Republic of Iraq
Ministry of Higher Education and
Scientific Research
University of Anbar
College of Computer Science and
Information Technology
Department of Computer Science

Applying Convolutional Neural Network Modified Based on Particle Swarm Optimisation Method for Image Retrieval

A Thesis
Submitted to the Department of Computer Science - College of Computer Science and Information Technology, University of Anbar as a Partial Fulfillment of the Requirements for Master Degree of Science in Computer Science

By

Sameer I. Ali Al-Janabi

Bachelor of Computer Science

Supervised By

Prof. Dr. Sufyan T. Faraj Al-Janabi

Prof. Dr. Belal Al-Khateeb

1441A.H. *2020 A.D.*

اسم الطالب: سمير اسماعيل علي
كلية الحاسوب وتكنولوجيا المعلومات ـ قسم علوم الحاسبات
عنوان الرسالة: تطبيق الشبكة العصبية الملتوية المعدلة على أساس طريقة تحسين سرب الجسيمات لاسترجاع الصور

Supervisor's Certification

I certify that I read this thesis entitled " ***Applying Convolutional Neural Network Modified Based on Particle Swarm Optimisation Method for Image Retrieval*** *" that was carried out under my supervision at the Department of Computer Science of the University of Anbar, by the student "****Sameer I. Ali Al-Janabi****" and that in my opinion it meets the standard of a thesis for the degree of Master of Science in Computer Science.*

Signature :
Name : ***Prof. Dr. Sufyan T. Faraj Al-Janabi***
Date : / /2020

Signature :
Name : ***Prof. Dr. Belal Al-Khateeb***
Date : / /2020

Certification of the Examination Committee

We the examination committee certify that we have read this thesis entitled " ***Applying Convolutional Neural Network Modified Based on Particle Swarm Optimisation Method for Image Retrieval*** *" and have examined the student "* ***Sameer I. Ali Al-Janabi*** *", in its contents and what is related to it, and that in our option it is adequate to fulfill the requirements for the degree of* ***Master of Computer Science***.

Signature:
*Name:****Assist. Prof. Dr. Loay Edwar George*** ***(Chairman)***
Date: / / 2020

Signature:
*Name****: Assist. Prof. Dr. Esam Taha Yassen*** ***(Member)***
Date: / / 2020

Signature:
Name: ***Assist. Prof. Azmi Shawkat AbdulBaqi*** ***(Member)***
Date: / / 2020

Signature:
Name: ***Prof. Dr. Sufyan T. Faraj Al-Janabi*** ***(Supervisor)***
Date: / / 2020

Signature:
Name: ***Prof. Dr. Belal Al-Khateeb*** ***(Supervisor)***
Date: / / 2020

Approved by the Dean of the College of Computer Science and Information Technology, University of Anbar.

Signature:
Name: ***Assist. Prof. Dr. Salah Awad Salman***
Title: ***Dean of the College***
Date: / / 2020

Student name: Sameer I. Ali Al-Janabi

Thesis title: Applying Convolutional Neural Network Modified Based on Particle Swarm Optimisation Method for Image Retrieval

Abstract:

Analysis of image contents has become one of the important subjects in modern life. In order to recognize the images in efficient way, several techniques have appeared and periodically enhanced by the developers. Image Retrieval (IR) becomes one of the main problems that face the computer society inside the revolution of technology.

To increase the effectiveness of computing similarities among images, hashing approaches have become the focusing of many programmers. These approaches convert images to strings of float numbers hash code. Indeed, deep learning (DL) in the past few years has been considered to be the backbone of image analysis using a convolutional neural network (CNN).

This work considers experimentation to find the best configuration of the sequential model for classifying images, beginning with four fully connected layers and ending with two layers. The best performance has been obtained in two layers the first layer consists of 128 nodes and the second layer is 32 nodes, where the accuracy reached up to 0.9012. This enables the design of high-performance image classifiers that can be applied several applications such as autonomous car driving systems.

Designing image classifiers has been achieved using CNN and the data extracted from CIFAR-10 dataset using inception model, these data are called transfer values (TRVs). Also, the Particle Swarm Optimization (PSO) algorithm is used to reduce the TRVs by generating templates. Each template has set of zeros and ones and the dataset is reduced according to this template. Finally, the TRVs are used to build models with high performance.

In this respect, the focusing of this thesis is to reduce the TRVs in order to obtain high performance image classifier models. Indeed, the PSO algorithm has been enhanced using crossover technique from genetic algorithm to obtain image

classifiers with high accuracy. This result reduces the complexity of models in terms of number of parameters used and the execution time.

The conducted tests showed the performance of the proposed systems, because these systems need less time for training and classification, also the accuracy still near to the original accuracy because the dataset has small number of features comparing to the number of TRVs.

