Name: Ohood Adel Salameh Darabee

Registration No: 21311351

Supervisor: Dr. Abdullah Kamal

Master thesis submitted and accepted date: 13 / 7 /2019

The names and signatures of the examining committee members are as follows:

1-Head of committee: Dr. Abdullah Kamal signature:**....................**

2- Internal Examiner: Dr. Badie Sartawi signature: **..................**

3- External Examinar: Dr.Ahmad Hasasneh signature: **..................**

Jerusalem- Palestine

1440/2019

Dedication

I dedicate this to souls in the sky; may god rest them in Heaven.

I dedicate this to my mother and family.

I dedicate this to my supportive husband and my only army; Lutfi

Thanks for all

Declaration

I certify that this thesis submitted for the degree of Master of Science is the result of my own research, except where otherwise acknowledged, and that this thesis (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed ………………..

Ohood Adel Salameh Darabee

Date: 13 / 7 / 2019

Acknowledgments

I would like to give my sincere thanks and gratitude to people that give assistant and support for me during master research.

Special mention for Dr Abdullah Kamal and all my doctors who inspired me to work in the field of recognition and classification. Thank you for your invaluable comments and suggestions.

I would like to give special thanks for the support and love of my husband, who supported me financially and morally.

I would like to thank Palestinian Red Crescent Society. Special thanks for the sign specialist Ms. Engy Alabed.

I would like to give special thanks to the Media institute at Al-Quds University. Special thanks for the photographer Mr.Shadi Barakat

Finally yet importantly thanks goes to Al-Quds University, especially for Faculty of high studies\ Computer science department.

Abstract

Sign language continue to be the best method to communicate between the deaf and hearing impaired. Hand gestures enable communication between deaf people during their daily lives rather than speaking. In our society, Arabic Sign Language (ArSL) is only known for deaf people and specialist, which makes the community of deaf people narrow. Recognizing and documentation of ASL have only been paid attention recently. This thesis proposes a real time gesture recognition system for Arabic sign language based on Restricted Boltzman Machines (RBMs) following by the use of tiny images. RBMs have the ability to code images as structures of limited features taken from a large alphabet. This process continue to repeat itself in a deep networking to get an efficient sparse representation of the initial data in the feature space. A complex problem of recognition in the input space is thus transformed into an easier one in the feature space. In this thesis, we show that ASL can thus be achieved using tiny images instead of conventional Bag-of-Words (BoWs) methods. After appropriate coding, a simple classifier in the feature space suffices to compute the probability of a particular character according to the input image.

Table of Contents

Declaration I

Acknowledgments II

Abstract III

Contents IV

List of Figures VI

List of Tables VIII

List of Acronyms IX

1. Chapter 1 1

  1.1. Introduction 1

  1.2. Problem definition and overview 2

  1.3. Introduction to deaf sign language 3

  1.4. Thesis objectives 6

  1.5. Major contributions 7

  1.6. Thesis benefits 7

  1.7. Thesis constraints 7

  1.8. Thesis organization 8

2. Chapter 2 9

Background and related work 9

  2.1. Introduction to Gesture Recognition 9

  2.2. **Introduction to ArSL** 10

  2.3. Current approaches to ArSL 11

    2.3.1. Classification methods 12

  2.4 Deep learning approaches 29

    2.4.1 Restricted Boltzmann Machines 30

    2.4.2. RBM learning algorithm 33

    2.4.3. Deep Belief Network 34

  2.5 Summary and recommendations 35

3. Chapter 3 36

Proposed Model 36

  3.1. Introduction 36

  3.2. The Database 38

  3.3. Image Pre-processing 39

3.3.1. **Image Grey Scaling**......39
3.3.2. **Use of Tiny Images**......40
3.3.3. **Data Whitening and Normalization**......41
3.4. Unsupervised Feature Space Construction.......42
3.5. Summary......51
4. Chapter 4......52
Results......52
4.1. Introduction......52
4.2. Data Collecting Results......53
4.3. Image pre-processing results......54
4.4 Features extraction based on RBM training on tiny images......58
4.6Summary......65
5.Chapter 5......66
Conclusion and Future Work.......66
5.1: **Introduction**......66
5.2: Contribution......66
5.3: Results:......66
5.4: Limitation.......66
5.5: Future Work......66
References......68

List of Figures

Figure 1—1 : Sign language recognition stages 2
Figure 1 2—Using iconic features from nature to represent words.[8] 4
Figure 1 3—Using miming in relating to space to describe prepositions.[8] 4
Figure 1—4 : Arabic sign language alphabets 5
Figure 0—1 Arabic sign language alphabets 11
Figure 0 : 2—K-Nearest Neighbor Algorithm example.[38] 15
Figure 0—3 Probabilistic parameters of a hidden Markov model (an example), where X represents the states, y represents possible observations, a represents the state transition probabilities, and b represents the output probabilities 18
Figure 04— HMM Chain to predict the weather (Bishop, 2006). 19
Figure 05— : HMM Chain for weather prediction (Bishop, 2006). 19
Figure 0—6 : High-level structure of an SVM approach. 20
Figure 0—7 : linear SVM. 21
Figure 0—8 : non-linear SVM. 21
Figure 0—9 : SVM mapping. 22
Figure 0—10 : Soft max linear classification for two different classes 23
Figure 0—11 : Logistic sigmoid activation function: with 24
Figure 0—12 : An example for scalar product of neural networks. 27
Figure 0—13 : Basic Structure of multilayer neural network.....................................27
Figure 0—14 : Left figure represent a classical Boltzmann machine, where there is a fully connection between the visible units themselves, hidden units themselves, and between the visible and hidden units. Right figure represent an RBM where there is no direct conn..31
Figure 0—15 : Restricted Boltzmann Machines (RBM layer wise) structure that build Deep Belief Network. 34
Figure 31—: the proposed model 37
Figure 32—: Arabic Alphabet sign language Database used in the proposed model..39
Figure 33—: The difference between colored and grey scaled image........................40
Figure 34—: Resizing image from the normal size to 100*100 pixel then to 42*42 pixel. Notice that the tiny image still recognizable..41

Figure 35—: On the left is normalized 8*8-patch sample, on the right whitened 8*8-sample patch....42
Figure 36—: shows the training of RBM machine using constructive divergence learning 44
Figure 01—: The Arabic Alphabet signs Dataset done by the specialist Ms. Engy Alaabed 54
Figure 02—: Samples from the training dataset (أ،ب،ث) taken under different attribution....54
Figure 03—: Samples of image cropping to eliminate the un-necessary elements in the image and centered the hand gesture. 55
Figure 04—: Samples of image grey scaled 56
Figure 05—: Samples of image resizing on the dataset....56
Figure 06— Applying data normalization on the Arabic Alphabet database....57
Figure 0—7: the 1024 features extracted by RBM training on the dataset. 59

List of Tables

Table 4.1	Classification Result 1	page 59
Table 4.2	Classification Result 2	page 60
Table 4.3	Classification Result 3	page 60
Table 4.4	Comparison of ArSlR systems Result	page 62

List of Acronyms

ASL	**Arabic Sign Language**
RBMs	**Restricted Boltzmann Machines**
BoW	**Bag of Words**
ArSLR	**Arabic Sign Language Recognition**
ROI	**Region of interests**
DBNs	**Beep Belief Networks**
CD	**Constructive Divergence**
SVM	**Support Vector Machine**
HCI	**Human Computer Interface**
K-NN	**K-Nearest Neighbor**
HMM	**Hidden Marcov Model**
NN	**Neural Network**
MLP	**Multi-layer perceptron**
ICA	**Independent Component Analysis**
DBN	**Deep belief Network**
RBM	**Restricted Boltzmann Machine**
MCMC	**Marcov Chain Monte-Carlo**

1. Chapter 1

Introduction

1.1. Introduction

The most natural way that human being used to communicate with each other is by using voice, gestures, and human-machine interfaces. However, the last method is still very primitive and force to adapt the machine requirements. Instead of using voice, hand gestures enable communication between deaf people during their daily live. However, Arabic Sign Language (ArSL) or gesture recognition is known only for deaf people and specialists in our society, thus the community of deaf people is very limited and narrow. Recently, sign language recognition systems for American, British, Indian, Chinese, Turkish, and many international sign languages have received much attention compared to the Arabic sign language. Therefore, developing an Arabic sign language recognition system (ArSLR) is needed. A review of recent developments in sign language recognition can be found in [1][2][3].

As shown in figure 1, ArSLR can be performed through two main phases: detection and classification. In the detection phase, each captured image is pre-processed, improved, and then the Regions of Interest (ROI) is identified using a segmentation algorithm. The output of the segmentation process can thus be used to perform the classification process. Indeed, accuracy and speed of detection play an important role in obtaining accurate and fast recognition process. In the recognition phase, a set of features are extracted from each segmented hand sign and then used to perform the recognition process. These features can therefore be used as a reference to understand the differences among the different classes. As mentioned earlier, ArSLR systems have only been paid attention recently, where few attempts have investigated and addressed this problem, see for example [4]–[6]. Therefore, the question of ArSL recognition is still an open question and need to be investigated using new and different machine learning methods. This thesis proposes a new Arabic sign recognition system based on new machine learning methods and a direct use of tiny images.

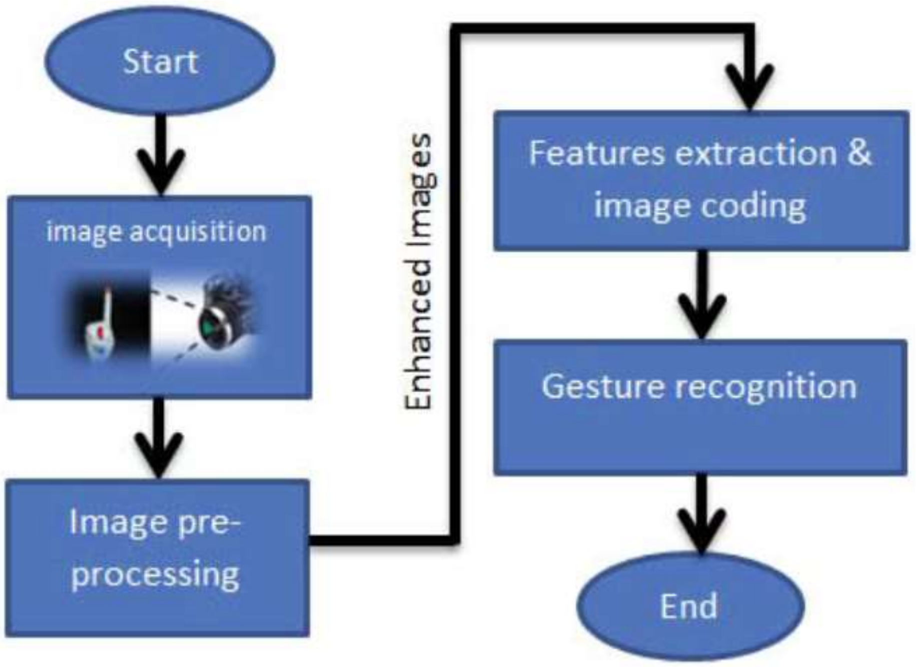


Figure 1—1 : Sign language recognition stages

In this introduction, we start by identifying the research problem statement and the substantial motivations to propose a new approach to ArSLR problem. We then describe the main objectives, advantages, and contributions to develop this work. We end this chapter by presenting an outline of the remaining chapters of this thesis.

1.2. Problem definition and overview

Deaf people use sign language to communicate with each other, but they cannot communicate with normal people because they cannot read sign language, which makes the community of deaf people very narrow. Statistics show that over 3% of the palestinian populations are hearing impaired [1]. In particular, according to the Palestinian Central Bureau of Statistics, 19% of the disabled Palestinian people are deaf and mute [7]. Indeed, helping those people is very important and thus developing systems capable of translating sign languages into text or spoken language is highly needed. Developing such systems will definitely participate in facilitating the communication between the hearing impaired and the normal people. Many approaches have been proposed to achieve sign language recognition; however, few of them have been developed to achieve ArSLR Therefore, the question of ArSLR is still open and need to be investigated using new machine learning methods. More precisely, a vision-based approach is proposed in this thesis based on a new machine learning method and a direct use of tiny images.

1.3. Introduction to deaf sign language

Deaf signs or gesture sings of sign language are a type of non-verbal communication used by a speaker to aid communication. Typically, deaf sign language is different from one region to another and from one language to another. For instance, researchers in the Middle East and Arab countries started to pay attention to deaf sign languages in early 90s. It has been shown that ArSL is the most difficult recognition task among others foreign sing languages due to its unique structure and its complex grammar [8]. Therefore, developing recognition systems for ArSL is a big challenge. Some characteristics of ArSl can be summarized as follows:

- The Arabic alphabets, numbers, and some specific words can be presented by one hand movement or gesture. Two consecutive movements or gestures can present other Arabic words. Finally, some words need more than three consecutive movements or gestures to be presented, more information can be found in [8].
- Arabic language has two forms of communication between Arab people, standard and local or colloquial [9]. Arabic communities combine both standard and colloquial dialect in there speech, where diglossic is the term that refers to these communities. Although Arabic is diglossic ArSL is not; For example the word happiness means both FARAH and SAADAH in Arabic. However, deaf persons can empress both FARAH and SAADAH by the same sign or gesture.
- Arabic language has very restricted and hard grammars, however Arabic sign language does not take into account all of these restrictions and grammars. ArSL consider the most spoken words and phrases with no ability to give the literary meaning.
- Each Arabic country has its own sign language. It is supposed that all Arab world should have one sign language but this was not possible, this refers to the regional dialect of each country. However, there is so much common

between these signs languages. Therefore, great efforts have been made to establish and identify the sign language that used in individual countries, such as Jordan, Egypt, and the Gulf states, the efforts is to standardize the language and spread it to all members of the deaf community and people whose concerned. Such efforts produced many sign languages, almost as many as Arabic-speaking countries, yet with the same sign alphabets [10], which is shown in figure 4.

- As shown in figures 2 and 3, the resources of the signs in ArSL is come from either miming or use the iconic feature to represent the word such as miming the rectangle shape to represent the word rectangle. Alternatively, borrowing signs from another sign languages such as American or French sign languages, especially signs in relation with preposition such as under, above and between.

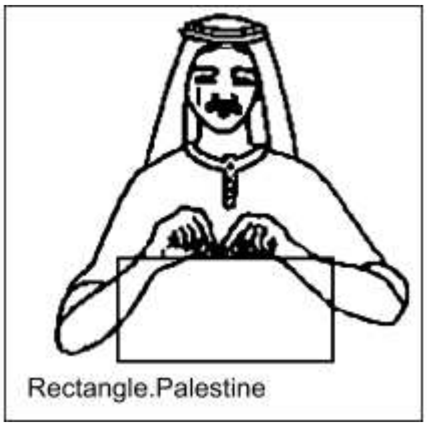


Figure 1 2—Using iconic features from nature to represent words.[8]

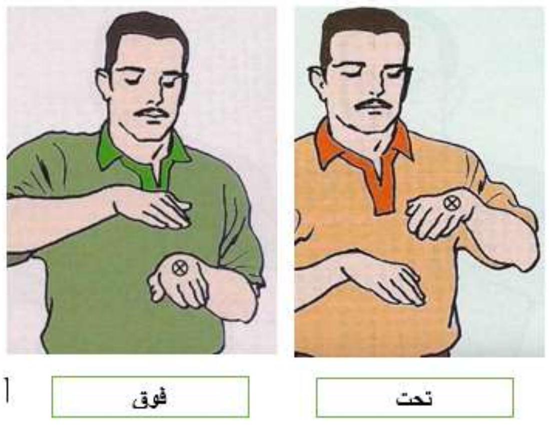


Figure 1 3—Using miming in relating to space to describe prepositions.[8]

- In ArSL , not only the configuration of hand is needed, but also the position of the hand in relation to the body is needed to present the meaning of the sign. In addition to that, sign can be repeated several times to present continuous words or verbs.
- ArSL can only deal with verbs, nouns and adjective in a direct way. Other terms may need hand motion or combined of more than two signs to be expressed. ArSL vocabulary is composed of synosigns, antosigns, homosigns, and compounds.

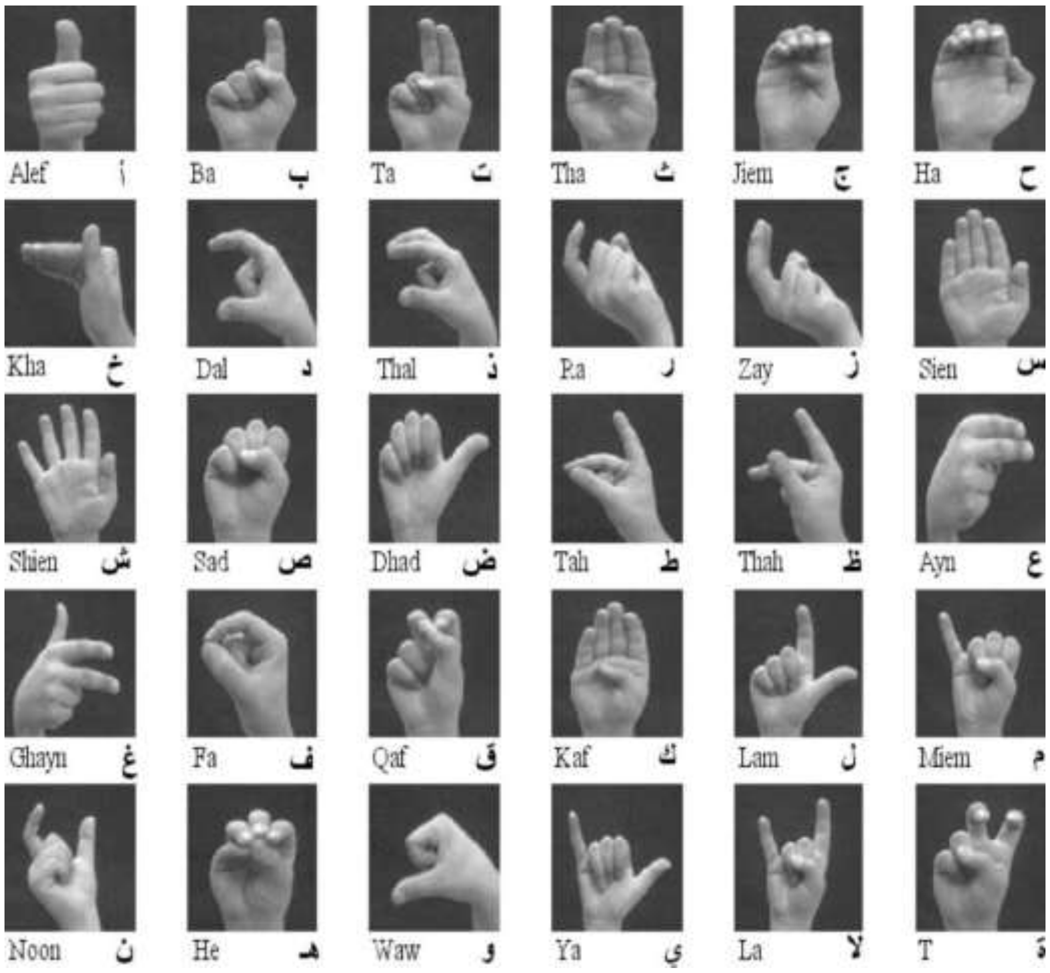


Figure 1—4 : Arabic sign language alphabets

- Sign Gestures in ArSL are a combined of three recognition phases: the first is alphabets recognition; the second is recognition of isolated word with one sign, and finally recognition of word that contains continuous signs. This thesis proposes to start with Arabic sign alphabets as the first step in ArSLR problem. In addition to that, the limited time, the limited resources, and the huge database of ArSL signs force us to implement recognition algorithms on a sample of signs to get better results. Gestures used in ArSL Alphabets as shown in figure 1, which only used to search and investigate the performance of the the proposed model. In this database, the signer perform either static signs that represents the letters and other words or perform non static signs(two sings or more)that means words in ArSL dictionary. Static sign represent 28 letters and the definition tools in Arabic such a(ال) . In this section, we will discuss several methods for image-based ArSL recognition. A new machine learning model is then proposed to recognize the 28 Arabic alphabet signs.

1.4. Thesis objectives

We investigates a new approach for ArSLR based on Restricted Boltzmann machines (RBMs) and a direct use of tiny images. Our main goals in this thesis are as follows:

- Develop a new gesture recognition system based on new machine learning techniques and a direct use of small images that could discriminate between the Arabic signs of alphabetical letters.
- Ideally, the performance of the proposed model should be proportional to the classification accuracy.
- The proposed system should also demonstrate that Deep Belief Networks (DBNs) coupled with tiny images followed by a simple classifier, can be used as an alternative approach to achieve vision-based gestures recognition for deaf and impaired people.
- Since most of the developed systems to ArSLR are either sophisticated or complex, the proposed model should simplify the overall classification process of ArSL.
- Complete one of the graduation requirements to get the master degree in computer science inshaAllah.