Keywords: ***Image Retrieval (IR), Deep Learning (DL), Convolutional Neural Network (CNN), Hashing Techniques, CIFAR-10 dataset, Transfer Values, Particle Swarm Optimization.***

Sameer I. Ali Al-Janabi

2020

Acknowledgements

All praises to Allah Almighty, who enabled me to complete this work successfully.

I wish to express my deep respect and thank to my supervisors Prof. Dr. Sufyan T. Faraj Al-Janabi and Prof. Dr. Belal Ismael Alkateeb for their appreciable advice, important comments and support during the research.

Special thanks to "all my teachers in the College of Computer Science and Information Technology" for everything.

I am grateful to the staff of the College of Computer Science and Information Technology

My thanks for all…

Dedication

This thesis is dedicated to:

My parents

My wife

My supervisors

My teachers

My brothers

My sisters

My relatives

and my friends.

Sameer I. Ali Al-Janabi
2020

Contents

List of Tables

List of Figures

Abbreviations

ANN	**Artificial Neural Network**
BOF	**Bag-of-Features**
BT	**Bootstrap Aggregation Trees or Bagging Trees**
CBIR	**Content-Based Image Retrieval**
CNN	**Convolutional Neural Network**
CNNH	**Convolutional Neural Network-Based Hashing**
DCH	**Deep Convolutional Hashing**
DL	**Deep Learning**
DNNH	**Deep Neural Network Hashing**
FastH	**Fast Supervised Hashing**
FSNS	**fast sparse technique for neurons selection**
IM	**Inception Model**
IR	**Image Retrieval**
JSH	**Jointly Sparse Regression**
KSH	**Supervised Hashing with Kernels**
MAP	**Mean Average Precision**
MKL	**Multiple Kernel Learning**
MLAGH	**Metric Learning-based Anchor Graph Hashing**
NNs	**Neural Networks**
PCA	**Principal Component Analysis**
PCA-H	**Principal Component Analysis Hash**
PSO	**Particle Swarm Optimization**
QaDWH	**Query-adaptive Deep Weighted Hashing**
Q-OMP	**Quaternion Orthogonal Matching Pursuit**
RBF	**Radial Basis Function**
RBM	**Restricted Boltzmann Machine**
ReLU	**Rectified Linear Unit**
SDH	**Supervised Discrete Hashing**
SDPH	**Supervised Deep Hashing**

SDPH+	**Supervised Deep Hashing Plus**
SGD	**stochastic gradient descent**
SKHL	**Semi-supervised Kernel Hyperplane Learning**
SSDH	**Semi-Supervised Deep Hashing**
SSPDH	**Supervised Semantics-Preserving Deep Hashing**
SSMDH	**Semi-Supervised composite Multi-view Discrete Hash**
ST-RPNN	**Siamese-Twin Random Projection Neural Network**
TRVs	**Transfer Values**

Chapter One
Introduction

Chapter One: General Introduction

1.1 Introduction

Computer vision has recently turned out to be essential technology because of its wide-going applications in various areas including diverse and smart monitoring, health and medicine, sports and entertainment, robotics, drones, and self-driving cars. Visual recognition errands, for example, image classification, localization, and detection are the center building blocks of a significant number of these applications. Ongoing improvements in Convolutional Neural Networks (CNNs) have prompted the remarkable execution in these state-of-the-art visual recognition tasks and systems [1], [2]. On the other hand, the advances in Deep Learning (DL) over the most recent years have made special infiltration in a few zones, especially in computer vision, in which machine insight has gone past human execution. The deep architecture joins the low-level features to abstract high-level characteristics with nonlinear transform. This results in the required power and ability to take in the semantic representation from images [3]. For example, deep reinforcement learning has been utilized in self-driving vehicles to solve in choices by utilizing criticism from numerous sorts of sensors around the vehicle [4].

Another strategy that is utilized in the image retrieval framework is a hash function, or compression function, which means the output is being shorter than the input. Hash functions are already utilized in many applications including numerous cryptographic tasks [5]. There are three fundamental aspects of the cryptographic hash functions that are used in the security field: preimage resistance, second preimage resistance, and collision resistance [6].

The use of hashing in image retrieval techniques is after analyzing image and finding the values of features to design image classifier.

1.2 Classification of Image Retrieval Techniques

There are several techniques used in the system of Image Retrieval (IR). Those methods may be classified according to the learning method into three categories: Supervised, unsupervised, and semi-supervised techniques, as shown in Figure 1.1.