1.5. Major contributions

The major contributions of this research work include the following:

- Create a dataset of Arabic alphabet sign language, which consists of more than 10,000 images. The database was created in collaboration with Red Christian Society in Ramallah.
- Development of new algorithm for Arabic alphabet sign recognition for the created data sets. Recognition stage involves training using 70% of the data set and testing stages using the rest of the data. The performance was evaluated based on that.
- Testing the developed model on ArSL alphabet only.

1.6. Thesis benefits

Developing this system can achieve several benefits for the society in particular; some of them can be summarized as follows:

- The number of specialists in Arabic sign language is very limited; therefore developing such system will definitely help the nonprofessional and non-specialists to communicate with the deaf and impaired people.
- Facilitate the communication process between the deaf people and normal people.
- Improve and enhance the education for deaf people.
- Increase the community of the deaf people so they can communicate and participate with the normal people in different things.

1.7. Thesis constraints

Working on new topics, such as DBNs, ArSLR, and Arabic signs database creation was a big challenge for the student. More precisely, during developing this thesis, several constraints and challenges have faced the researcher, which can be summarized as follows:

- The creation of the dataset was not simple, we have arranged several meetings with the Red Christian Society in Ramllah to create the database. In other words, we spend three months to create about 10,000 images for the 28 alphabetic characters of Arabic language.

- Limited experience in topics related to Arabic Sign Language and its features.
- Limited experience in image processing and in particular in the recognition techniques and classification algorithms that have been used in image recognition. In addition to that, pre-processing the database to be appropriate for the features extraction process was not simple, where we have tried different methods of data whitening and normalization techniques.
- Limited experience in machine learning methods and in particular the Deep Belief Network, which is still a new machine learning method and a lot of research works can be developed and investigated based on this approach.

1.8. Thesis organization

The rest of the thesis is organized as follows. In chapter 2, we first introduce the problem of ArSLR in detail and then present background information about the existing ArSLR approaches.

Energy-based models, such as Boltzmann Machines (BMs), RBMs, and DBNs, are also described in chapter 3. This chapter also presents the Constructive Divergence (CD) learning algorithm for training RBM models. Then, it presents a theoretical background for Support Vector Machine SVM and Softmax classifiers. Finally, it concludes towards the choices of the proposed model for an ArSLR task.

In chapter 4, we used the created database of Arabic sign alphabets to experimentally investigate several parameters and factors that play important roles on the classification performance. We also study the role of normalization on the selection of spatial frequencies in the initial image set. The chapter concludes that DBNs can capture a set of high-level interesting features, and thus their use in image coding could simplify the problem of ArSLR.

In chapter 5, we present our conclusions and suggestions for future research. A number of directions for future development of ArSLR using DBNs are proposed.

2. Chapter 2

Background and related work

2.1. Introduction to Gesture Recognition

In the daily life, we use hand gestures to express the meaning of something in a natural communication way, especially among people who have some difficulty in speaking or hearing. Gesture recognition is a computer science topic that aims to interpreting human gestures using mathematical algorithms. Hinton and Lin Shao defined gesture recognition as master field in Human Computer Interface (HCI) technology, which means identification, and recognition of gestures originates from any type of body motion especially face and hand motions [11]. This technique widely used to help impaired people to interact with hardware devices like computers without any interventions of mouse, keyboards and mikes.

The main goal of gesture recognition is to translate a specific body gesture into a specific task using a computerized system. This can be as a control device system that depends only on gestures. In this technology, a camera captures the movement of the human body and transfers the image as an input to the control devices or applications in the computer. Gesture recognition is a topic under HCI field in computer science that achieves the interpreting human gestures via mathematical algorithms.

Gesture recognition can be achieved by either glove-based system or vision-based gesture system. In this thesis, we will consider vision-based gesture system that depends on camera-based data collection. Generally, most of the vision-based hand gesture recognition systems consists of several phases, which are different from one application to another.

Phase one is data acquisition, which includes data collection of hand gestures as a dataset that can be used later to develop the system. This data can be collected using sensors or camera technology.

Phase two is to make several pre-processing steps on the collected data to improve and enhance the quality of the images and highlight their features.

Phase three is feature extraction, which transforms the enhanced data into a set of sparse features or derived values that can be used finally to perform further tasks, like image classification or gestures recognition.

The main challenge of hand gesture recognition is to perform image segmentation process, where hand segmentation process has to be accurate. In other words, hand gesture recognition has been taken under several conditions of light, position and high camera resolution. In addition to that, hand gestures are so similar in their shapes to each other, which makes the recognition process is more difficult in some cases. Therefore, any hand gesture recognition system has to address and pre-process such challenges.

Gesture recognition has widely been investigated; however, most of the research works have been focused on "Latin" languages, and few of them focused on Arabic language. We have also seen that current research in ArSLR has only been satisfactory for alphabet recognition with accuracy exceeding 98%. Regarding that it only focuses on the medium size vocabularies with less than 300 signs, which can performed using static gesture only. Continuous ArSLR is still in its early research stages. A detail description of Arabic sign language is introduced in the following section.

2.2. **Introduction to ArSL**

Traditionally, there are three vocabulary levels of ArSLR systems, which can be summarized as follows: alphabets, isolated words, and sentence level recognition. In this thesis, we propose **a new machine learning method** that addresses the first level of ArSLR, i.e. a vision-based ArSL alphabet recognition system is proposed. Therefore, this thesis concentrates on describing the Arabic alphabets that have been used in sign language recognition systems using different approaches. Under this scenario, the signer performs each letter separately, where the 28 letters alphabets used for ArSL represented by a static posture and the vocabulary size is limited as shown in figure 2.1. Even though the Arabic alphabet only consists of 28 letters, the ArSl ses 39 signs, where the eleven additional signs represent basic signs combining two letters. For example, the two letters ”ال” are quite common in Arabic (similar to the article ”the” in English language). Therefore, most literature on ArSLR uses these basic 39 signs.

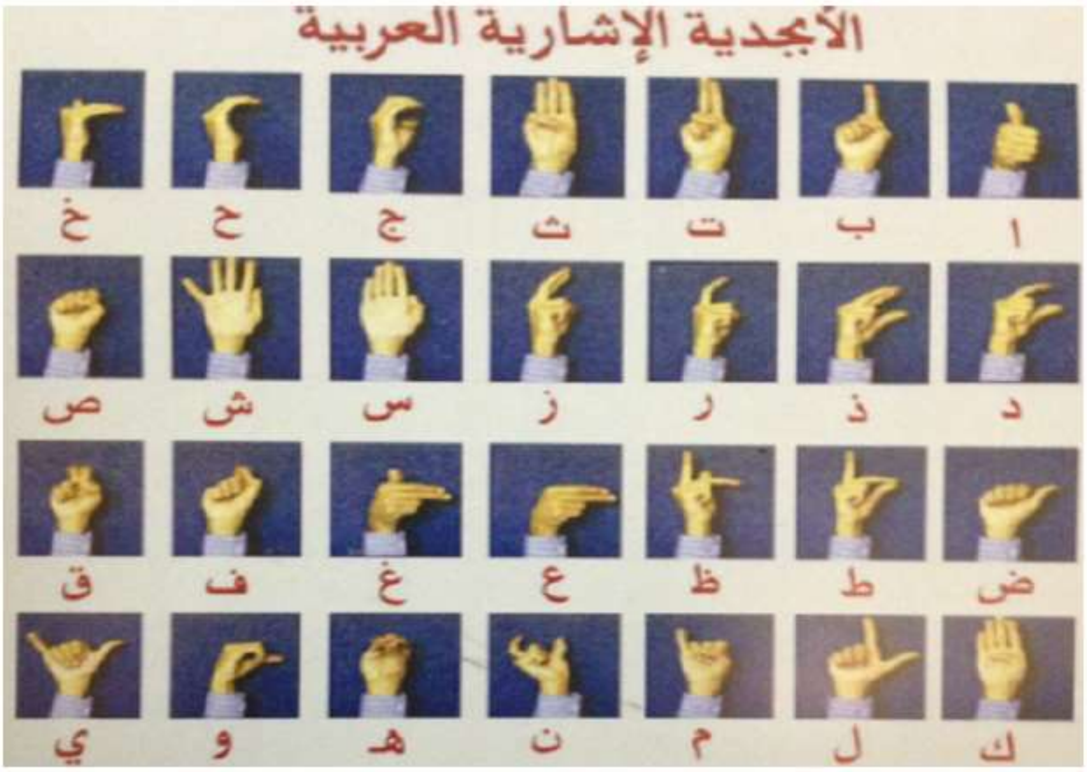


Figure 0—1 Arabic sign language alphabets

In the following section, some of the current approaches that have been developed to achieve Arabic sign language alphabet recognition are introduced and discussed in more details.

2.3. Current approaches to ArSL

Although most of the proposed approaches to the problem of ArSLR have given rise to sensor-based techniques, some others are based on vision-based techniques. The task of ArSLR usually requires firstly to produce an appropriate code for the initial data and secondly use this code to classify and learn the alphabets.

Sensor-based model, which usually employs sensors attached to the hand glove. Look-up table software is usually provided with the glove to be used for hand gesture recognition. Some sensor-based models have been recently developed can be seen in [12]–[15] for instance. However, the vision-based model, which usually uses video cameras to capture the movement of the hand. As mentioned before, image-based techniques exhibit a number of challenges, including lighting conditions, image background, face and hands segmentation, and different types of noise. A typical image-based recognition system consists of 5 stages: image acquisition, pre-processing, segmentation, feature extraction, and classification. In the following sub-sections, we make a survey of the different coding and learning approaches that have been used in the context of ArSLR.

2.3.1. Classification methods

In this work, we used the terms "recognition" and "classification" to speak about the process of "instance recognition" or "identification". So, it is important to underline that we are interested in instances recognition as presented in [16], and not in concepts recognition which is known as "categorization". A category or a class represents a set of entities grouped together under one or more common characteristics. For example, the books are categorized into beginner and advanced, while an instance recognition or identification is the process by which an entity is identified in an image, with respect to the objects, the viewing angle, brightness, etc. Note that the Recognition term is used in this thesis to generically refer to the problem as a whole, which represents the classification process itself.

As stated in [16], the goals of machine learning is to extract and learn a set of patterns and features from a knowledge dataset and thus use the extracted features to predict seen and unseen patterns/objects. Many machine learning approaches have been already applied to different classification problems such as optical character recognition, face detection, speech recognition, traffic sign recognition, autonomous, …etc [16]–[20].

Machine learning is a branch of Artificial Intelligence and it focuses on the statistical nature of learning. During our research works, we have seen that different people have defined machine learning in different ways. For example, Thomas Mitchell has defined machine learning as a computer program which learns from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E [21]. Arthur Samuel has also defined machine learning as a field of study that gives computers the ability to learn without being explicitly programmed [22]. As mentioned before, the fundamental goal of machine learning is to develop learning algorithms, which can extract and exploit the statistical relationships that are presented in the input image. These learning techniques can be organized in three main classes: supervised learning algorithms, unsupervised learning algorithms, and semi-supervised learning algorithms. In supervised machine learning approaches, the learning process is usually performed with the presence of an “expert”, teacher, or knowledge of output. In this case, a generated function that labels the inputs to desired output, where human experts as stated earlier provide the labels “right answers”. Some examples of supervised learning algorithms are: Neural Network [23], Support Vector Machines [24], Decision Trees [25], Bayesian Classifiers [26],.. etc. Unsupervised machine learning methods usually model a set of inputs by themselves, so that the learning procedure does not include any knowledge or experience about the output class, where the data is unlabeled. Some unsupervised learning algorithms are: genetic algorithms [27], k-Nearest Neighbor algorithm (k-NN) [28], clustering approaches [29], etc. Semi-supervised learning techniques typically learn from a combination of a limited set of labeled data with a large amount of unlabeled data which can be inexpensive to generate. For instance, semi-supervised and transductive SVM [30], [31] and co-training [32] are two examples of semi-supervised learning algorithms.

Supervised discriminative approaches have been used to distinguish between the different classes. In other words, they can be used to discriminate between *n* different classes in the representation data space Y, by finding the “decision frontier” that will assign each class to each point of the data space. To achieve that goal, it is possible to use discriminative or generative machine learning approaches. The Generative methods aim at first finding the optimal model that explains and fits the original data and then, using the generated model, find the frontier between the data, some generative approaches including Naïve Byes classifiers, Bayesian filtering techniques, hidden Markov model, & Markov-Chain Monte Carlo. While, the Discriminative methods aim at directly finding the best way to separate the classes (they directly search for the decision frontier). Some discriminative approaches including: neural network classifiers, support vector machine approaches, & softmax regression. The most generative and discriminative approaches that have been used in recognition will be illustrated later in this chapter.

Different classification approaches (generative & discriminative methods) have been already developed and used to achieve ArSLR, for instance see [33]–[36]. In particular, the authors in [9] developed a neuro-fuzzy system. The proposed model includes five main stages including: image acquisition, filtering, segmentation, hand outline detection followed by features extraction. The experiment considered the use of the bare hand and achieves hit rate of 93.6%. The author in [8] has also introduced an automatic recognition of the Arabic sign language letters. For feature extraction, Hu's moments are used and for classification, the moment invariants are fed to SVM. A correct classification rate of 87% was achieved.

The authors in [11] used a polynomial classifier to recognize alphabet signs. It is a glove-based experiment with different colors for the fingertips and wrist region. Length and angels and other geometric measures considered as features. The hit rate was about 93.4% representing 42 gestures but with about 200 samples only. In [12], the authors proposed to use recurrent neural networks for alphabet recognition. Two different signers used to build a database of 900 samples representing 30 different gestures. Colored gloves similar to the ones in [11] were used in their experiments. This proposed model achieved an accuracy rate of 89.7% while a fully recurrent network improved the accuracy to 95.1%.

Therefore, Gesture recognition systems are either glove-based witch depends sensors to collect data, or free hand based if no gloves or sensors used. We have seen that most of the learning and recognition methods have been focused on using neural networks, K –nearest neighbor, support vector machine, and hidden Markov model. The recognized data have to be classified in to sections or sectors and need to be labelled to a specific gesture using a simple classifier like softmax regression. In the following section, we introduce a detailed description of these classification methods.

2.3.1.1. K- Nearest Neighbour Rule.

K- Nearest Neighbor rule (K-NN) is one of the simplest supervised machine learning methods; it is a type of instance or lazy learning [37]. Classification to a certain object can be done by assigning the object to the most common class amongst its K-NN, which is a positive small integer.

We illustrate this technique on the Iris data. Suppose a new Iris to be classified. The idea is most likely near to observations from its own proper population. So we look at the nearest five observations from all previously recorded Irises, and classify the observation according to the most frequent class among its neighbors.

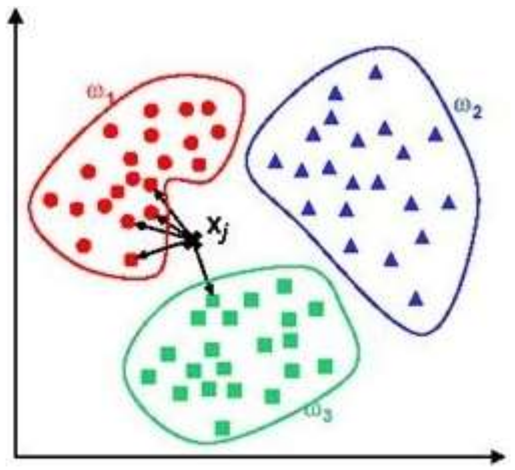


Figure 0 : 2—K-Nearest Neighbor Algorithm example.[38]

As shown in figure 2.2, the K-Nearest rule is used in statistical pattern recognition. Suppose the library contains feature vector for K-sample, each sample is represented by *i* features. The algorithm of K-NN is summarized by:

1) Positive integer *k* is declared along with the new sample.
2) Choose the *k* entries in the database; it should be closest to the new sample.
3) Find the most common classification of these entries.
4) Give this classification to the new sample [38].

In other words, the matching process can be performed as follows:

1- Calculate the distance between the test feature vectors and each feature vector in the library as follow:

$$d_i = \sqrt{\sum_{j=1}^{n}\left(X_j - T_j\right)^2} \qquad (2.1)$$

Where:

i represents number of samples.

n represents number of coefficients.

X_j represents the coefficients for each sample.

T_j represents the coefficient for test.

2- Compare the result distance. The test sample is consigned to the sample creating the smallest distance. For example, using the K-Nearest neighbor rule. Let us assume that the library contains the following feature vectors:

Sample 1 $\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$

Sample 2 $\begin{pmatrix} 2 \\ 3 \\ 4 \\ 5 \end{pmatrix}$

Sample 3 $\begin{pmatrix} 4 \\ 6 \\ 8 \\ 10 \end{pmatrix}$

Sample 4 $\begin{pmatrix} 2 \\ 4 \\ 6 \\ 8 \end{pmatrix}$

Sample 5 $\begin{pmatrix} 1 \\ 2 \\ 4 \\ 8 \end{pmatrix}$

Test $\begin{pmatrix} 2 \\ 5 \\ 8 \\ 11 \end{pmatrix}$

Solution:

$$d_1 = \sqrt{(1-2)^2 + (2-5)^2 + (3-8)^2 + (4-11)^2} = 9.16$$

$$d_2 = \sqrt{(2-2)^2 + (3-5)^2 + (4-8)^2 + (5-11)^2} = 7.48$$

$$d_3 = \sqrt{(4-2)^2 + (6-5)^2 + (8-8)^2 + (10-11)^2} = 2.45$$

$$d_4 = \sqrt{(2-2)^2 + (4-5)^2 + (6-8)^2 + (8-11)^2} = 3.74$$

$$d_5 = \sqrt{(1-2)^2 + (2-5)^2 + (4-8)^2 + (8-11)^2} = 5.92$$

Result:

The test is similar to sample # 3, because the distance between the test and sample 3 is smaller than the other distances.

K-NN makes prediction just in time by calculating the similarity between an input sample and each training instance with no learning to any models [38]. The K-NN Classifier is therefore a simple classifier that works well on basic recognition problems, however it can be slow for real-time prediction if there are a huge number of training sets and is not robust to noisy data. In [4] the researcher himself mentioned that K-NN algorithm is not recommended in ArSL because after examination the hit rate of each sign did not exceed 50% for most signs. Not only the researcher mentioned that K-NN is very simple to handle ArSLR but also he recommended the use of enhanced recognition algorithms such as neural networks and fuzzy logics.

2.3.1.2. Hidden Markov Model

Hidden Markov Model (HMM) is probabilistic sequence classifier whose function is to assign label to each object in this sequence. Given a sequence of objects, letters or words. It computes the probability distribution all over the possible sequence of labels and choose the best label of sequence. The main task of HMM is to sequence labelling objects and classify it, for example, the word of speech is a sequence of words that HMM can label every word on it to the categories Noun, Verb… etc.

As shown in figure 2.3, HMM is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (i.e. hidden) states. The HMM can be represented as the simplest dynamic Bayesian network.

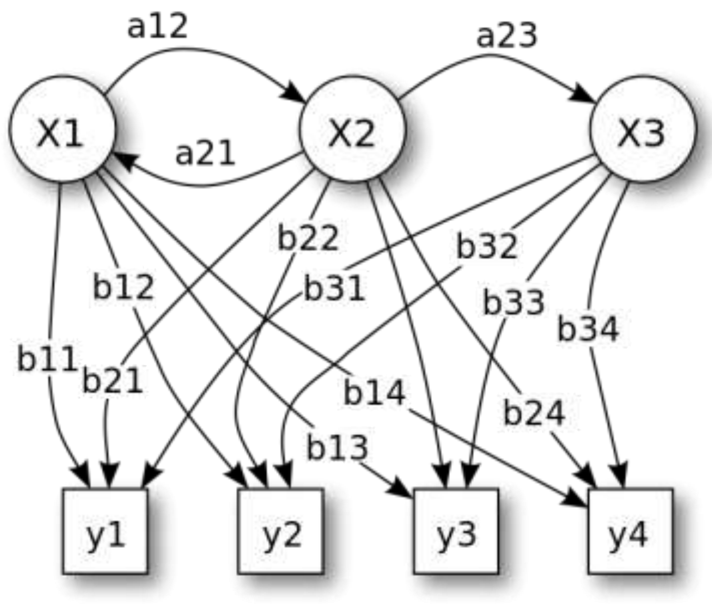


Figure 0—3 Probabilistic parameters of a hidden Markov model (an example), where X represents the states, y represents possible observations, a represents the state transition probabilities, and b represents the output probabilities

HMM is one of the most important machine learning models, it has been used in speech recognition and language processing [39], [40]. Marcov chain is defined by a set of objects and a set of connections between these objects, with each connection associated with a weight. In Marcov chain the weights are probabilistic that is the sum of all probabilities in all connections must be one. Marcov chain job is to assign probability to each sequence.

To better understand the HMM model, let us assume we want to predict the weather using Hidden Marcov Model, the model have two states sunny and rainy. Let us assume that after a rainy day the next day will be sunny with a probability of 30%. The day after the sun day will be rainy with the probability of 40% . The state of the example is shown below:

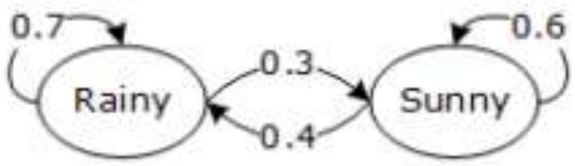


Figure 04—HMM Chain to predict the weather (Bishop, 2006).

One most important to know in HMM for the previous example is the current state that contains all information about previous observation. Now assume that a prisoner is locked in a basement with no windows, the only way to know whether it is sunny or rainy is to lock on the jailer shoes, if jailer shoes are dirty then it is 90% rainy and if it was clean then the weather is sunny of 40% chance.

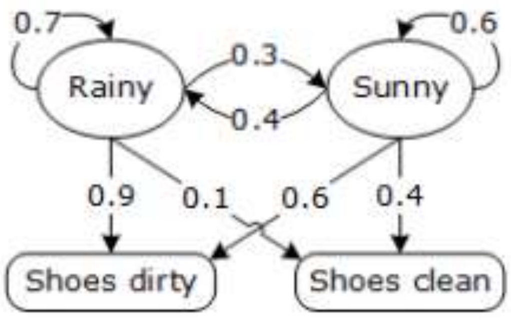


Figure 05—: HMM Chain for weather prediction (Bishop, 2006).

As mentioned earlier, HMM can be used in speech recognition tasks and applying in pattern recognition like hand and face recognition, see for instance [41]. The author in use this model to recognize some Arabic words from signing it using hand sensor glove, then assign each sign to its related word [42]. The advantages of HMM is that it has strong statistical foundation and can handle inputs of variable length they are the most flexible generalizations of sequence profiles. Moreover, it can be combined in to libraries and allowed consistent treatment of insertion and deletion objects in the form of locally learnable. However, the disadvantages of such models can be summarized as follows:

a) It has a large number of unstructured parameters and it cannot express dependencies between hidden layers [43].
b) The types of prior distributions that can be placed on hidden states are severely limited.

c) It is not possible to predict the probability of seeing an arbitrary observation. This second limitation is often not an issue in practice, since many common benefits of HMM's do not require such predictive probabilities.

2.3.1.3. Support Vector Machine

SVM is a supervised statistical learning algorithm 1. It can be used to train the data and predict the patterns of the data to create a "decision-maker". Figure 2.10 illustrates the high-level view of how we perform these tasks using an SVM approach, where the input corresponds to the dataset and the output results correspond to the classification of the data which is used for testing. More precisely, using this algorithm we perform two different tasks as shown in figure 2.6:

- **Learning**: by training the input examples (data) using SVM-train function.
- **Prediction**: new examples are used for testing. Usually, if the problem task is to classify the observations in a set of finite labels, the task is said to be a classification task.

The SVM is a discriminative model able to construct a hyperplane or a set of hyperplanes in a high-dimensional space and can be used for further tasks such as classification. SVMs methods were first proposed by [24] to find the decision rules based on the feature space to discriminate the different classes. It has been shown that this approach has a good performance on character recognition, text classification, and face recognition [44].

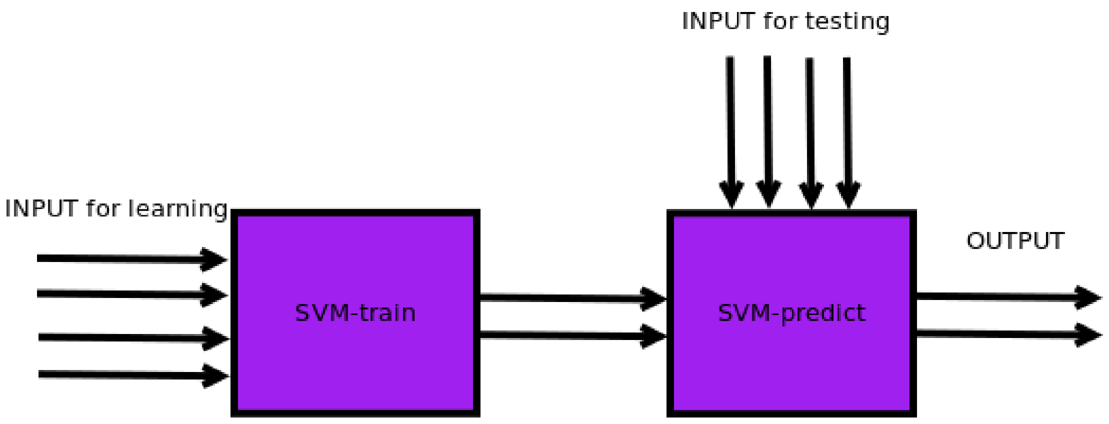


Figure 0—6 : High-level structure of an SVM approach.

SVMs depends on the rule of decision planes that gives accurate decision boundaries. A decision plane job is to separate between a set of objects with different class memberships. A schematic example is shown in figure 2.7 below. In following example, the objects is either in GREEN or RED classes. The line defines a boundary on the both sides, the GREEN objects on the wright side and the RED objects on the left side. Any new object is then classified and labelled to any of the two classes.

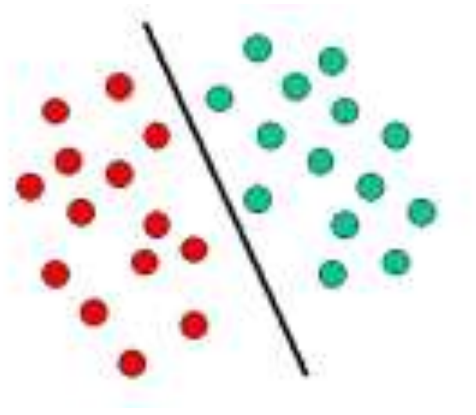


Figure 0—7 : linear SVM.

The figure above showing an example of a linear classifier. There are different classifications that might be little sophisticated structures needed to do the optimal separation; classifying new objects referring to the examples are available. This situation is depicted in figure 2.8 in the next page. Compared to the previous figure, the separation between these two classes required a curve (non-linear). This kind of classification and separation tasks, which draw separating curves to distinguish between objects in classes, is called hyper-plane classifiers. Support Vector Machines are suitable to manage such tasks.

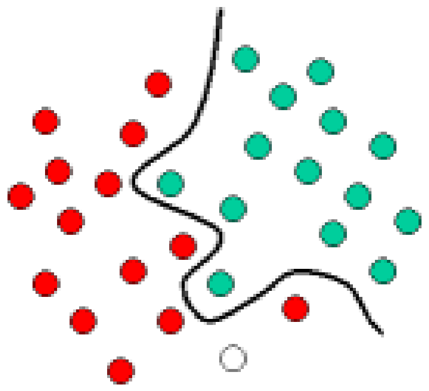


Figure 0—8 : non-linear SVM.

Figure 2.9 below shows the basic idea behind Support Vector Machines. Here we see the original objects (left side of the schematic) mapped, i.e., rearranged, using a set of mathematical functions, known as kernels. Mapping is the process of arranging the objects. In the new setting, mapping objects is the solution instead of the complex non-linear curve. What we have to do is to find the best line that separates the two classes.

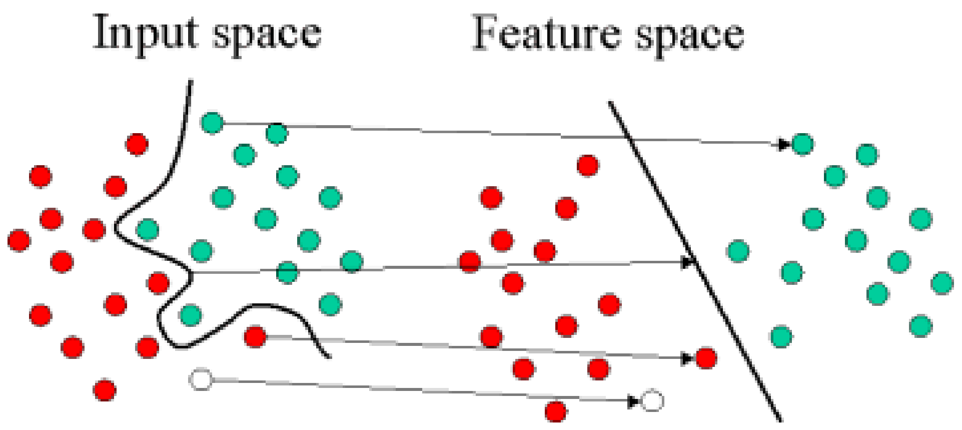


Figure 0—9 : SVM mapping.

Support Vector Machine (SVM) is primarily a classification method that performs classification tasks by constructing hyper planes in a multidimensional space that separates cases of different class labels. SVM is suitable for classification and regression and can handle multiple variables. A detailed description of SVM applications can be found in [44].

As illustrated in [45], the SVM advantages can be summarized as follows:

- SVM is good if we do not have any idea about the data.
- SVM can handle unstructured and semi- structured data such as images and text.
- With an appropriate kernel function, we can solve any complex problem;
- Neural network is solved for local optima but SVM is not.
- It scales relatively well to high dimensional data.
- The risk of over fitting in SVM models are less.

While the SVM disadvantages can summarized as follows:

- It is not easy to choose an optimal kernel function.
- Long training time for large datasets.
- In SVM it is hard to understand the variable weights of the final model.

- Since the final model is not so easy to see, we cannot do small calibrations to the model hence it is tough to incorporate our business logic.

2.3.1.4. Softmax regression

Softmax regression model generalizes logistic regression to classification problems where the class label can take on more than two possible values. This will be useful for such problems as ArSLR sign classification, where the goal is to distinguish between 28 different alphabetic characters. This model is a supervised machine-learning algorithm, which can be used to predict of the probability of occurrence of an event based on the input data. In statistics, this model is a probabilistic, linear classifier (for instance see figure 2.9).

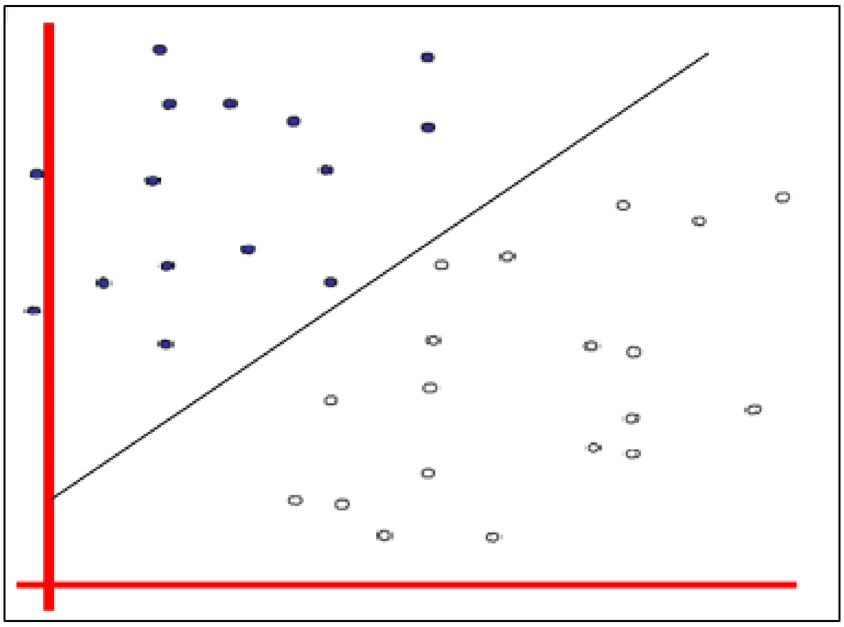


Figure 0—10 : Soft max linear classification for two different classes

As illustrated in [46], in logistic regression, the training set $\left\{\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right\}$ of *m* labelled examples, where the input features are $x^{(i)} \in \Re^{n+1}$, where with the logistic regression, we have a binary classification setting, so the labels were $y^{(i)} \in \{0,1\}$, and our hypothesis took the form:

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}} . \qquad (2.2)$$

This function can be used as an "activation function" for a mathematical model of a neuron and it is shown in figure 2.10 and the model parameters θ were trained to minimize the following cost function:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta\left(x^{(i)}\right) + \left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)\right]$$

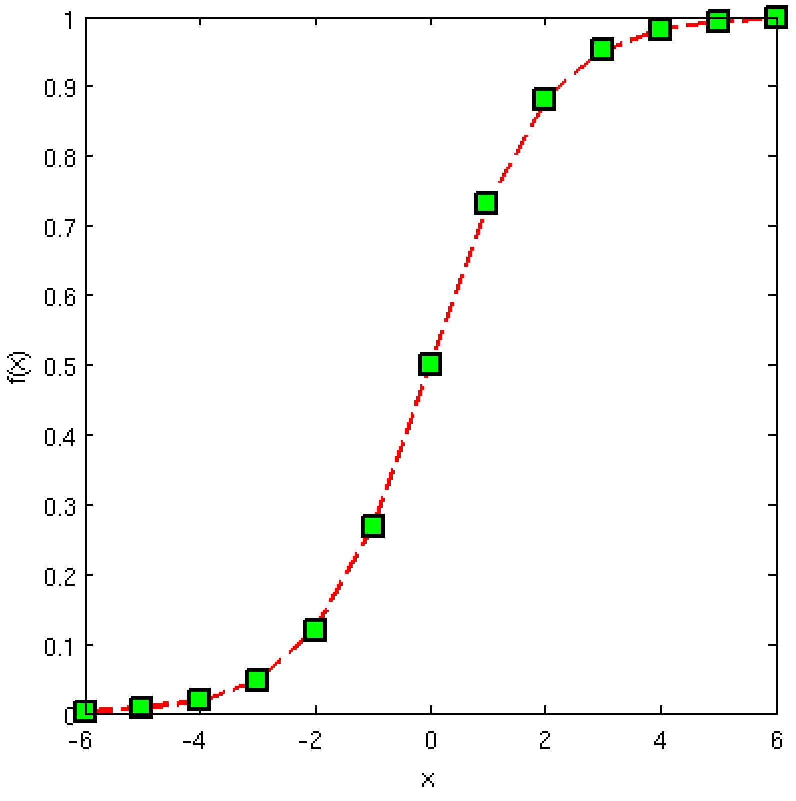


Figure 0—11 : Logistic sigmoid activation function: with .

However, in the softmax regression setting, we are interested in multi-class classification, and so the label y can take on k different values, rather than only two classes as in the binary classification. Therefore, in our training set $\left\{\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)\right\}$, we now have that $y^{(i)} \in \{1, 2, \ldots, k\}$. Therefore, in the ArSLR task for instance, we would have k = 28 different classes.

Given a test input x, we want our hypothesis to estimate the probability that $p(y = j \mid x)$ for each value of $j = 1, \ldots, k$. In other words, we want to estimate the probability of the class label taking on each of the k different possible values. Thus, our hypothesis will output a k dimensional vector (whose elements should sum to 1) giving us our k estimated probabilities. Concretely, our hypothesis $h_\theta\ x$ takes the form:

$$h_\theta\left(x^{(i)}\right)=\begin{bmatrix} p\left(y^{(i)}=1\,|\,x^{(i)};\theta\right) \\ p\left(y^{(i)}=2\,|\,x^{(i)};\theta\right) \\ \vdots \\ p\left(y^{(i)}=k\,|\,x^{(i)};\theta\right) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}\begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix} \qquad (2.3)$$

Here $\theta_1,\theta_2,\ldots,\theta_k \in \Re^{n+1}$ are the parameters of our model. Notice that the term $\frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}$ normalizes the distribution, so that it sums to one as mentioned earlier.

For convenience, we will also write θ to denote all the parameters of our model, which are given by the following matrix:

$$\theta=\begin{bmatrix} \theta_1^T \\ \theta_2^T \\ \vdots \\ \theta_k^T \end{bmatrix}$$

Like in the logistic regression, the parameters of a softmax regression can be learned minimizing the following cost function:

$$J\left(\theta\right)=-\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k}1\left\{y^{(i)}=j\right\}\log\frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}\right] \qquad (2.4)$$

The idea behind this model is to minimize this cost function $J\ q$ by changing the model parameter q. This can be achieved using a gradient descent procedure

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$ for all *j*, where θ is the learning rate of the model. The partial derivative of the cost function is

$$\frac{\partial}{\partial \theta_j} J(\theta) = \nabla_{\theta_j} J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[x^{(i)} \left(1\left\{ y^{(i)} = j \right\} - P\left(y^{(i)} = j \mid x^{(i)}; \theta \right) \right) \right] + \lambda \theta_j , \quad (2.5)$$

where the conditional probability of a given class with respect to the model parameters is

$$P\left(y^{(i)} = j \mid x^{(i)}; \theta \right) = \frac{e^{\beta \theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\beta \theta_l^T x^{(i)}}} . \qquad (2.6)$$

It was stated that the use of a Softmax regression depends on the assumption of obtaining the linear separation of the data using DBNs. However, if this assumption is false, a nonlinear classifier, like SVM, is required to perform the classification process. Therefore, in this thesis, for the classification process we plan to first use a Softmax regression and then use nonlinear SVM classifier to perform the classification process, similar results will underline that DBNs has managed to extract sparse features and thus using these features to code the input data to become linearly separable.

2.3.1.5 Neural networks

During our research study, we have seen that most of the existing discriminative approaches have focused on Neural Networks (NN) [47] and SVM models [24]. We have noticed that these two approaches are based on a scalar product. More formally, NN are based on the scalar product in the data representation space, xi, i.e. to measure the projection of one vector onto another. To illustrate that, see for instance the following example shown in figure 2.11.

Where the unit v_i has an activation function, a_i, given by the following equation:

$$a_i = \sum_{j=1}^{n} w_{ij} x_j = \langle w, x \rangle \qquad (2.7)$$

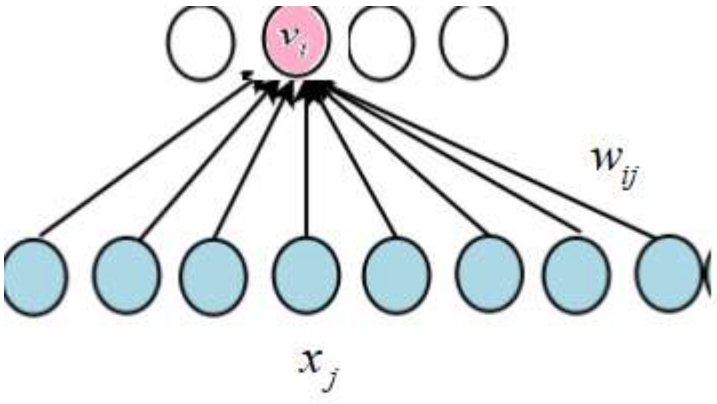


Figure 0—12 : An example for scalar product of neural networks.

where this activation function can be represented as a scalar product of the input vector x_j, and the weights w_{ij}. Besides, this function defines the decision frontier to separate the data into different classes. Similarly, SVM provide an efficient way to map the data into a high-dimensional feature space as explained before. NN and SVM can therefore, be used to achieve ArSLR to distinguish between the 28 alphabetic characters or classes.

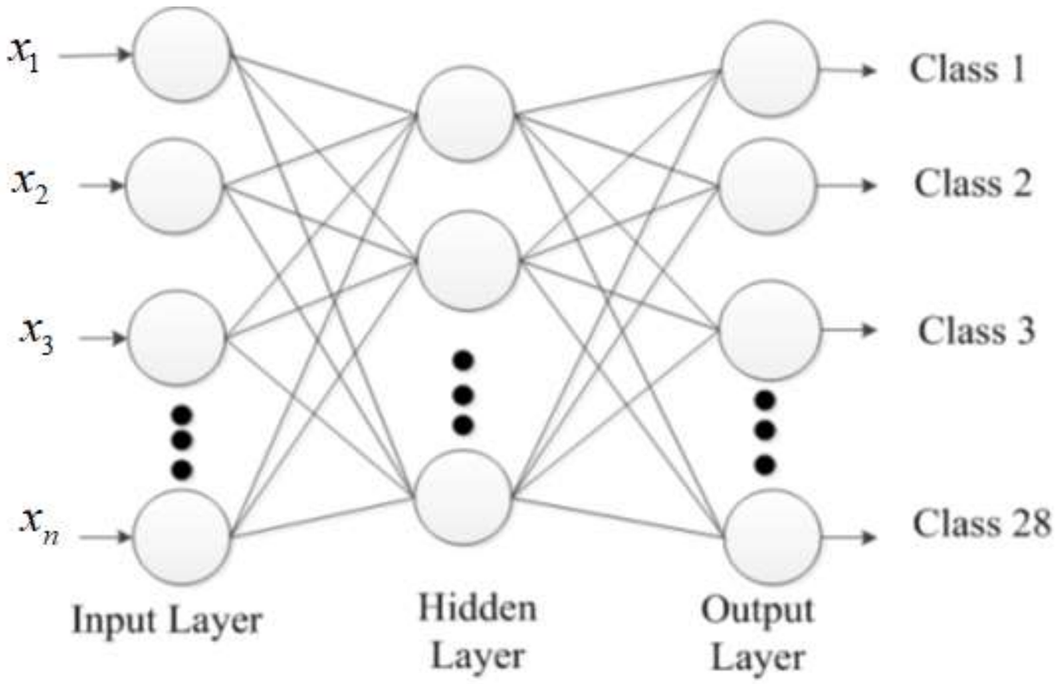


Figure 0—13 : Basic Structure of multilayer neural network.