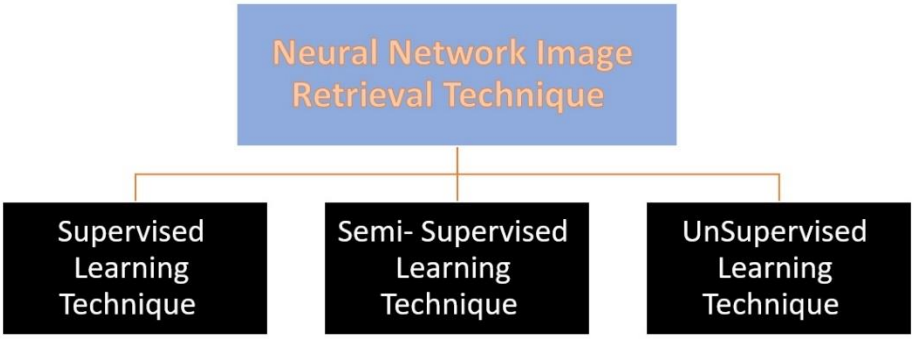

Figure 1.1: Main Classification of Neural Network Based Hash Encoding for Image Retrieval Techniques.

The emphasis of this thesis is mainly on supervised techniques due to their importance and wide-spreading use in many applications. Where this technique uses labelled data for learning models which make the learning method faster.

The semi-supervised category used both methods (supervised and unsupervised) in models.

Unsupervised techniques use no labeled data where the model train while working in real world as in Chess game.

In general, there are some challenging limitations of most IR techniques. These include the following [7]:

- Prediction semantic features from primitive features in the image or video.
- Reaching to the relationship between semantic features and primitive features.
- Indexing the contents of multimedia by reducing the needing to the human interaction.
- Enabling the system to understand and interpret the user request semantically.

Furthermore, it is possible to recognize several advantages and disadvantages of neural network based hash encoding, as summarized below:

Advantages:

1- Security in hash function generation [8].
2- Speed of hash function generation [8].
3- Extracting patterns and detecting behavior that are complex for noticing by humans or another computer technique [9].

Disadvantages

1- The neural network needs for training to work.
2- High processing time also is needed for large size neural networks [10].

1.3 IR techniques based on CNN with hash encoding

Over the past decade, there has been significant research effort dedicated to the development of intelligent driver assistance systems and autonomous vehicles, which is intended to enhance safety by monitoring the on-road environment [11]. Image content analysis as an important environment sensing modality has immensely progressed in recent years. One of the fundamental problems in image content analysis to efficiently recognize the environment is retrieving relevant contents from a large different scene database [12], which encourages approximate nearest neighbor search prosperous [13].

To reduce the computational cost in calculating similarities, hashing techniques have attracted broad attentions in the Big Media research area due to the efficiency of compact binary codes [14], [15]. It aims to construct a series of hash functions to map data points from the original space into compact binary codes and preserve the data structure in the original space. Hashing is a powerful technique for nearest neighbor search with hamming distance computation [16]–[18], because bit-wise XOR operation is performed to calculate the individual similarity, which is advantageous for improving computational efficiency. In addition, the compact

binary codes are also beneficial for storage efficiency compared to real-valued representations.

Existing hashing techniques can be classified into two categories: data-independent [19]–[23] and data dependent [24]–[28]. For the first category, random projections are employed to map data points into a feature space, then binarization are performed. For the second category, various statistical learning techniques are utilized to learn hash functions. In the pipelines of most existing hashing methods, input image is firstly represented by a vector of hand-crafted visual descriptors [29], [30] to capture the image semantics against image noise and redundant information [31]. Secondly, the projection and quantization steps are employed to encode the vector into a binary code.

The retrieval performance of conventional hashing methods is limited, mainly resulting from two aspects: on the one hand, the fixed hand-crafted features represent the visual similarities of images rather than the semantic similarities [32]. On the other hand, feature representation and projection are mostly studied as two separated problems, which leads to the suboptimal binary codes generated, as the hand-crafted feature representation is not optimally compatible with the binary codes. Recent revolution in Deep Learning [33] [34] shows the impressive feature representation power of (CNN) [35]–[37], which has been demonstrated by the progress in many visual tasks, such as image classification [35], [38], [39], object detection [40], face recognition [41] and many others [42], [43].

The accomplishments are attributed to the ability of CNN, which can learn the rich mid-level image representation to capture the semantic information [44]. Hashing techniques also benefit from the improvement of CNN to obtain high-quality binary codes with the semantic features of images. Recently, several CNN-based hashing methods have been proposed, such as CNNH [45], DNNH [6] and others [46]–[50], which have testified the satisfactory performance of binary codes obtained by CNN-based hashing. With the development of CNN, it is necessary to study new algorithms to learn more effective binary codes with less bits and make full use of supervised information to capture more representative features.