Neural network is called Multi-Layer Perceptron (MLP). As shown in figure 2.12, the data are feed-forward from the input layer, through at least one hidden layer and then to the output layer. Training MLP usually involves two main issues; weights update and choosing an appropriate network. Concerning the adjustment of weights, the authors in [47] introduced back-propagation as an alternative learning algorithm to Boltzmann Machine for multi-layer neural networks training. Back-propagation is also called a propagation of error, where it is a supervised learning algorithm to teach NN to perform a given task for networks, which have no feedback. A detailed description of back-propagation learning algorithm can be found in [47].

The classification process is based on the outputs of MLP, which are based on the weight w. In other words, it has been shown that it is possible to use the extracted weight features to separate the different classes within the framework of Bayesian theorem, $P(x_t = c \mid y_t; w)$. However, these output units do not sum to 1, and thus cannot be used directly to perform the classification process. To overcome this problem, one way is to use a softmax regression as mentioned before to transform these units into real probability values so that the summation of these values will be equal to 1 for a given observation y_t and it can thus be achieved using the following softmax formula:

$$P(x_t = c_i \mid y_t; w) = \frac{e^{w_i^T y_t}}{\sum_{l=1}^{k} e^{w_l^T y_t}} \qquad (2.8)$$

Using this formula, it is therefore possible to use NN to find the decision frontier to distinguish or separate between the different classes.

After presenting the different existing image-based approaches that have been used to achieve ArSLR, we have noted that these approaches generally include two main phases of coding and classification. We have also seen that most of the coding methods are based on hand-crafted feature extractors, which are empirical detectors. By contrast, a set of recent methods based on deep architectures of neural networks give the ability to build it from theoretical considerations.

ArSLR therefore requires projecting images onto an appropriate feature space that allows an accurate and rapid classification. Contrarily to these empirical methods mentioned above, new machine learning methods have recently emerged which strongly related to the way natural systems code images [48]. These methods are based on the consideration that natural image statistics are not Gaussian as it would be if they have had a completely random structure [49]. The auto-similar structure of natural images allowed the evolution to build optimal codes. These codes are made of statistically independent features and many different methods have been proposed to construct them from image datasets. Imposing locality and sparsity constraints in these features is very important. This is probably due to the fact that any simple algorithms based on such constraints can achieve linear signatures similar to the notion of receptive field in natural systems. Recent years have seen an interesting interest in computer vision algorithms that rely on local sparse image representations, especially for the problems of image classification and object recognition [16], [50], [51]. Moreover, from a generative point of view, the effectiveness of local sparse coding, for instance for image reconstruction [52], is justified by the fact that a natural image can be reconstructed by a smallest possible number of features. Research ends up that Independent Component Analysis (ICA) produces localized features. Besides, it is efficient for distributions with high kurtosis well representative of natural image statistics dominated by rare events like contours; however, the method is linear and not recursive. These two limitations are released by DBN [19] that introduce nonlinearities in the coding scheme and exhibit multiple layers. Each layer is made of a RBM, a simplified version of a Boltzmann machine proposed by Smolensky [53] and Hinton [54]. Each RBM is able to build a generative statistical model of their inputs using a relatively fast learning algorithm, Constructive Divergence (CD), first introduced by Hinton [54]. Another important characteristic of the codes used in natural systems, the sparsity of the representation [48], is also achieved in DBN. Moreover, it has been shown that these approaches remain robustness to extract local sparse efficient features from tiny images [55]. This model has been successfully used in [16] to achieve semantic place recognition. The hope is to demonstrate that DBN coupled with tiny images can also be successfully used in the context of ArSLR.

2.4 Deep learning approaches

DBN is a deep learning approach and is made of an RBM, a simplified version of a Boltzmann Machine proposed by Smolensky [53] and Hinton [54]. Each RBM is able to build a generative statistical model of their inputs using a relatively fast learning algorithm CD which is first introduced by Hinton [54] which will be explained later. Another important characteristic of the codes used in natural systems, the sparsity of the representation [48] is also achieved in DBN.

Recently, deep learning approaches, like DBN, have been indeed become popular in extracting a set of independent features, which can be used later as a powerful way to code the initial data. Deep machine learning approaches have extensively been used in different classification problems (phone recognition, natural language processing, audio processing, hand-written character recognition, robot place recognition, and object recognition), see for instance [14], [16], [17], [20], [51] – [55]. The authors in [40], have shown that DBN are still able to extract sparse efficient features from tiny images 32×32 pixels. These extracted features can thus be used to generate a sparse representation of the initial data. A sparse representation means that an image is coded by the smallest possible number of features which therefore simplifies the overall classification process.

A DBN is typically constructed by multiple layers of RBM stacking. So that, the hidden layer of one RBM becomes the visible layer of another higher RBM layer. DBN is trained in a greedy layer-wise and bottom-up fashion introduced by [34]. A detailed description of RBM learning algorithm will be illustrated later. It has been shown that training DBN in a greedy layer-wise way works better than back-propagation learning algorithm with random initialization [56].

2.4.1 Restricted Boltzmann Machines

A general Boltzmann machine which is a probabilistic generative model introduced by Hinton [57]. As shown in figure 2.13 (left), a classical Boltzmann machine consists of two layers; the visible layer, **v**, and the hidden layer, **h**, and a symmetric interactions between the visible and hidden units that is described by a weighted matrix **W**. In this case, as shown in figure 2.13 (left), the units in each layer are fully connected and are also connected to all other units in other layers. This connectivity complicate the learning process and needs long time for convergence.

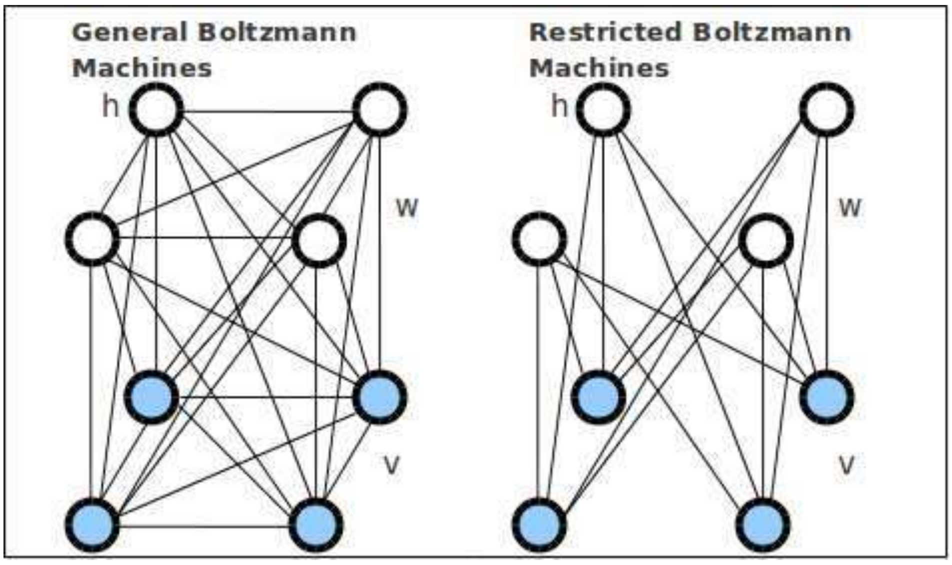


Figure 0—14 : Left figure represent a classical Boltzmann machine, where there is a fully connection between the visible units themselves, hidden units themselves, and between the visible and hidden units. Right figure represent an RBM where there is no direct conn

In 1986, Smolensky introduced RBM [53] as a powerful learning algorithm which simplifies the connectivity between the units as shown in figure 2.13 (right). In other words, in RBM, the visible units are independent given the hidden ones and the hidden units are independent given the visible ones. RBM trains deep networks in a greedy layer-wise fashion, i.e. one hidden layer at a time, which minimizing the following energy function:

$$E(\mathrm{v},\mathrm{h};\theta) = -\sum_i \sum_j v_i h_j w_{ij} - \sum_{i\in \mathrm{v}} b_i v_i - \sum_{j\in \mathrm{h}} c_j h_j \qquad \mathbf{(2.9)}$$

where the visible and hidden layers are fully connected through a set of weights w_{ij} and biases $\{b_i, c_j\}$, and there are no direct connections between units of the same layer. Therefore, according to Gibbs distribution; the probabilities of the state for a unit in one layer conditional to the state of the other layer can therefore be easily computed as follows:

$$P(\mathrm{v},\mathrm{h};\theta) = -\frac{1}{Z(\theta)} e^{-\beta E(\mathrm{v},\mathrm{h};\theta)} \qquad (2.10)$$

where, Z is a normalizing constant. Thus as illustrated in [58]. Thus, after marginalization:

$$\begin{aligned} P(\mathrm{h};\theta) &= \sum_{\mathrm{v}} P(\mathrm{v},\mathrm{h};\theta) \\ &= \frac{\sum_{\mathrm{v}} e^{-\beta E(\mathrm{v},\mathrm{h};\theta)}}{\sum_{\mathrm{v}}\sum_{\mathrm{h}} e^{-\beta E(\mathrm{v},\mathrm{h};\theta)}} \end{aligned} \qquad (2.11)$$

It can thus be easy to write down the conditional probability of a single unit being either 0 or 1 given the states of the other units as follows:

$$P(h_j = 1 \mid \mathrm{v};\theta) = \frac{P(h_j = 1, \mathrm{v};\theta)}{P(\mathrm{v};\theta)} \qquad (2.12)$$

where the probability of hidden binary units given the visible units and model parameters can thus be computed as follows:

$$P(h_j = 1 \mid \mathrm{v};\theta) = \sigma\left(c_j + \sum w_{ij} v_i\right) \qquad (2.13)$$

Once the hidden binary states are computed, we produce a reconstruction of the original patch by setting the state of each visible unit to be one with probability:

$$P(v_i = 1 \mid \mathrm{h};\theta) = \sigma\left(b_i + \sum_j w_{ij} h_i\right) \qquad (2.14)$$

Since binary units are not appropriate for multivalued inputs like pixel levels, as suggested by Hinton [59], in the present work visible units have a zero-means Gaussian activation scheme:

$$P(v_i = 1 \mid \mathrm{h};\theta) \leftarrow \mathcal{N}\left(b_i + \sum_j w_{ij} h_i, \sigma^2\right) \qquad (2.15)$$

Concerning the variance of the noise (sigma square), it is possible to learn it for each visible unit, but this is difficult using CD as it is time-consuming. It is more appropriate to first normalize the data components to have zero-mean and unit variance and then use a unit variance and zero-mean for the Gaussian noise, which has been already illustrated before.

Therefore, the parameters of RBM can be learned from the data using different training techniques. Some of them are: maximizing the log-likelihood, Markov-Chain Monte-Carlo (MCMC) sampling techniques like Gibbs sampling, and CD learning algorithm. However, in the next section we will concentrate on the use of CD as a faster and powerful learning algorithm to maximize the log-likelihood gradient.

2.4.2. RBM learning algorithm

It has been mentioned that there are several ways to learn RBM parameters, some of them are: maximizing the log-likelihood, MCMC sampling techniques like Gibbs sampling, and CD learning algorithm. Therefore, maximizing the log-likelihood of the model is one way to learn RBM parameters in a gradient ascent procedure. So that, the partial derivative of the log-likelihood for an energy-based model can be expressed as follows:

$$\frac{\partial}{\partial\theta}L(\theta) = -\left\langle \frac{\partial E(v,\theta)}{\partial\theta} \right\rangle_{\text{data}} + \left\langle \frac{\partial E(v,\theta)}{\partial\theta} \right\rangle_{\text{model}} \qquad (2.16)$$

where model is an average with respect to the model distribution and data an average over the sample data. Therefore, the energy function of an RBM is given by:

$$E(v,\theta) = \log \sum_h e^{-E(v,h;\theta)}$$

and

$$\frac{\partial E(v,\theta)}{\partial\theta} = \sum_h p(h \mid v;\theta) \frac{\partial E(v,h;\theta)}{\partial\theta} \qquad (2.17)$$

However, computing the likelihood needs to compute the partition function, which is usually intractable. However, Hinton [60] proposed an alternative learning technique called CD. This learning algorithm is based on the consideration that minimizing the energy of the network is equivalent to minimize the distance between the data and a statistical generative model of it. A comparison is made between the statistics of the data and the statistics of its representation generated by Gibbs sampling. Hinton [61] showed that usually only a few steps of Gibbs sampling (most of the time reduced to one) are sufficient to ensure convergence. For an RBM the weights of the network can be updated using the following equation:

$$w_{ij} \leftarrow w_{ij} + \eta[\langle v_i^0 h_j^0 \rangle_{\text{data}} - \langle v_i^n h_j^n \rangle_{\text{recon}}] \qquad (2.18)$$

Where η is the learning rate, v_0 corresponds to the initial data distribution, h_0 is computed using equation 4, η is sampled using the Gaussian distribution in equation 6 and with n full steps of Gibbs sampling, and η is again computed from equation 4.

2.4.3. Deep Belief Network

Deep Belief Network (DBN) is an unsupervised layered machine learning system, each layer in DBN is made of a RBM as shown in figure 2.14, a simplified version of a Boltzmann Machine proposed by Hinton [54]. Deep architecture learning has indeed recently become popular as a powerful way to code data using a set of independent features [16], [62]. It used fast learning technique based on constructive divergence. The two most significant properties of DBN are that there is an efficient, layer-by-layer procedure for learning the top-down, generative weights that determine how the variables in one layer depend on the variables in the layer above.

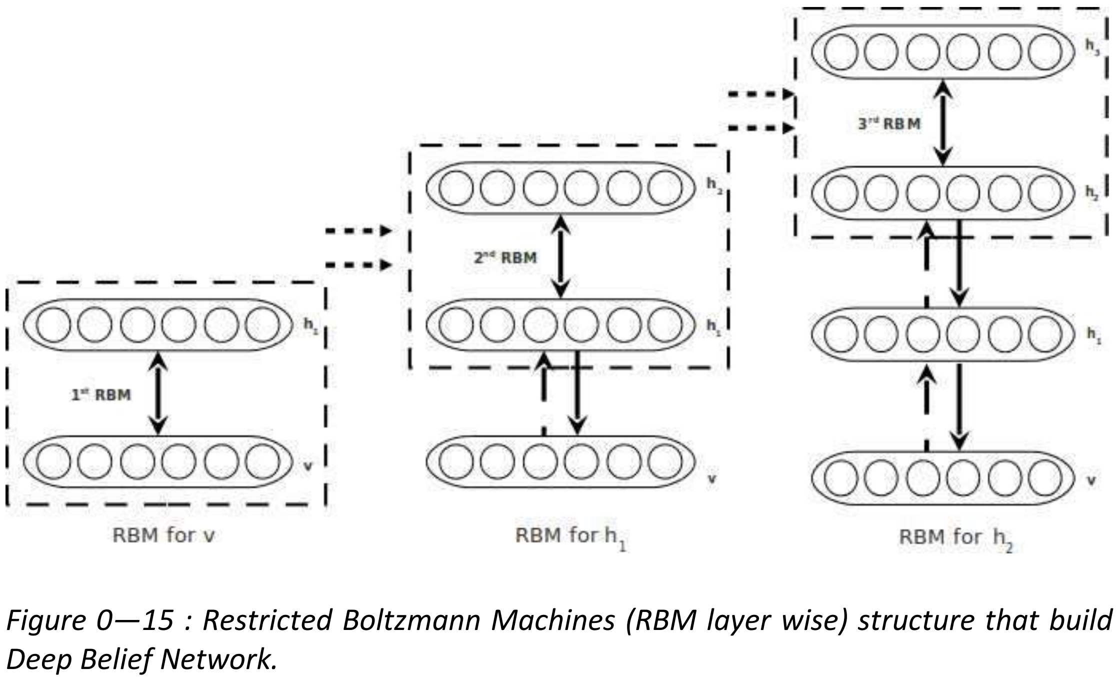


Figure 0—15 : Restricted Boltzmann Machines (RBM layer wise) structure that build Deep Belief Network.

As mentioned earlier, a DBN is a stack of RBM trained in a greedy layer-wise and bottom-up fashion as mentioned before which is introduced by Hinton in 2006 [19]. The first model parameters are learned by training the first RBM layer using the constructive divergence. Then, the model parameters are frozen and the conditional probabilities of the first hidden unit values are used to generate the data to train the higher RBM layers. The process is repeated across the layers to obtain a sparse representation of the initial data that will be used as the final output.

2.5 Summary and recommendations

In this chapter, we introduced the problem of ArSL and illustrated the current approaches that addressed such problem. We saw that most of these approaches are either sensor-based techniques, or vision-based techniques. In particular, the classification methods (K-nearest neighbor rule, hidden Markov model, support vector machines, and sigmoid function) that have been used to achieve ArSL.

These approaches have led to notable successes to achieve vision-based sign language recognition. However, they are based on complex or sophisticated techniques in order to achieve good classification results. In other words, performing the task of ArSLR using a simple classification method is still an open question. Therefore, ArSLR requires an appropriate code that allows fast and robust classification. That is why we presented in this chapter a recent machine learning method DBN as an alternative to hand-designed feature coding approaches. We hope that such approach is suitable for creating an appropriate representation for our classification problem.

3. Chapter 3

Proposed Model

3.1. Introduction

We have discussed several approaches of machine learning models in the previous chapters. In this chapter we introduce the alternative proposed model of AsLR that depends on DBN with RBM.

This chapter focuses on how to create suitable database for 28 Arabic alphabets signs. We will focus on the differences of data preprocessing methods that we used especially in the effectiveness of using normalization and tiny images. Then we are going to present how to extract features from huge dataset using DBNs over RBM training model and clarifying all the factors that play role in training RBM, concerning the conditions required to build an optimal feature space and the optimal values of each factor referring to the position of the model and dataset.

Our aim here is to extract the best features obtained from the dataset after RBM training. We hope that it will be more promising in feature extraction and get high results, due to the statistical independences of the feature and there sparse nature RBM.

Proposed Model.

The methodology of the proposed model contains four steps where each step depends on the previous one:

1. Data collecting and image capturing.
2. Image preprocessing and the use of tiny images.
3. Feature extraction.
4. Gesture recognition.

The few researches that have been done so far on ArSL do not contain enough database to use in our proposed model. In addition, they prohibited to use their own database.   Therefore, we start our model by building a database that contains at least 100 images for each of the 28 Arabic alphabet signs. We adjust this database to several preprocessing techniques to make it suitable for RBM training to extract high quality features [4][5][8]. Then we label them for classifications using SVM technique. Finally, we have to test our model results and give competence between them and all illustrated methods presented in chapter 2[1][2] . . . [14].

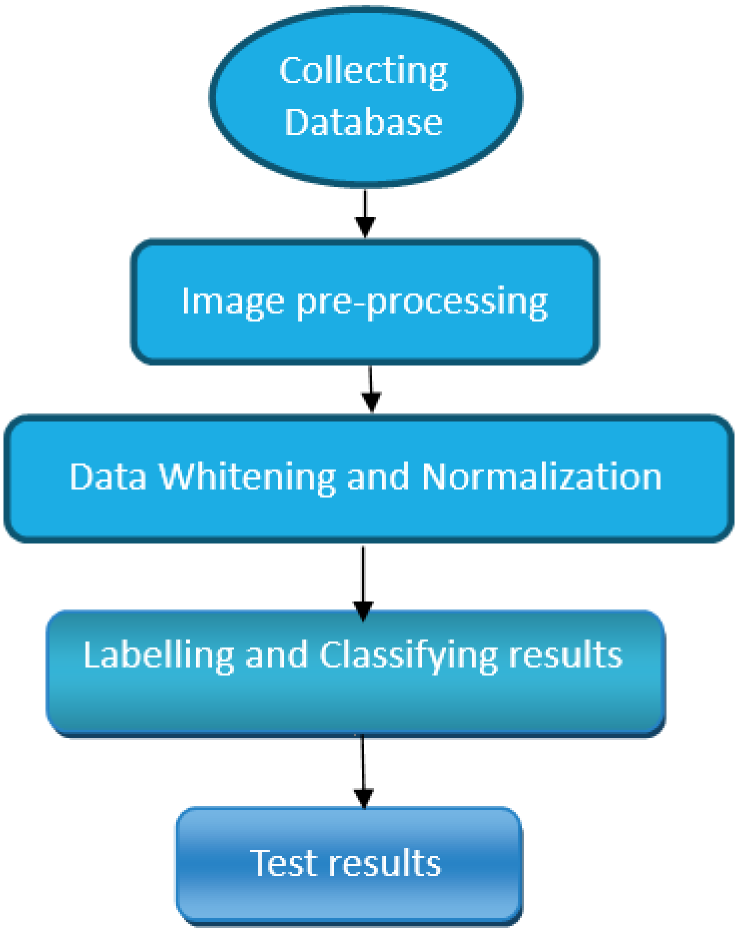

Collecting Database
Image pre-processing
Data Whitening and Normalization
Labelling and Classifying results
Test results

Figure 01—: the proposed model

3.2. The Database

The first step is to collect a dataset for the 28 Arabic alphabet signs. It was difficult due to the lack of studies and resources on ArSL. The absence of published database in the literature review and studies could not help us to easily compare our results with theirs in order to put a hand on weaknesses and strengths of the system. Therefore there is a persistent need of high quality images with certain number of image pauses for each letter [8][16]. Thus our database is a collections of about 360 photo shuts for each sign alphabet. Each sign alphabet collection must obey the following conditions

1) Different orientations

2) Different light and image pauses with the hand centerated

3) All the images have the same height and width (uni-sized)

4) All the images have the same extension. [63]

We built a database consist of 10152 images for Arabic 28 alphabet signs with an average of about 360 images for each sign letter. We took also another data set of 7000 images to test 250 images for each sign letter.

As we mentioned above, we presented the sign letters as a static postures while the words can be presented by one or more consecutive static postures. For example to perform (مرحبا) we need to combine two static postures to perform it [8][10][12]. Hence, most literature in ArSLR uses the 28 letters that can be represented by only one posture [1][3][4][37].

Our dataset contains a huge dataset of more than 10,000 images, while most literature methods of feature extraction and learning handled quite smaller amount of dataset comparing to ours[14][19]. Dealing with huge dataset and patches is a time consuming during feature extraction and learning, and cannot give satisfying results. Researches showed that features extracted by DBNs are more promising for image classification under huge size of image datasets [16][19][54]. For example, using a huge dataset of more than 100,000 images gave highly precise extraction results under DBNs. However to achieve that we need to do some changes to the images of the database in order to make it suitable for DBNs. These changes is called pre-processing that includes changes that could reduce the time and effort of DBNs [29].

Figure 3.2 shows a sample of our dataset, the size of each image is 3840×5760 as it was taken by highly resolution camera. The images contain a lot of hand features, borders, corners and edges as they are created for segmentation. Our mission is to extract small batches that can be used as inputs of DBN and reduce the size of the whole images to an acceptable value that can be directly used as the input of the network.

clearly, we can see 28 Arabic alphabet. Each alphabet is done by different sign (different combination of fingers on a hand) [8][10]. It is obvious that signs are so much similar to each other with a slice difference between the sign of one alphabet and the other. This puts much pressure on choosing a sophisticated machine learning that will extract features and distinguish the difference between the signs [17]. This gives us a count of our proposed model that contains techniques that will raise the extracted feature quality.

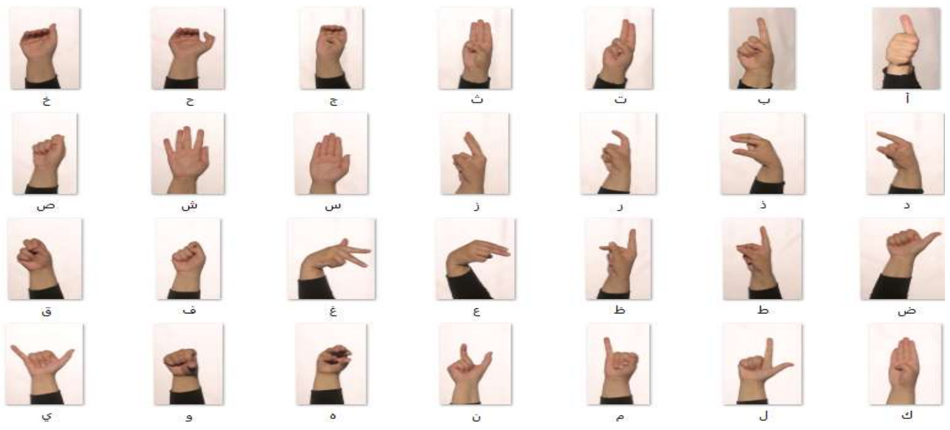


Figure 02. Arabic Alphabet sign language Database used in the proposed model

3.3. Image Pre-processing

This step helps to transfer the images from real to intensity one. Building an effective RBM learning machine requires careful consideration of the type of image pre-processing that reduces the noise of the images and transfer it to low level[16][18].

It is important to do some processing on the original data. The images we have taken is too large in size, high quality coloured and definitely contains some details that must be handled to convert the images from high level to low one .The next sections include image pre-processing phase with the three inner functions: image grey-scaling, image resizing and image normalization.

3.3.1. Image Grey Scaling

Due to researches, grey scaled images are sufficient for feature extraction and learning methods so there is no need to use complicate and hard methods to process colour image[1]…[16]. Instead, grey scaling is used to simplify the image pixels component with less information need to be provided about each pixel, which reduce the colour channels from 3 channels to one channel. Hence transferring the image from high to low level could reduce loading and processing time, maintain the image important information, and structure [11]. (See Fig. 3.3.)

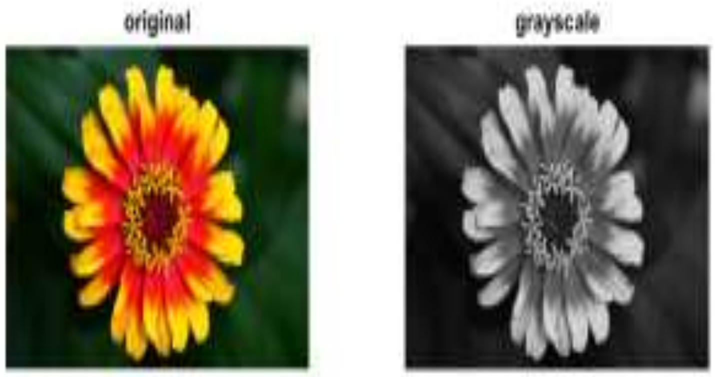


Figure 03—: The difference between colored and grey scaled image

3.3.2. Use of Tiny Images

According to studies, it was found that the input dimension for DBN layer is about 1000 units (for example 32×32 pixels). Smaller patches could give low quality features, while larger patches could be time consuming and heavy load on DBN that results in deep negative impact on the constructive divergence. Thus, to solve the problem we do not need to use the regular image size, instead we try to resize it. Researches showed that resizing images into tiny images gives much satisfying results specially when combined to RBM. Therefore, we can replace the bag of words (huge normal dataset) with tiny images in order to achieve fast classification and recognition process [16][18]. Of course, we want to avoid this bag of words because it cannot be easily handled by any classification or recognition process. Keep in mind that resizing an image from its regular size to tiny one (less than 1000 pixel) will not change any details and it will be sill recognizable to human eye [19].

Furthermore, tiny images have been efficiently used for classification images from 80 million dataset images. This research show also that combining DBNs with tiny images led to assign each image by a small binary vector due to the component of the feature existed in considered image [20]. The binary vector act like a barcode; each barcode represented each image and feature. Therefore, instead of using large dataset with regular size images, we use the same dataset with resized images. By this technique, we can achieve better loading, classification and feature extraction especially with DBN approaches. Figure 3.4 shows a resizing image from its normal size to 32×32 pixel, we can see that the face in the picture with its details still recognizable [16][18][19].

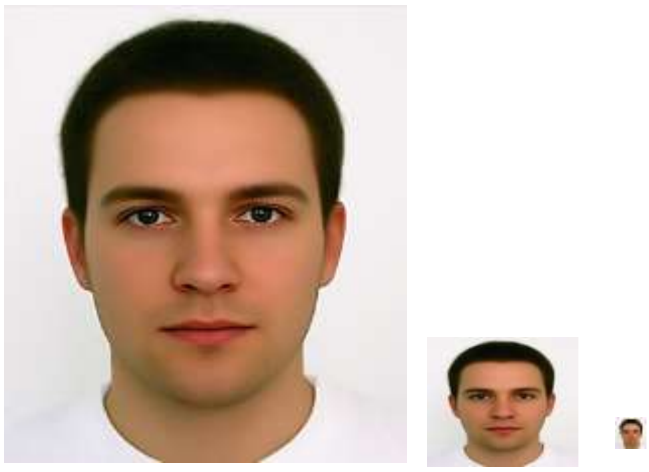
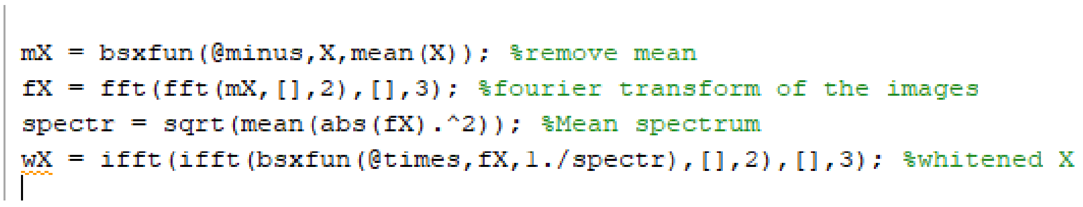


Figure 04—: Resizing image from the normal size to 100×100 pixel then to 42×42 pixel. Notice that the tiny image still recognizable

3.3.3. Data Whitening and Normalization

Natural images usually have strong correlation and relationship that represent the 1st and 2nd order statistics. Data whitening is simply remove these statistics. Therefore, after this whitening it is possible to highlight the higher order statistics which represents the edges, corners, lines etc. Data whitening is important for all deep learning procedures (unsupervised learning methods), because dimension reduction helps to significantly speed up the unsupervised feature extraction and learning algorithms [54][57].

Image whitening is image pre-processing step that removes correlations of order two. These researches showed also that data whitening is helpful to pre-processing strategy, especially in independent component analysis [19]. Whitening algorithm in figure 3.5 is done by finding the inverse square root of the covariance matrix for set of images (considering images as matrices). We can whiten a set of images by multiplying the data by the inverse of the square root of the mean. We represent each image by a matrix X of order height (number of rows) × width (number of columns).



```
mX = bsxfun(@minus,X,mean(X)); %remove mean
fX = fft(fft(mX,[],2),[],3); %fourier transform of the images
spectr = sqrt(mean(abs(fX).^2)); %Mean spectrum
wX = ifft(ifft(bsxfun(@times,fX,1./spectr),[],2),[],3); %whitened X
```

Figure 05--: Image whitening algorithm

Until now, we accomplished two goals with whitening: make the feature less correlated to each other, and giving all of the features the same variance.

Now we need to find the eigenvector, and project on it all dataset in order to remove the correlation between the components that normalize it. This process assign 1 for all the variance components, and 0 for all mean components. The following equation explain this process

$$\text{Normalized-data} = (\text{original-data} - \text{mean}(X_i)) \backslash \text{std}(X_i). \quad (3.1)$$

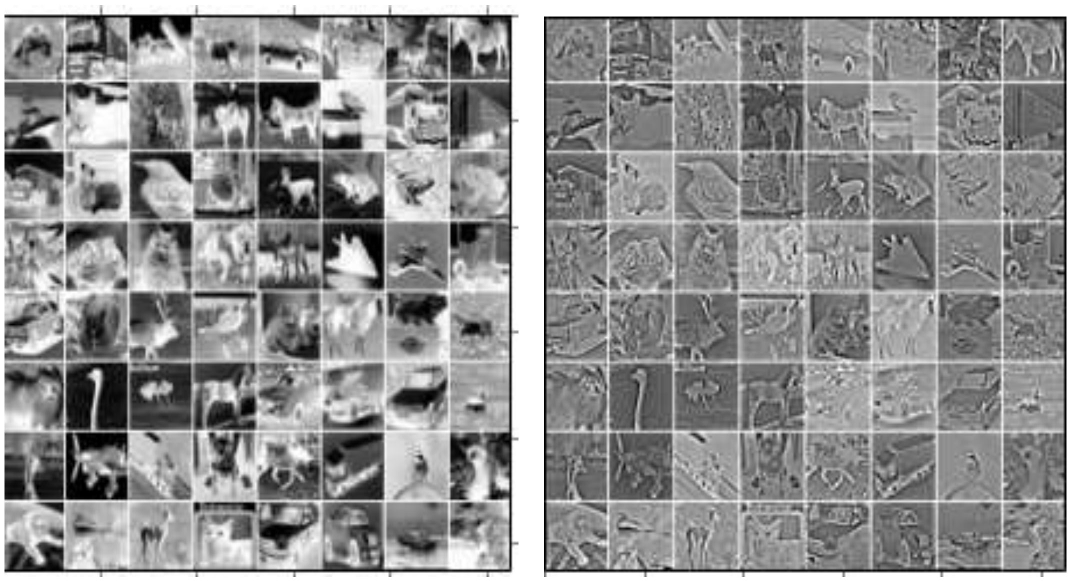


Figure 06—: On the left is normalized 8×8-patch sample, on the right whitened 8×8-sample patch

This local normalization considered as another way to pre-process data. Figure 3.6 is a result of a study done to try to figure out the effect of both normalization and whitening pre-process techniques on a dataset of images. The authors argue in this study both whitening and normalization is affective. The use of any of them is depends on the quality of the images on the dataset and the largeness of it [16] [51].

In our proposed model we find that it is better to use normalization on our dataset images because normalization is not only aim to remove noise but also to bring the image into a satisfaction range of intensity values of normal distribution. We built a dataset by using a high accuracy camera in order to get a highly intensity and resolution images. We found out that whitening the dataset images remove important features of the hand shape of the letters. This problem is due to removing the second higher statistics of the images that eliminates important features from images. While normalization do not remove statistical orders and change only the brightness of the images.

3.4. Unsupervised Feature Space Construction.

As mentioned before RBM machines are unsupervised machine learning, so it learns how to construct data by itself [51] [54] [56]. It makes several backward and forward passes between both hidden and visible layers. Through the construction phase, the visible layer is the activation layer and the input in the forward pass. Each node in the visible layer is multiplied by the same weight and then the sum of those products is added to the hidden layer bias at each node. In the reconstruction phase, the hidden layer is the activation layer and the input in the backward pass. Each node in the hidden layer is multiplied by the same weight, and then the some of those products is added to the visible layer bias at each node [16] [18].

Hinton proposed different procedure. He recommended training the RBM using constructive divergence learning procedure. First of all a set of weights is taken randomly to link the visible and hidden layers which provides the configuration of the first hidden layer unit probabilities, then the another visible layer is reconstructed from the hidden units arrangement. The constructive divergence is then computed to update weights and biases. Finally the results is updated via random set of images called mini-patches. The whole procedure is repeated until construction procedure converges to reconstruction [16] [54] [57]. Suppose that α is a sigmoid function referring that RBM is demonstrated under Gaussian distribution $\alpha = \frac{1}{1+e^{-x}}$. v_i is the visible layer, h_j is the hidden layer and w_{ij} is the waited matrix that is assigned with Gaussian distribution because we have seen that the natural images have zero mean and a unit standard deviation (due to data normalization), so the equations that led to constructive divergence CD are: Visible layer = $\alpha(\sum h_j \times w_{ij})$

A simple pseudo code will try to figure out how to calculate the visible layer.

```
Input: $v$ = visible layer, $h$ = hidden layer, $w$ = weight matrix

Output= calculating visible vector

For $i = 0$ →visible layer number

For $j$ = 0 →hidden layer number

 $v_i += h_j \times w_{ij}$

 $v_i$ = sigmoid ($v_i$)

Return $v$
```

Hidden layer= $\alpha(\sum v_i \times w_{ij})$

A simple pseudo code will try to figure out how to calculate the visible layer.

```
Input: $h$ = hidden layer, $v$ = visible layer, $w$ = weight matrix

Output= calculating hidden vector

For $j = 0$ → hidden layer number
For $i = 0$ →visible layer number
$h_j += v_{ij}$
$h_j =$ sigmoid $(h_j)$
Return $h$
```

Finally, we calculate the constructive divergence and weighted matrix as follow:

$$CD(i,j) = (< v(i) \times h(j) >_{original} - < v(i) \times h(j) >)_{reconstructive} \qquad (3.2)$$

$$w(ij) = w(ij) + learning\_rate \times (CD_{pos} - CD_{neg}) \text{ [16][18]} \qquad (3.3)$$

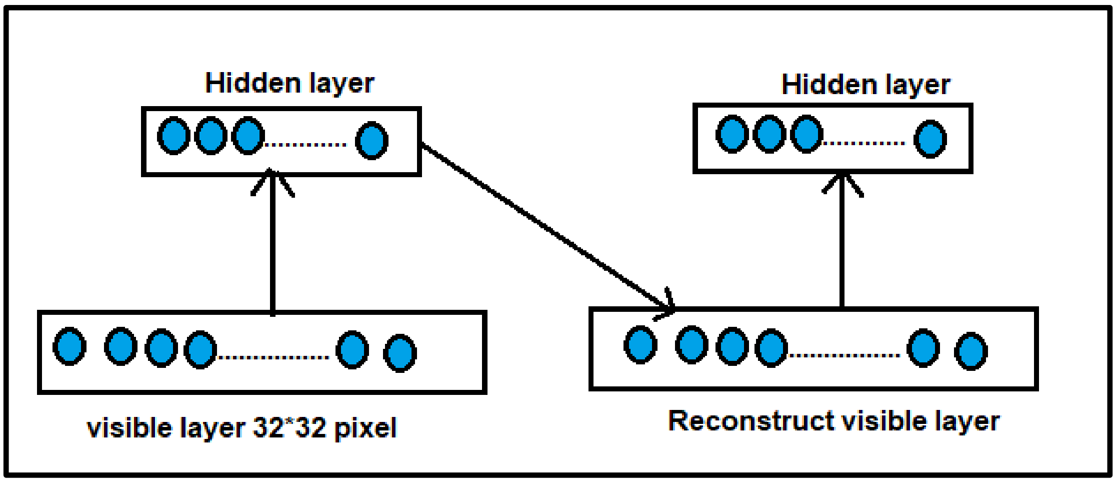


Figure 07—: shows the training of RBM machine using constructive divergence learning

During our research, we found that RBMs always developed using binary visible and hidden layers; many other types of units may be used. Such as softmax units, multinomial units, Bernoulli units, Gaussian and linear units. The raised question is which one to use? The answer is the use of any of these types refers to the type of the problem we need to solve.

For example, Bernoulli units suits the cases of hand written digits, but in our case, we use matrices of pixels in natural images. Hinton suggested replacing the binary visible units with Gaussian units for the input layer [16][18][57].

1) The learning rate η

2) The size and the set of mini-patches γ

3) The initial values of weights and biases.

4) Momentum μ

5) Weight decay λ

6) Penalty.

7) The number of hidden and visible units.

All these parameters are significantly tested and concerned during our research work to find out the optimal values for each parameter to achieve successful RBM training [16] [57] [59].

The learning rate (η):

This rate is the most important parameters in training RBM. The newly information is updated from the oldest information. Setting accurate learning rate value will lead to good recognition and feature extraction. Higher or lower values may week the feature extraction result and slow down learning. Researchers tested several values to determine the accurate learning rate that gives high quality features. Using higher values cause overload on the weights of RBM layer and thus RBM may learn nothing and the error rate will increase [16][18][19].

If we use lower learning rate values, the RBM learning will take more image epochs to derive the features and the result is slowdown in process without good features. Therefore, the value of the learning rate must suit both convergence and number of epochs. During our research we practically start using large values of η, for example when we used η=0.05 the reconstruction error was increased. Thus the weights and features have completely exploded with the first few epochs and the RBM did not learn anything. We found out that using large value of η cause imbalance of the weights and biases. We practically found that the best values of learning rate is ranging between 1-5 × 10 $^{-3.}$ The question is how to decide which value to use? The answer is by experiment only; for example RBM learning with η = 0.02 was better than learning RBM with η = 0.002, the first case converted after 300 epochs while the second case converted after 1000 epochs [16].

The best way for choosing the best learning rate is to compare between weights and weight updates, the weight updates should be 10 $^{-3}$ times the weights. That is correct since the updates are being small in RBM training until feature deriving is completed. In our proposed model we decided that the learning rate that suits our dataset size and achieve good feature extraction results is = 2×10^{-3}. Higher rate may cause wrong learning while lower rate let us get time consuming for learning[54][57].

The existence of η updated the weighted matrices as follows:

$$w(ij) = w(ij) + \eta \times (CD_{pos} - CD_{neg}) \qquad (3.4)$$

The size and the set of mini-patches (γ)

Due to the largeness of dataset, it is more efficient to split the training dataset into mini-patches, where each one contains either 10 or any other value between 100 and 200 images. Making mini-patch size too large is a mistake, although increasing the mini-patch size conduct more reliable gradient estimation, it defects the values of learning rate and weights updates.

The size of mini-patch must be decided carefully. The best min-patch size is equal to the number of different classes. Each mini-patch should contain one example of each class to reduce the error in the gradient of the training set. For example, it was found that if the min-patch size is 10 then the RBM would not learn good features. On the other hand using a mini-patch size of 100 makes an RBM learning machines more high quality features [16].

The existence of the size of mini-patch overriding the equation of weights (features) as follows:

$$w(ij) = w(ij) + \eta \times (CD_{pos} - CD_{neg}) \setminus \gamma \qquad (3.5)$$

As we see the size of mini-patch influence the extraction of features, it speeds up the training process of RBM and make balance between the constructive divergence and the learning rate. In our proposed model, after doing many experiments we noticed that when we use large mini batches. For example γ =200 the extracted features were not different from those obtained using γ=100. We noticed that the total number of features is higher when γ=100, while when we use γ=10 the RBM did not learn any features. Finally, we divided the whole dataset of 10,000 images into 50 mini-batches; the size of each mini-batch is 200 images [16].

The initial values of weights and biases.

It is important to decide the initial values of weights and biases to start RBM training. Typical way is to start the training with small random values from Gaussian distribution with zero mean and 0.01 standard deviation. Another way is to use uniform distribution small random values. Since our RBM is set to be Gaussian Bernoulli model as mentioned earlier, it is best to set the initial values and biases using Gaussian distribution. Hinton recommended that all initial weights and biases should be either zero or – 4 to encourage the sparsity [54] [59].

Momentum (μ)

A simple way to speed up the learning process is to set up a value of momentum. Momentum considered as important standard parameter for RBM, researches recommended that momentum values should range between 0 and 1. Good recipe for picking momentum value is to start with momentum of 0.5 for the first 3 - 4 epochs and rise the momentum values to 0.9 for the rest of epochs during training. This may increase the construction error and cause more instability during training. To avoid this we have to decrement the learning rates by factors of two until RBM stability. The weighted equation is updated as follow:

$$w(ij) = \mu w(ij) + \eta \times (CD_{pos} - CD_{neg}) \setminus \gamma \qquad (3.6)$$

We can notice that momentum with learning rate controls the weights update and make the learning process more stable[16][18].

Weight Decay (λ)

Weight decay λ is another important input factor that added to the normal gradient to penalize large weights. Weight Decay λ updated the equation (3.6) as follows:

$$w_{ij} \leftarrow \mu \times w_{ij} + \eta \times \left\lfloor ((\left\langle v_i^0 h_j^0 \right\rangle - \left\langle v_i^n h_j^n \right\rangle) \setminus \gamma) - \lambda \times w_{ij} \right\rfloor \qquad (3.7)$$

The importance of using accurate values of λ is summarized as:

1- Reducing over fitting to the training data.
2- Shrinking useless weights.
3- Improving the quality of the extracted features.
4- Improving the mixing rate of alternative Gibbs Marcov chain to mix more rapidly [16] [57].

Referring to equation (3.7) we need to set up values for λ that gives balance to both equation sides. It was found that λ values should be the sum of the squared weights multiplied by the coefficient or weight cost. Researchers investigated the acceptance values of λ, which range between 0.01 and 0.00001. The use of large values gives shallow features. On the other hand, the use of low values gives interesting features [16] [57].

In our proposed model, we figure out the best λ value to be 0.0002 after testing several values. It appears that 0.0002 gives better interesting features to our dataset and model.

Penalty term

Hidden units that are only rarely active are typically easier to interpret. The hidden units takes about half the time of the active units. In addition, discriminative performance is sometimes improved by using features that are only rarely active [58].

Sparse activities of the binary hidden units can be achieved by specifying a sparsity target p, which represents the desired probability that the unit is active. Additional penalty term q is then used to encourage the actual probability of being active. The update is the penalty proportional to $q - p$. Computing the sparsity of the hidden unit is done by averaging its activation across training as follows:

$$q_{new} = \lambda q_{old} + (1-\lambda) q_{current} \qquad (3.8)$$

Where q current is the mean activation probability of the hidden unit on the current mini-batch, λ is the decay rate that varies between 0.9 and 0.99 [16] [19].

For logistic units this has a simple derivative of $q - p$ with respect to the total input to a unit. We used scaled parameter called cost, which adjust both biases and weights of each hidden units. The cost equation is set as follow

$$Cost = -p \log q - (1-p) \log(1-q) \qquad (3.9)$$

A recipe for sparsity is to set the target between 0.01 and 0.1^9, and set λ to be between 0.9 and 0.99. When we add the sparsity penalty to the learning rule, the new weight update formula becomes

$$w_{ij} \leftarrow \mu \times w_{ij} + \eta \times \left\lfloor ((\langle v_i^0 h_j^0 \rangle - \langle v_i^n h_j^n \rangle) \setminus \gamma) - \cos t \times (p_t - q) \right\rfloor \qquad (3.10)$$

The number of hidden and visible units.

The last important factor of training RBM is to choose the number of hidden layers. There are three different situations for their size:

1- Less than the size of input layer.
2- More than the size of input layer.
3- The same size of input layer.

The size of the hidden layer can be identified by three factors:

1) The size of visible layer.

2) The target to achieve.

3) The redundancy of the batches in the training set [54].

According to our dataset, the size of the first hidden layer could be either equivalent or reduced to half of the visible layer size. For example if we use equivalent size of 256 for both visible and hidden units, many features will be blank and flat and the RBM result will not be satisfied. Therefore, reducing the size to 16 for both visible and hidden units, the RBM extracts very good features with fully matrix used.

In our proposed model, we decide to set both visible and hidden units to the same size of 32, because we assure that this choice have been examined to give the best features extraction during the training of RBM [16].

DBN is high sophisticated technique for machine learning due to the existence of layer by layer operations from the top down learning. Variables on a layer determines the variables on the next layer. After training, the values of the variables in a layer become as a vector that passes the weights in the reverse direction. DBN is powerful due to extract sparse features from large dataset.

So far we presented RBM machine, that is described in the algorithm that shown below, which is the feature extracted model that we are going to use. We determined all variables and factors that must be chosen carefully to give high RBM training and to extract interesting features [16] [18] [19] [57]. Also, we selected random batch v_0 from the database, and initialized a weight matrix w_{ij} with Gaussian distribution. We set the weights increment $w_{inc_{ij}}$ to zero. b_i and c_j represents the visible and the hidden biases. Epochs stands for the number of epochs we need to ensure convergence. Finally, each of the input factors: learning rate η, size of mini-batch γ, momentum μ, the weight decay λ and the unit variance, assigned a value. The algorithm is as shown:

```
Input: $v_i, w_{ij}, w_{inc_{ij}}, b_i, c_j, \eta, \lambda, \mu, \gamma, \sigma$
Output: a set of features stored in the weighted matrix
for $e = 1 \rightarrow$ epochs do
    for $j = 1 \rightarrow$ number of hidden layer do
        compute the binary hidden units $h_0 = \alpha(c_j + \sum v_{i0} \times w_{ij})$
    end
    for $i = 1 \rightarrow$ number of visible layer do
        compute the visible units $v_1 = \alpha(b_i + \sum ph_{0j} \times w_{ij})$
    end
    for j=1 $\rightarrow$ number of hidden layer do
        compute the binary hidden units $h_1 = \alpha(c_j + \sum_i w_{ij} pv_{i1})$
    end
    for i= 1 $\rightarrow$ number of visible layer do
        for $j = 1 \rightarrow$ number of hidden layer do
            $w_{inc_{ij}} \leftarrow \mu \times w_{inc_{ij}} + \eta(((v_0 \times h_0) - (pv_1 \times h_1)) \setminus \gamma) - \lambda \times w_{ij}$
            $w_{ij} \leftarrow w_{ij} + w_{inc_{ij}}$
        end
        $b_i \leftarrow b_i + \eta(v_0 - pv_1)$
    end
    for $j = 1 \rightarrow$ number of hidden layer do
        $c_j \leftarrow c_j + \eta * (h_0 - h_1)$ $c_j \leftarrow c_j + \eta(h_0 - h_1)$
    end
end
```

Restricted Boltzman Machine Training with Constructive Divergence Algorithm[54]

We can evaluate the performance of RBM only based on the accuracy achieved during the classification process, and the time consumed until divergence. The RBM efficiency depends on factors such as learning rate η, size of mini-batch γ, momentum μ, the weight decay λ and the size of the dataset [63][64].
Our model is designed to run only once to extract the sparse features, then the model procedure labelled and classified them into 28 different classes. The model execution took two hours to diverge and extract interesting features using I5 laptop with 8 GB ram and 2.6 GH$_z$ 4 CPU.

3.5. Summary

In this chapter we presented the proposed model. We give detailed explanation about each stage of it starting from collecting data base to image pre-processing and its importance, highlighting the importance of using tiny images and its effectiveness in RBM training. Then we give details about the training RBM and the factors that play role in extracting interesting features.

The next chapter introduce the results of each of the stages that mentioned above that shows the powerfulness of using DBN over RBM with constructive divergence.

4. Chapter 4

Results

4.1. Introduction

In the previous chapter, we presented our proposed model for ArSLR. We gave a detailed description of the Arabic alphabet database that have been collected and the condition of building such database to meet our requirements. Then we showed how to pre-process these images in the dataset to convert it from high to low-level data input for RBMs training .

In this chapter, we will show the results of each of the previous stages. In addition, we will classify the features that are obtained from RBM training using SVM, give percentages for hit, and miss rates for each letter in the Arabic Alphabet. We will discuss our results and comparing them to others in related work, focusing on the similarities and differences, and explaining the unsatisfied results.

4.2. Data Collecting Results

We created two kinds dataset according to our proposed model:

Dataset A is a training dataset of ArASL that consists of more than 10,000 images. These images were taken for a sign language specialist. Our goal was to collect images for each alphabet letter that can show the exact hand shape of the sign letter. We were considering the hand palm itself because each alphabet letter sign of the sign language must be done by one static posture as we said before. The background of the photo shoot was white because it helps to highlight the hand shape and gives concentration to only the hand sign itself. The camera focus (the distance between the camera and the hand of the specialist) was not static; there were a contrast in lightness and orientation to get different positions for each letter sign in the Alphabet.

We collected between 220 – 300 images for each of the 28 Arabic language letters. We successfully built multivariate dataset with different attributes. We considered that each letter in the Arabic Alphabet is a class, so our dataset consists of 28 different classes with different attributes.

Our dataset is very large and contains different classes with possible attributes. We successfully built a dataset of coloured hand gestures, with high customization on the shape of the hand shape and finger combination that represents the letter sign. There is no available open source dataset because there are a few researches about Arabic Alphabet signs. Therefore, we can claim that our dataset is the best one for Arabic Alphabet sign until now. Not only because of its hugeness but also because of the wide variety in the images and signs that represents Arabic Alphabet letters.

Dataset B is a testing dataset, which consists of 150 images for each class or letter with the same attributes. This dataset will give the final unbiased evaluation and validation of the proposed model. The testing dataset also contains 28 different classes with deferent attributes; each class is for testing the similar class on dataset A. Our testing dataset is designed to cover all the images and Arabic sign letters on dataset A in order to give fairness, integrity and transparency for testing. Figure 4.1 shows sample images of 28 ArSL in the dataset A and B. While figure 4.2 shows sample of different attribution for the same letter.

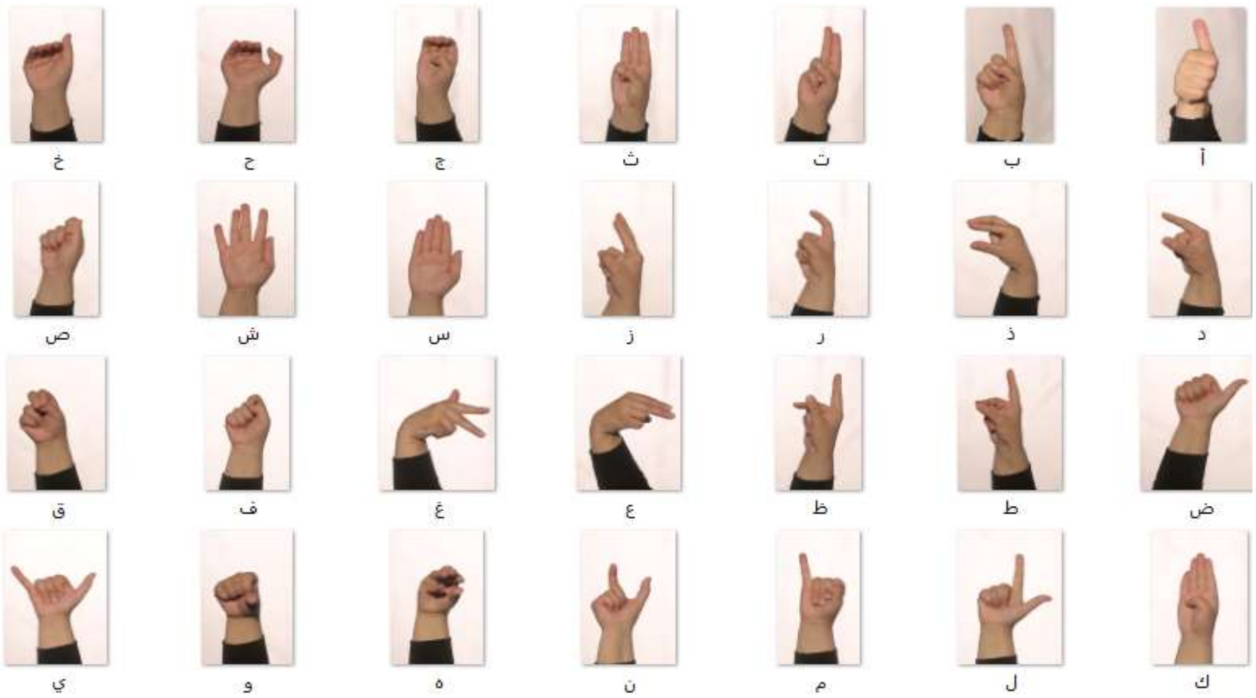
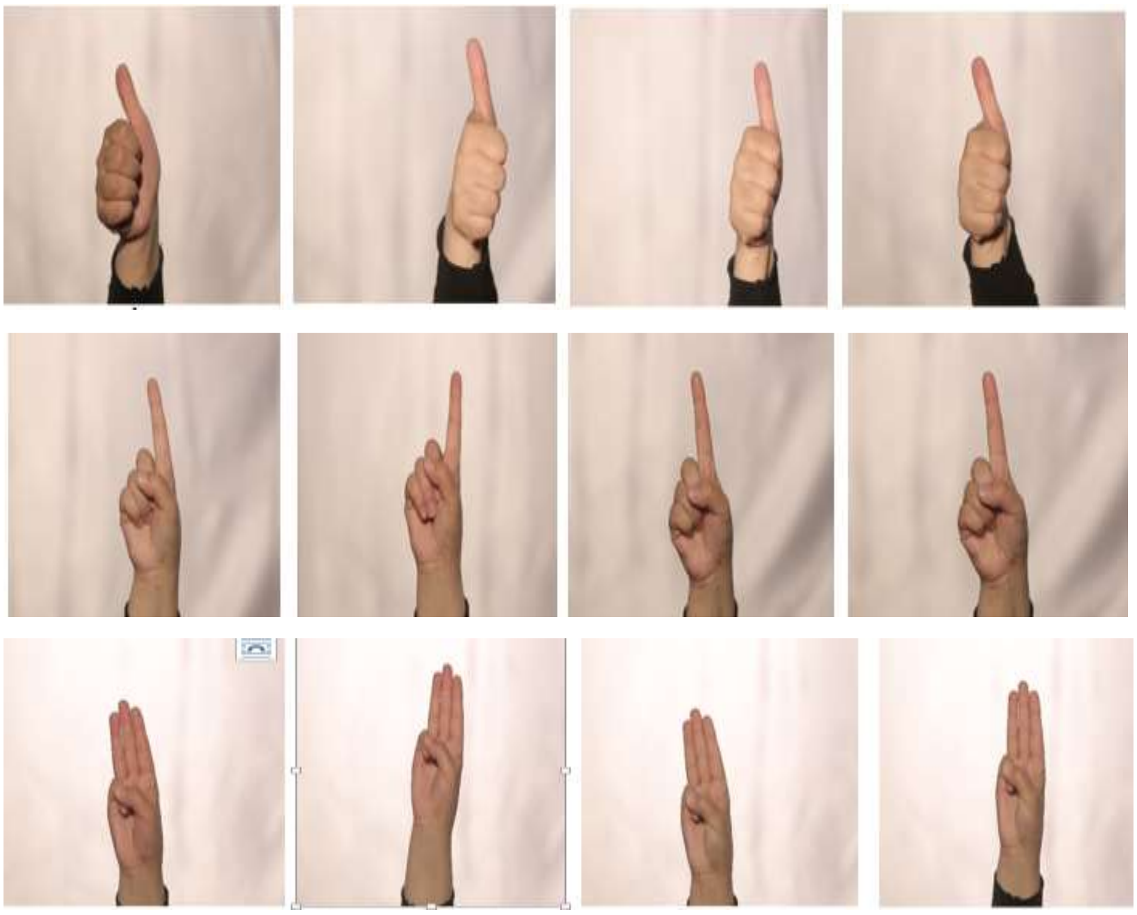


Figure 01—: The Arabic Alphabet signs Dataset done by the specialist Ms. Engy Alaabed



4.3. Image pre-processing results

Figure 02—: Samples from the training dataset (أ،ب،ث) taken under different attribution

The next step of the proposed model is to prepare the dataset images for extracting features. Image pre-processing stage handle cropping, grey scaling, resizing and normalization. This image pre-processing transformed the images of the dataset from high level to low-level data by eliminating the non-necessary elements of the images such as RGB channels, and the background of the images that do not have interesting information for training. The results came as the following.

I. **Image Cropping**: Since the images is taken with high-qualified camera there was an empty space in the images around the hand gesture, we need to focus on the hand shape itself regardless the empty space, which can considered as the notes of the images. As shown in figure 4.3 image cropping helps to get rid of the unwanted sides and space around the hand gesture. Therefore, this process contribution is to make the images of the dataset more fitting for training stage.

II. **Image Grey Scale**: We performed image grey scaling on all images in the dataset. This process as shown in figure 4.4 gives the minimal information we need for each pixel, ensure that the images still recognizable and the hand gesture still be seen.

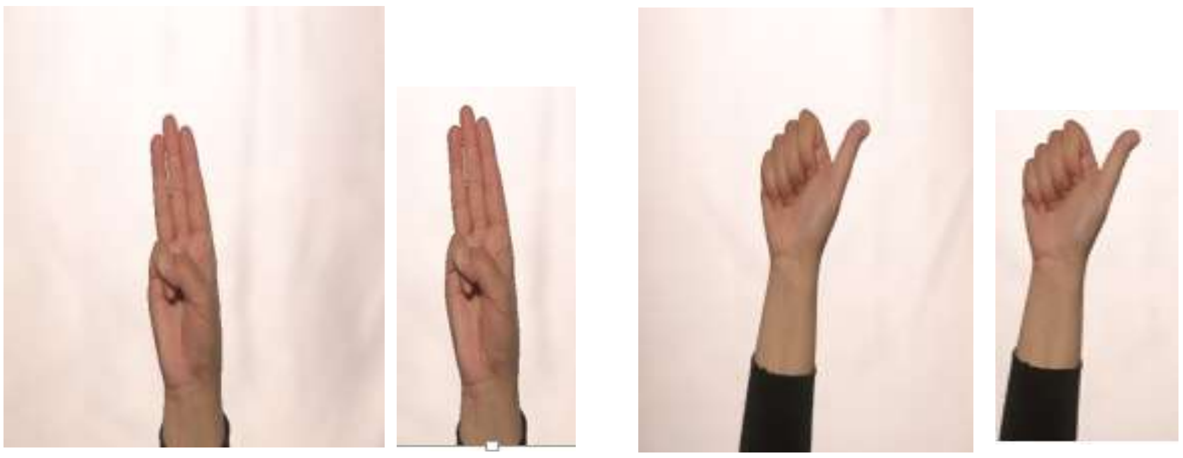
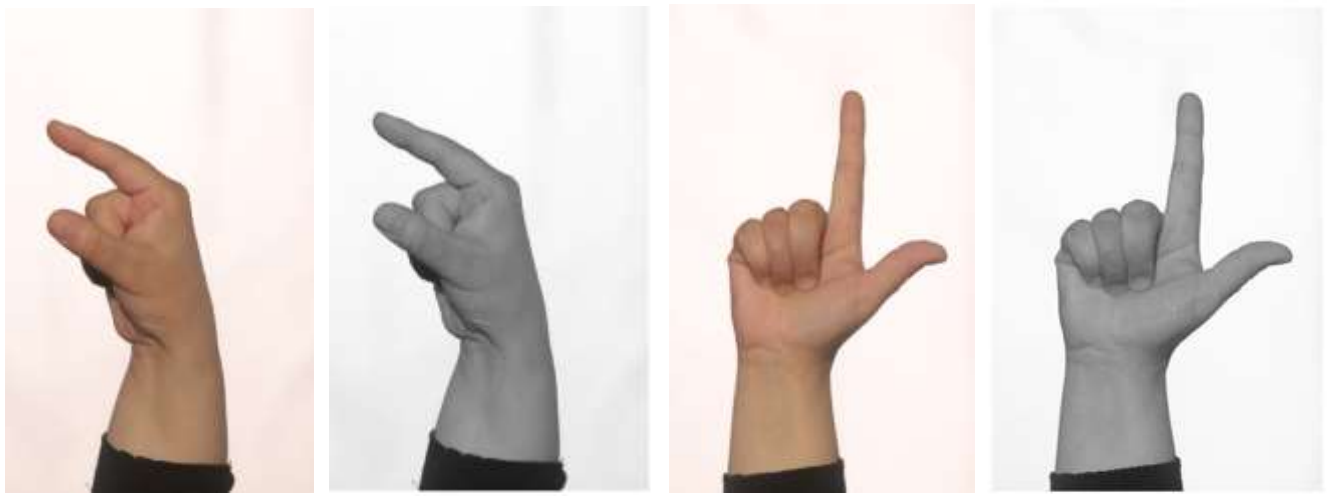


Figure 03—: Samples of image cropping to eliminate the un-necessary elements in the image and centered the hand gesture.



Grey scaled images is defers from RGB colored images that all the components have

Figure 04—: Samples of image grey scaled

the same intensity, so we can reduce the storage space into 8-bit integer. Doing this we get much simpler and transformed from high-level data to raw data that fits RBMs training stage.

III. **Image Resizing**: It is done by scaling the size of the realistic images in order to make them appropriate for DBNs. Using tiny images combined with DBN approach led to code each image by a small binary vector that can be optimally used to define the features.

We resized the images in the dataset into 32×32. It is noticeable in figure 4.5 that the image still recognizable after resizing. Resizing has a great impact on the system because of the powerfulness of speeding up the time of loading, uploading and training. In addition, using tiny images makes the multiplication of the connexion weights act positively on the convergence of the CD algorithm.

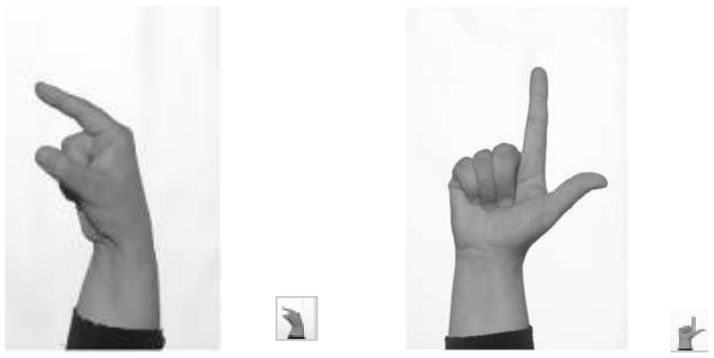


Figure 05—: Samples of image resizing on the dataset

IV. Image Normalization: We did this image pre-processing in order to bring the images in our database to the lowest level. We managed to delete everything but the edges and the corners of an image because our focusing is on the hand shape during our research.

The results of the normalization stage in Fig 4.6 came very promising. It gives semantic for the very important parts of the images in the database: edges, corners and lines give the required shape for the hand gesture. We convert the images from highly structured images contains significant statistical redundancies with strong correlation to raw images that suits RBM training. This assures avoiding distraction data from the images and gives detailed information about the structures of the views on the image. This process enhanced RBM to extract features that are more important. The shape of the hand is clearer in most of alphabets figures, and clearly, we can distinguish the letters within only the shape and the orientation. Notice that Fig. 4.7 shows the whitening result. Whitening remove some of the edges and corners from the shape of the hand in the images; this due to removing both first and second higher order statistics. Using whitening could not enhance our model because of the absence of important statistics from the images.

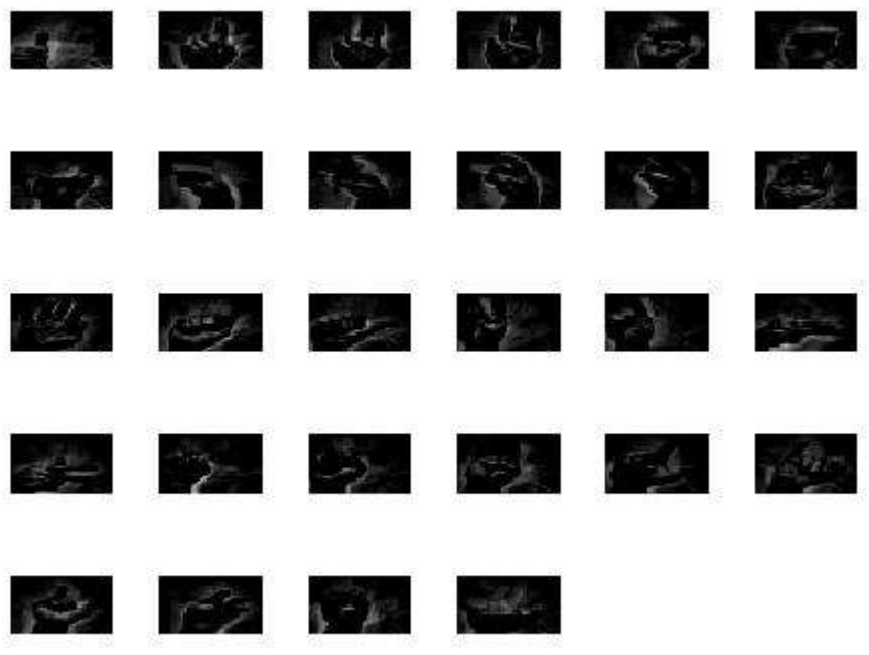


Figure 06— Applying data whitening on the Arabic Alphabet database

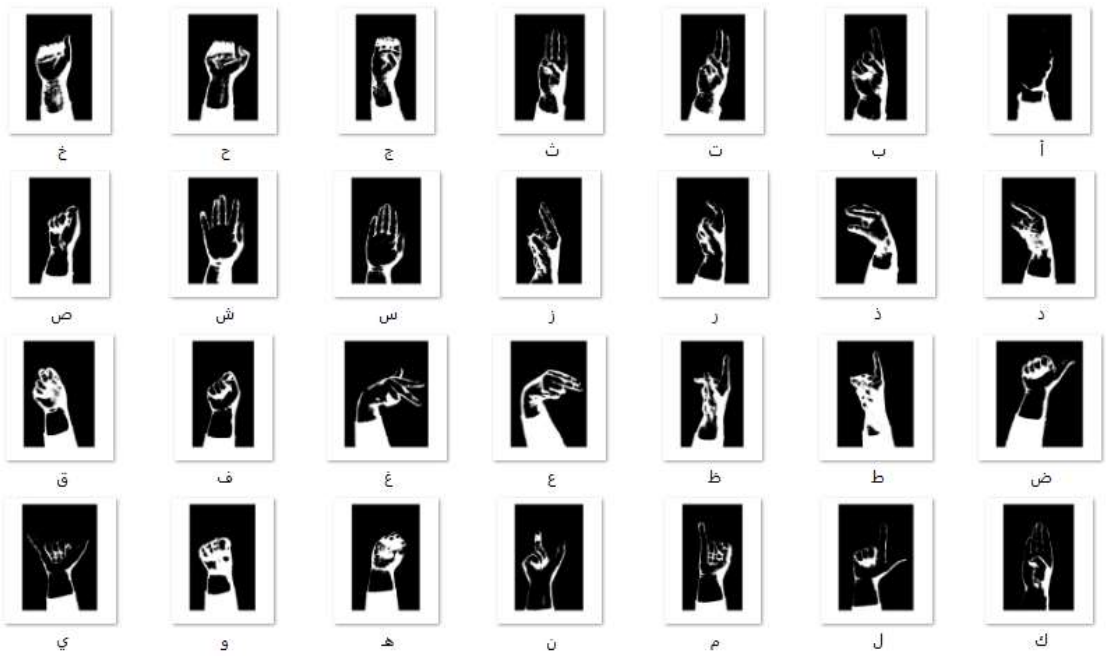


Figure 0—6 Applying data Normalization on the Arabic Alphabet database

4.4 Features extraction based on RBM training on tiny images

We present the feature extraction results based on DBN and the training of RBM. Fig. 4.7 shows the features that have been extracted from the first layer of RBM training on 10000 tiny (32×32) normalized images. Notice that those features covered the 28 letters of the Arabic Alphabet signs. The experiment is applied within the parameters that have been assigned to significant values for each. The first hidden layer contains 300 epochs with 200 sized mini-patch and learning rate of 2×10^{-3}.

We can see that the size of both visible and hidden layers was suitable because there are a few blank features on the training result. There are affective redundancy covered every sign letter assure that RBM learned interesting features.

Based on the experiment we noted that the values of the significant factors of RBM affect to express the features too much, because the number and the types of the features was satisfying. The selection of the learning rate value allow the network to converge faster toward meaningful features. The value of the weight decay was significant to penalize large values that could happen during learning. Moreover, the penalty term contribute to encourage the sparsity on the hidden activation units. These observations put on count the powerfulness and stabilization of our proposed model.

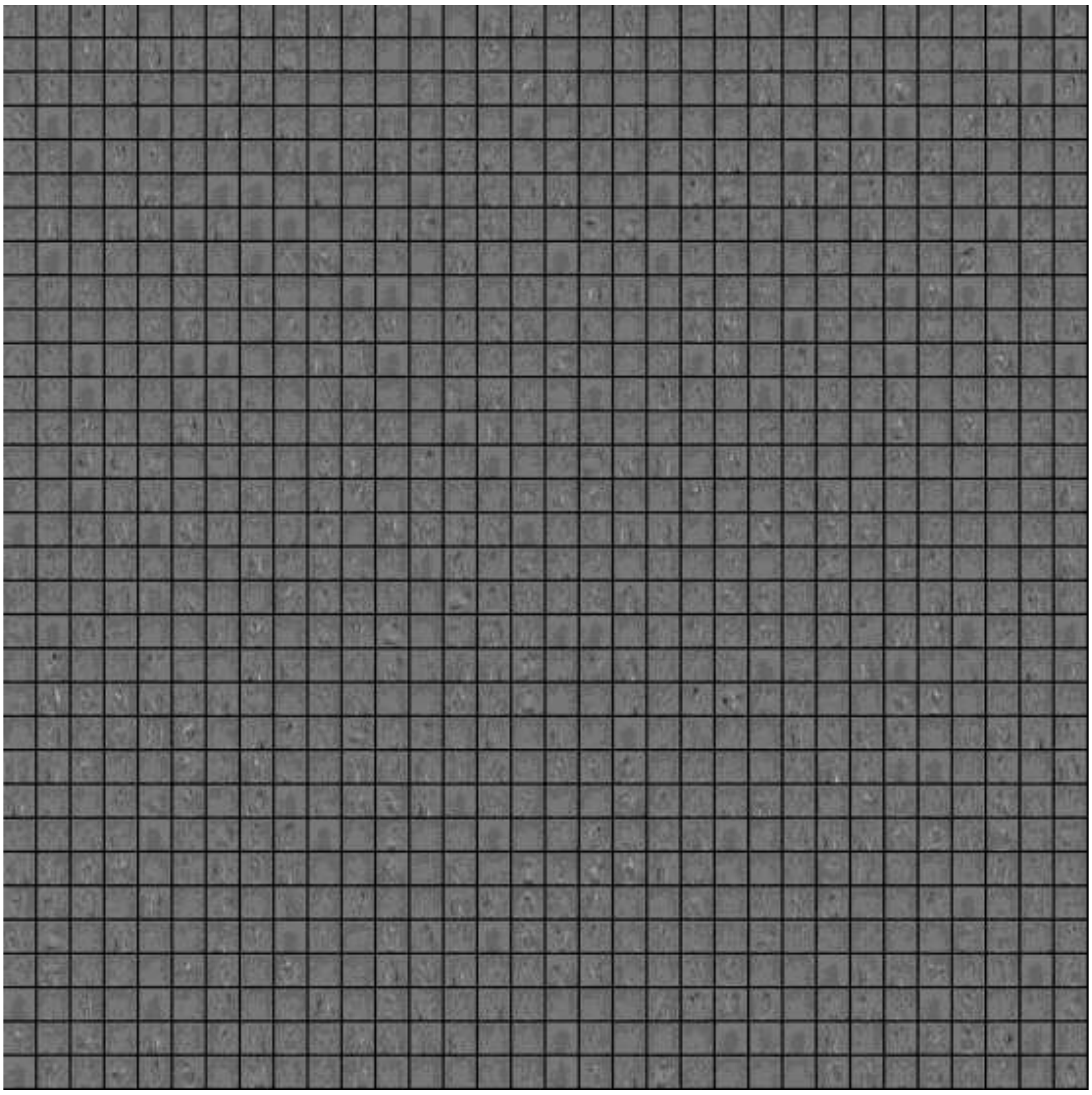


Figure 0—7: the 1024 features extracted by RBM training on the dataset.

4.5 Classification results

The final stage of our proposed model is classification; we classified the features using SVM as a non-linear separable classifier. Classification means assigned images to 28 different classes of features of the RBM learning.

After appropriate coding for the features that are extracted from RBM training phase, we labelled the letters from 1 to 28. Labeling means to assign a name for each class that comes up with the classification results. For example, class 1 will be the letter أ, class 2 will be the letter ب ….etc. Classification done to try to get better hit-rate as follows

Classification results 1: We use only 50 testing images for each letter sign, the figure below show the results. We can see that the results are varying in hit rates due to the quality of the images for each alphabet sign and the complication of the hand shape, we can see that the complication of alphabet signs such as أ،د،ث for example is not complicated as other alphabet signs such ص، ض، ط،ق،ف . Hence, the letters أ،د،ل،ك،ت succeeded to score high hit rates above 80% while the very complicated letter signs hit rates varies from 25% to 70 %.

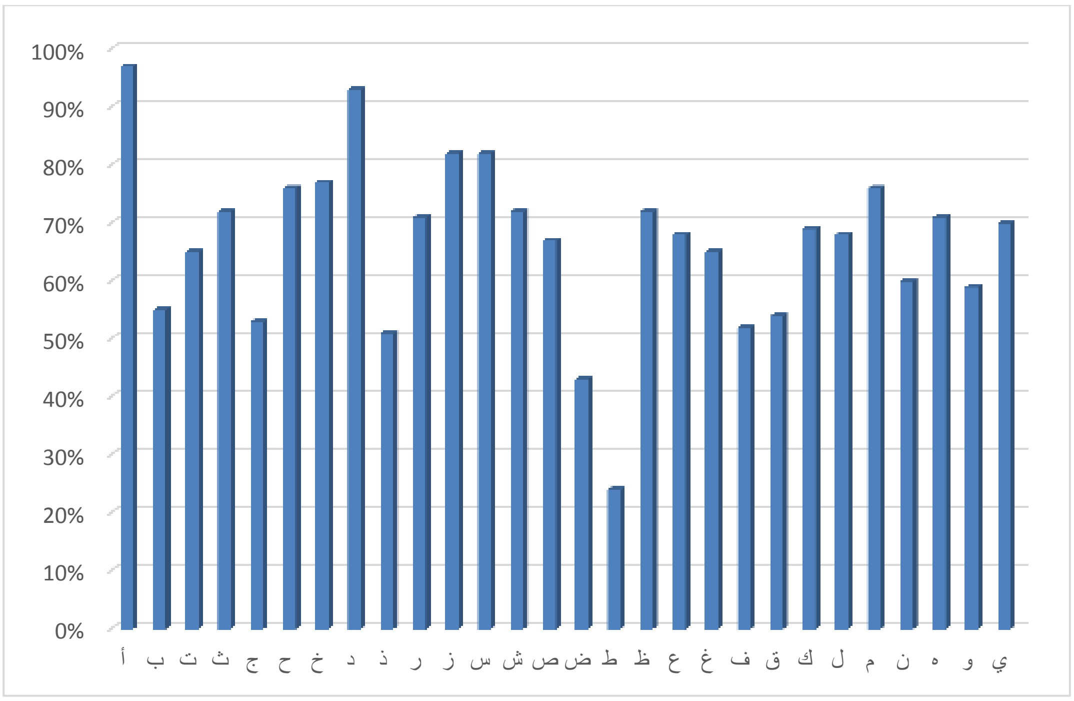


Figure 4.8: Classification result 1:- 50 images test

Arabic Alphabet Letter	Hit rate
أ	97%
ب	55%
ت	65 %
ث	72 %
ج	53 %
ح	76 %
خ	77 %
د	93 %
ذ	51 %
ر	71 %
ز	82 %
س	82 %
ش	72 %
ص	67 %
ض	43 %
ط	24 %
ظ	72 %
ع	68 %
غ	65 %
ف	52 %
ق	54 %
ك	69 %

ل	68 %
م	76 %
ن	60 %
ه	71 %
و	59 %
ي	70 %

Table4.1 Classification result 1.

<u>Classification results 2:</u> we raised the testing images to 100 testing image for each letter. The hit rates is raised significantly, because we have about 250 images for each letter in the database and the 50 images for testing means that the ratio between the images in the database and the testing images is 1:5 and it is very low percentage. The hit rate score for letters such as ب،ت،ث،ج،ذ،ط،ش،ر،ص،ض،ظ،ع،غ،ف،ق،ك،ل،ن،ه،و،ي increased by 10 % .

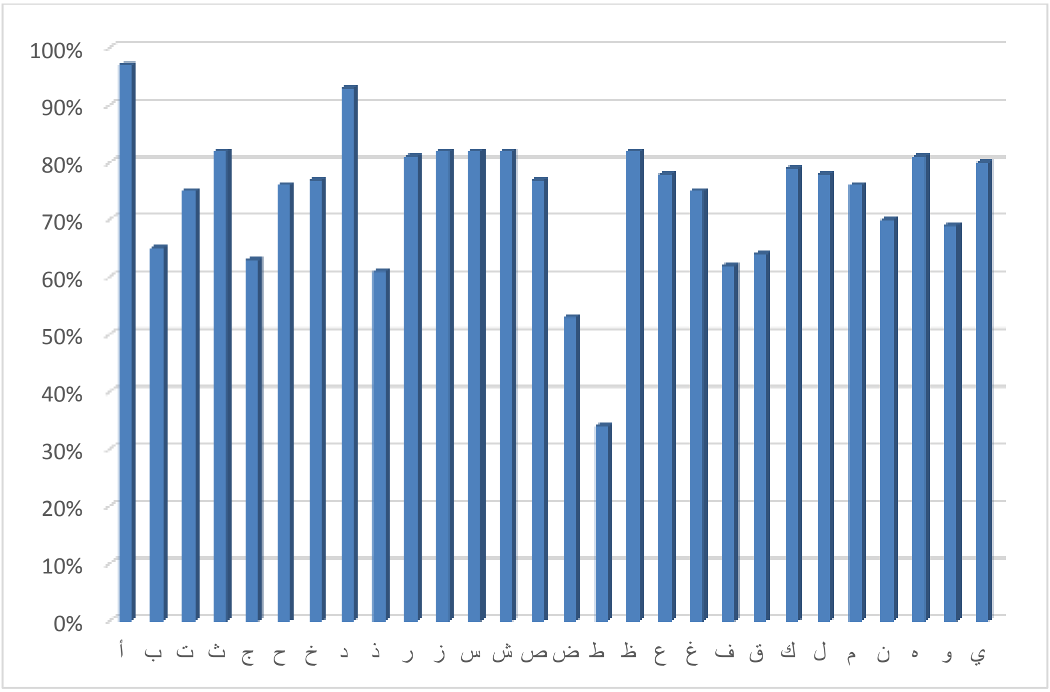


Figure 4.9: classification result 2 100 testing images

Arabic Alphabet Letter	Hit rate
أ	97 %
ب	65 %
ت	75 %
ث	82 %
ج	63 %
ح	76 %
خ	77 %
د	93 %
ذ	61 %
ر	81 %
ز	82 %

س	82 %
ش	82 %
ص	77 %
ض	53 %
ط	34 %
ظ	82 %
ع	78 %
غ	75 %
ف	62 %
ق	64 %
ك	79 %
ل	78 %
م	76 %
ن	70 %
ه	81 %
و	69 %
ي	80 %

Table4.2: Classification result 2

Classification result 3: We tested150 images for each sign letter, the hit result also arise to about 86% for all. We can see that most of the letters hit rate is above 70% . The letters ض،طstill having low hit rate, this is due to the complication of the signs (the shape of the letter) and the similarities between the both letters in hand shape give some misunderstanding to the model.

Arabic Alphabet Letter	Hit rate
أ	97%
ب	75%
ت	85 %
ث	92 %
ج	73 %
ح	76 %
خ	77 %
د	93 %
ذ	71 %
ر	91 %
ز	82 %
س	82 %
ش	92 %
ص	87 %
ض	63 %
ط	44 %
ظ	92 %
ع	88 %
غ	85 %
ف	72 %
ق	74 %
ك	89 %

ل	88 %
م	76 %
ن	80 %
ه	91 %
و	79 %
ي	90 %

Table 4.3: Classification result 3

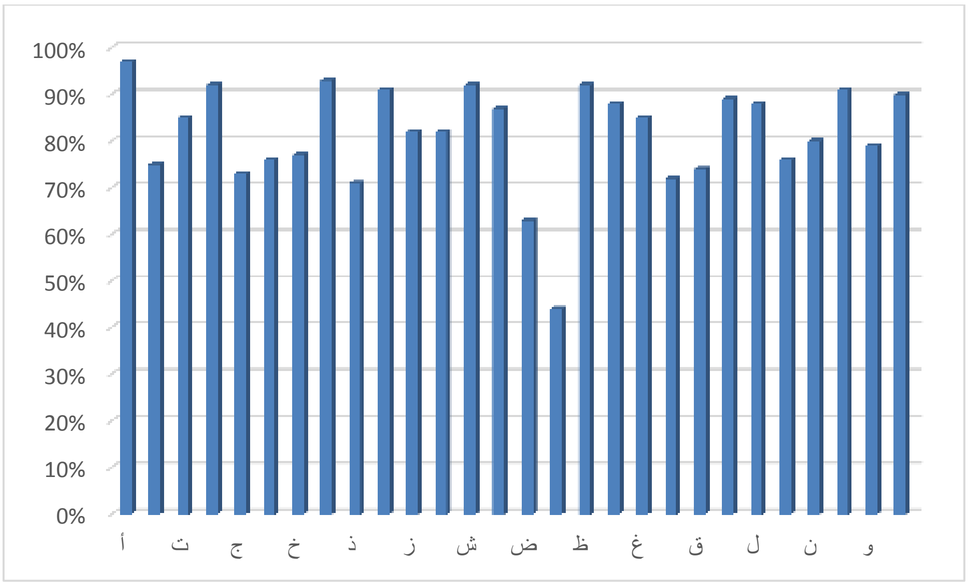


Figure 4.10: Classification result 3   150 testing image

Figure 4.12 shows the hit rate of the three classification results, which approve that how the result is getting much better when we use 150 testing images. If we take a look on the letter ط result we notice that it arose from about 30% to about 50% but still not satisfying result due to the big similarity of the shape of the letter ط and the letter ظ (very slice different on the shape of the hand). Maybe the hit rate of the letter ط will increase if we use more related image on the database for example 450 images instead of 350 images.

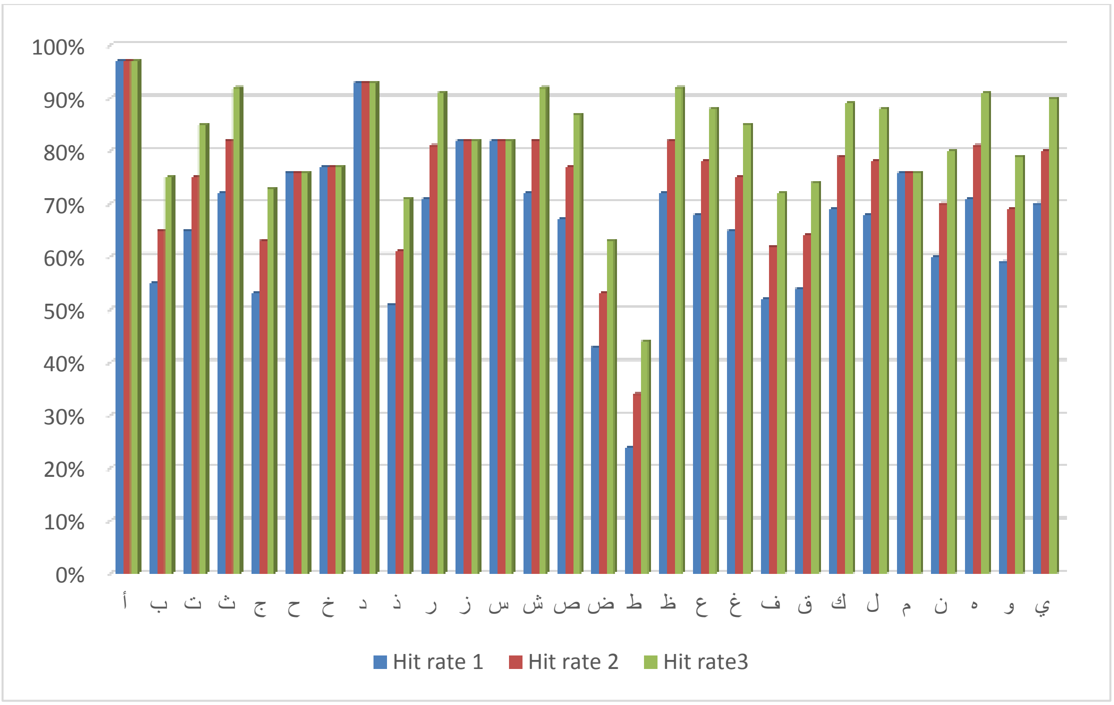


Fig: 4. Average classification rate from the three different classification results

The results of the proposed model is then compared to the results in the literature review. The hit rate of our proposed model 86% is set within the range of the hit rates in [4][6][9][13][35]. The model classification results is then tested using softmax regression to assure sparsity and linear segmentation of the features. Our model still have the advantage of having large quantity and high quality datasets for booth training and testing stages. Moreover, the use of sensors versus free hand affects the hit rate in each of the researches in this field. Table 4.4 shows the results of the proposed model compared to the results in the literature review.

	Instrument used	Number of letters	Classifier	Hit rate
R.Naoum [4]	Free Hands	28 letters	KNN	80%
Almohandes[6]	Cyber Glove	15 letters	SVM	99.6%
Elbendary [9]	Free Hands	15 letters	MLP	83.7%
Assalah [13]	Glove	30 letter	Polynomial classifiers	93.4%
Maraqa [35]	Free Hands	20 signs	Neural Networks	90%
The proposed model	Free Hands	28 letters	SVM	86%
The proposed model	Free Hands	28 letters	Softmax Regression	84%

Table 4.4 Comparison of ArSLR results systems

4.6 Summary

In this chapter, we give results of each step in our proposed model. Testing is done using SVM classifier and the results is very realistic comparing to other results in the preview. We achieved using RBM coupled with tiny images image based recognition system to ArSL with very interesting and promising results with 86% hit rate.

5.Chapter 5

Conclusion and Future Work.

5.1: **Introduction**.

This chapter concludes the thesis and give a review of the importance of **the** proposed model. Focusing on the thesis contributions, results, limitation and future work.

In this thesis, we gave a detailed model for ArSLR system using DBN with RBM. We collected two datasets for training and testing. Training dataset is then preprocessed to convert it from sophisticated dataset to low level one. Data Normalization is done to identify the higher order statistics of the images in the dataset. The algorithm in the proposed model extracted very significant and interesting features which have been labelled and classified in to 28 classes. The proposed model achieved hit rate of 86% .

5.2: Contribution

In our research, we have three main contributions: 1) Extracting features for ArSL alphabets using DBN with RBM. 2) Building a unique huge datasets for both training and testing. 3) The use of tiny images instead of bag of words.

5.3: Results:

Our proposed model is evaluated and tested using SVM. The proposed model achieve 86% recognition rate witch is a very accepted result due to the hugeness of the dataset and the rigidity of the model.

5.4: Limitation.

The limitation of our research is as follows:

1) The creation of the dataset was not simple; there were no open source datasets that can help us in our research.
2) Limited literature review in the ArSLR field.
3) Limited experience in multi-posture ArSLR.

5.5: Future Work

There are several open issues in the field of ArSLR systems. Most of them cannot extract features for Arabic signs that compose of two or more static posture. In addition, there is a problem in the capacity of the dataset in ArSLR systems.

Our future work is to build a recognition system based on DBN with RBM algorithm that can extract not only signs with static posture such as alphabets and numbers, but also extract features for more than two postures signs such as Arabic welcoming words, Arabic places and time descriptive words….etc.

The system could be used as an Arabic sign language web service that can be used in the conferences and meetings attended by deaf people. Also the system could be used as a mobile application that gives real time translation for sign language.

References

[1] M. Mohandes, M. Deriche, and J. Liu, “Image-based and sensor-based approaches to arabic sign language recognition,” *IEEE Trans. Human-Machine Syst.*, vol. 44, no. 4, pp. 551–557, 2014.

[2] P. Ketki Vijay, N. Nilakshi Suhas, C. Suparna Chandrashekhar, and D. Ketaki Dhananjay, “Recent Developments in Sign Language Recognition : A Review,” *ISSN (Print*, no. 2, pp. 2278–5140, 2012.

[3] A. M. Riad, H. K. Elmonier, and A. S. Asem, “Signsworld; Deeping into the Silence World and Hearing Its Signs (State Of The Art),” *Int. J. Comput. Sci. Inf. Technol.*, vol. 4, no. 1, pp. 189–208, 2012.

[4] R. Naoum, H. Owaied, and S. Joudeh, “Development of a new Arabic sign language recognition using k-nearest neighbor algorithm,” 2012.

[5] N. Salleh, J. Jais, L. Mazalan, … R. I.-… on M. and, and undefined 2006, “Sign language to voice recognition: hand detection techniques for vision-based approach,” *Citeseer*.

[6] M. A. Mohandes, “Recognition of Two-Handed Arabic Signs Using the CyberGlove,” *Arab. J. Sci. Eng.*, vol. 38, no. 3, pp. 669–677, Mar. 2013.

[7] “احصاءات | الجهاز المركزي للاحصاء الفلسطيني.” [Online]. Available: http://www.pcbs.gov.ps/site/lang__ar/507/default.aspx#م. [Accessed: 24-Apr-2018].

[8] M. A. Abdel-Fattah, “Arabic sign language: A perspective,” *J. Deaf Stud. Deaf Educ.*, vol. 10, no. 2, pp. 212–221, Apr. 2005.

[9] N. El-Bendary, H. M. Zawbaa, M. S. Daoud, A. E. Hassanien, and K. Nakamatsu, “ArSLAT: Arabic Sign Language Alphabets Translator,” *Int. J. Comput. Inf. Syst. Ind. Manag. Appl.*, vol. 3, pp. 2150–7988, 2011.

[10] S. M. Halawani, “Arabic Sign Language Translation System on Mobile Devices,” *Int. J. Comput. Sci. Netw. Secur.*, vol. 8, no. 1, pp. 251–256, 2008.

[11] P. Ketki Vijay, N. Nilakshi Suhas, C. Suparna Chandrashekhar, and D. Ketaki Dhananjay, “Recent Developments in Sign Language Recognition : A Review,” *ISSN (Print*, no. 2, pp. 2278–5140, 2012.

[12] H. Khaled, S. G. Sayed, E. S. M. Saad, and H. Ali, “Hand Gesture Recognition Using Modified 1$ and Background Subtraction Algorithms,” *Math. Probl. Eng.*, vol. 2015, pp. 1–8, Apr. 2015.

[13] K. Assaleh, T. Shanableh, and M. Zourob, “Low Complexity Classification System for Glove-Based Arabic Sign Language Recognition,” Springer, Berlin, Heidelberg, 2012, pp. 262–268.

[14] M. Mohandes and M. Deriche, “Arabic sign language recognition by decisions fusion using Dempster-Shafer theory of evidence,” in *2013 Computing,*

Communications and IT Applications Conference (ComComAp), 2013, pp. 90–94.

[15] M. Abull-ela, M. F. Tolba, and A. SamirElons, “Pulse-coupled neural network feature generation model for Arabic sign language recognition,” *IET Image Process.*, vol. 7, no. 9, pp. 829–836, Dec. 2013.

[16] A. Hasasneh, E. Frenoux, and P. Tarroux, “SEMANTIC PLACE RECOGNITION BASED ON DEEP BELIEF NETWORKS AND TINY IMAGES.”

[17] Y.-A. Daraghmi and A. M. Hasasneh, “Accurate Real-Time Traffic Sign Recognition Based on the Connected Component Labeling and the Color Histogram Algorithms.”

[18] R. Sarikaya, G. E. Hinton, and B. Ramabhadran, “Deep belief nets for natural language call-routing,” in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 5680–5683.

[19] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A Fast Learning Algorithm for Deep Belief Nets,” *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[20] A. Mohamed, T. N. Sainath, G. Dahl, B. Ramabhadran, G. E. Hinton, and M. A. Picheny, “Deep Belief Networks using discriminative features for phone recognition,” in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 5060–5063.

[21] T. M. (Tom M. Mitchell, *Machine Learning*, 1st ed. McGraw-Hill, 1997.

[22] A. L. Samuel, “Some Studies in Machine Learning Using the Game of Checkers,” *IBM J. Res. Dev.*, vol. 3, no. 3, pp. 210–229, Jul. 1959.

[23] D. Rumelhart and J. McClelland, “Explorations in the Microstructure of Cognition (Volume 1: Foundations).,” 1988.

[24] V. Vapnik, “Statistical learning theory. 1998,” 1998.

[25] J. R. Quinlan, “Induction of decision trees,” *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.

[26] P. Domingos and M. Pazzani, “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss,” *Mach. Learn.*, vol. 29, no. 2/3, pp. 103–130, 1997.

[27] K. Deb, “An introduction to genetic algorithms,” *Sadhana*, vol. 24, no. 4–5, pp. 293–315, Aug. 1999.

[28] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When Is ‘Nearest Neighbor’ Meaningful?,” Springer, Berlin, Heidelberg, 1999, pp. 217–235.

[29] M. Steinbach, G. Karypis, and V. Kumar, “A Comparison of Document Clustering Techniques.”

[30] T. J.- ICML and undefined 1999, “Transductive inference for text

classification using support vector machines,” *cs.columbia.edu*.

[31] K. P. Bennett and A. Demiriz, “Semi-Supervised Support Vector Machines.”

[32] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” in *Proceedings of the eleventh annual conference on Computational learning theory - COLT’ 98*, 1998, pp. 92–100.

[33] M. Mohandes, Junzhao Liu, and M. Deriche, “A survey of image-based Arabic sign language recognition,” in *2014 IEEE 11th International Multi-Conference on Systems, Signals & Devices (SSD14)*, 2014, pp. 1–4.

[34] O. Al-Jarrah and A. Halawani, “Recognition of gestures in Arabic sign language using neuro-fuzzy systems,” *Artif. Intell.*, vol. 133, no. 1–2, pp. 117–138, Dec. 2001.

[35] M. Maraqa and R. Abu-Zaiter, “Recognition of Arabic Sign Language (ArSL) using recurrent neural networks,” in *2008 First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT)*, 2008, pp. 478–481.

[36] M. Maraqa, F. Al-Zboun, M. Dhyabat, and R. A. Zitar, “Recognition of Arabic Sign Language (ArSL) Using Recurrent Neural Networks,” *J. Intell. Learn. Syst. Appl.*, vol. 4, pp. 41–52, 2012.

[37] P. R. Naoum, P. R. Naoum, D. H. H. Owaied, and S. Joudeh, “Development of a New Arabic Sign Language Recognition Using K-Nearest Neighbor Algorithm,” 1173.

[38] O. Sutton, “Introduction to k Nearest Neighbour Classification and Condensed Nearest Neighbour Data Reduction,” 2012.

[39] Torralba, Murphy, Freeman, and Rubin, “Context-based vision system for place and object recognition,” in *Proceedings Ninth IEEE International Conference on Computer Vision*, 2003, pp. 273–280 vol.1.

[40] A. Torralba, R. Fergus, and Y. Weiss, “Small codes and large image databases for recognition,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[41] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[42] N. Tubaiz, T. Shanableh, and K. Assaleh, “Glove-Based Continuous Arabic Sign Language Recognition in User-Dependent Mode,” *IEEE Trans. Human-Machine Syst.*, vol. 45, no. 4, pp. 526–533, Aug. 2015.

[43] “Advantages and disadvantages of hidden markov model.” [Online]. Available: https://www.slideshare.net/joshiblog/advantages-and-disadvantages-of-hidden-markov-model. [Accessed: 29-Jul-2018].

[44] H. Byun and S.-W. Lee, “Applications of Support Vector Machines for Pattern Recognition: A Survey,” *LNCS*, vol. 2388, pp. 213–236, 2002.

[45] “204.6.8 SVM : Advantages Disadvantages and Applications – Statinfer.” [Online]. Available: https://statinfer.com/204-6-8-svm-advantages-disadvantages-applications/. [Accessed: 28-May-2018].

[46] F. Ufldl, “Softmax Regression,” pp. 3–7, 2015.

[47] D. E. Rumelhart and J. L. Mcclelland, “Parallel distributed processing: explorations in the microstructure of cognition. Volume 1. Foundations.” MIT Press,Cambridge, MA, 01-Jan-1986.

[48] B. A. Olshausen and D. J. Field, “Sparse coding of sensory inputs,” *Curr. Opin. Neurobiol.*, vol. 14, no. 4, pp. 481–487, Aug. 2004.

[49] D. J. Field, “What Is the Goal of Sensory Coding?,” *Neural Comput.*, vol. 6, no. 4, pp. 559–601, Jul. 1994.

[50] Jianchao Yang, Kai Yu, Yihong Gong, and T. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1794–1801.

[51] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun, “Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition,” in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[52] K. Labusch and T. Martinetz, “Learning Sparse Codes for Image Reconstruction.”

[53] P. Smolensky, “Information processing in dynamical systems: Foundations of harmony theory,” 1986.

[54] G. E. Hinton, “Training Products of Experts by Minimizing Contrastive Divergence,” *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, Aug. 2002.

[55] A. Torralba, R. Fergus, and Y. Weiss, “Small codes and large image databases for recognition,” 2008.

[56] A. Krizhevsky, G. H.-P. of the Thirteenth, and undefined 2010, “Factored 3-way restricted boltzmann machines for modeling natural images,” *proceedings.mlr.press*.

[57] G. Hinton, T. Sejnowski, and D. Ackley, *Boltzmann machines: Constraint satisfaction networks that learn*. 1984.

[58] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” 2009.

[59] G. H.-N. networks: T. of the trade and undefined 2012, “A practical guide to training restricted Boltzmann machines,” *Springer*.

[60] F. Huang, Y. Boureau, Y. L.-C. V. and, and undefined 2007, “Unsupervised learning of invariant feature hierarchies with applications to object recognition,” *ieeexplore.ieee.org*.

[61] G. H.-N. computation and undefined 2002, “Training products of experts by minimizing contrastive divergence,” *MIT Press*.

[62] H. Larochelle, Y. Bengio, … J. L.-J. of machine, and undefined 2009, “Exploring strategies for training deep neural networks,” *jmlr.org*.

[63] Jaworski, Maciej, et al. "Resource-Aware Data Stream Mining Using the Restricted Boltzmann Machine." International Conference on Artificial Intelligence and Soft Computing. Springer, Cham, 2019.

[64] Sanjaya, Prima, and Dae-Ki Kang. "Optimizing restricted Boltzmann machine learning by injecting Gaussian noise to likelihood gradient approximation." Applied Intelligence 49.7 (2019): 2723-2734.

تحليل وتمييز إشارات الصم والبكم باستخدام خوارزمية التعلم الآلي العميق

إعداد: عهود عادل سلامة درابيع

إشراف: د. عبد الله كمال

الملخص

لغة الإشارة هي أفضل وسيلة للتواصل والتخاطب بين الأشخاص ذوي الإعاقات السمعية، إيماءات اليد تساعدهم في إيصال أفكارهم والتخاطب فيما بينهم طيلة أيام حياتهم. في مجتمعنا لغة الإشارات العربية معروفة فقط لدى الأشخاص ذوي الإعاقة السمعية والمختصين الذين يشرفون على تعليمهم، لهذا يبقى مجتمع ذوي الصم والبكم محدوداً وضيق الأفق.

مؤخراً عملت أبحاث على توثيق لغة الإشارة للصم والبكم العربية وتمييزها بالحاسوب باستخدام خوارزميات التعرف والتمييز الآلية. هذه الرسالة تقدم نظاماً حقيقي الوقت يعمل على تحليل وتمييز الإشارات لأحرف اللغة العربية باستخدام خوارزمية التعلم الآلي العميق والصور المتناهية الصغر.

خوارزمية التعلم الآلي العميق لديها القدرة على تشفير الصور واستخلاص ملامح وخصائص الإشارات من قاعدة بيانات كبيرة تحتوي على صور لتلك الإشارات، استخلاص الملامح والمميزات تتم بشكل متكرر حتى الوصول إلى ملامح متفرقة تمثل كل مجموعة منها ملامح حرف من حروف اللغة العربية وإشارته الخاصة فيه في عالم الصم والبكم.

هدفنا في هذه الرسالة هو تمييز إيماءات وإشارات الصم والبكم التي تمثل أحرف اللغة العربية وذلك عن طريق جمع قاعدة بيانات كبيرة تحتوي على صور تمثل هذه الإشارات، ومن ثم القيام بمعالجة هذه الصور وتحليلها وتبسيط محتوياتها واستخدام الخوارزمية السابقة الذكر لاستخلاص الخصائص والمميزات لتلك الإيماءات ومن ثم تصنيفها في مجموعات كل مجموعة تمثل حرفاً من حروف اللغة العربية.