1.4 Related Work

The most important related work of this thesis can be divided into two sub-sections: the first one for Conventional Hashing while the second one is for Deep Hashing.

1.4.1 Conventional Hashing

Hashing technique is becoming an important part for approximate nearest neighbor search. The first type of these methods generates hash functions randomly that is not based on any training data. The method proposed by A. Gionis *et al*. [19] in 1999, that called Locality-Sensitive Hashing (LSH), which is considered as one of good method in this category, because it uses random linear projections for mapping data into binary codes. It has been proven that it is necessary to increase code length to reduce the Hamming distance between two binary codes which satisfy better performance.

The second type of these methods uses training data to learn similarity preserving hash functions, that also can be divided into unsupervised and supervised methods, based on supervised information. Spectral Hashing (SH) designed by Y. Weiss *et al*. [24] in 2009 is an example of un supervised methods, which get balanced binary codes by spectral graph partitioning solving problem.

Another method was proposed by Wang *et al*. called PCA-Hash (PCA-H) [25] in 2010, where this method is a data-dependent projection learning such that every hash function is built to sequentially correct errors made from the past one. Also, Gong and Lazebnik [26] designed an Iterative Quantization (ITQ) method in 2011, by simultaneously maximizing the variance of each bit, and minimizing the quantization error of mapping data to the vertices of a binary hypercube.

Supervised methods are designed to learn hash functions and getting semantic similarity. SDPH approach is an important approach in supervised approaches[4].

Also supervised hashing has better accuracy comparing to unsupervised methods in several applications which made supervised techniques focusing of the researchers [5]. In this respect, several architectures had been proposed, such as Supervised Hashing with Kernels (KSH) which was designed in 2012 by Liu *et al*.

[51]. This architecture is a kernel-based method. The idea of this method is learning binary codes by reduction hamming distance between similar pairs and increasing the distance between dissimilar pairs.

Another method called Supervised Discrete Hashing (SDH) [27] proposed by Fumin S. *et al.* in 2015, This method takes the advantages of label information to get the binary codes by combining hash codes generation and training of the classifier. Also the method called Column Sampling based Discrete Supervised Hashing (COSDISH) proposed by W.-C. Kang *et al.* in 2016 [28] is a discrete supervised hashing approach which can impact all points of the training data to solve FastH (Fast Supervised Hash) problem [52] that cannot use all training points because of large time complexity.

1.4.2 Deep Hashing

DL is a set of techniques that used to design and learn artificial intelligent systems and extract important features of images in efficient way. Recently CNN have been utilized in IR field. Krizhevsky *et al.* in 2012 [35] used the seventh layer features in the classification model, which has satisfy high performance on ImageNet.

Semantic hashing model is considered the first model that used DL for hashing. However, this model utilizes Restricted Boltzmann Machine (RBM) in binary codes learning which is not suitable for real applications.

With the boosting studies of CNN for image classification, CNN-based hashing is researched recently. Xia *et al.* [45] propose a supervised hashing method CNNH in 2014 to learn compact binary codes which takes CNN to learn a set of hash functions for the first time, and demonstrate the possibility of CNN applying to hash. However, CNNH is a two-stage method, a matrix-decomposition algorithm applied for learning binary codes in the preprocessing stage. It is unfavorable when the data size is large.

After that effective and more deep networks are designed by K. Simonyan and A. Zisserman in 2014 [53] and S. R. Christian Szegedy *et al.* in 2015 [38].

Moreover, the learned image feature cannot be used to learn better binary codes due to the separated stages. Subsequently, Lai et al. in 2015 [6]improve CNNH by proposing a one-stage CNN-based hashing method DNNH for

simultaneous feature learning and hashing coding, which enforces the image representation and hash coding to improve each other in a joint learning process. It presents better performance on several benchmarks. However, many hyper-parameters need to be adjusted in this model for better performance. The gap is to reduce the time of processing to make the models of classifier more efficient.

1.5 Problem Statement

Recently with the wide use of intelligent systems applied in several applications in our life, IR systems became an important category of such systems. The working method of IR system is composed of DL techniques using CNN and images datasets to learn this system to behave as human and take suitable decision.

There are several requirements to make this system successful: firstly, extracting important Transfer Values (TRVs) from images; secondly, learn the system by using images datasets to classify new images that is face it in real world. Thus, powerful computers are needed to executes these systems. On the other hand, when using such systems in the real world, they should be operated on devices that might have limited resources in terms of RAM (Random Access Memory) and CPU (Central Processing Unit). For example, in autonomous car driving system there is a generation of big data resulting from movement of car and taken images that should be processed in order to make a quick decision based on the analyzing the contents of these images.

To address these problems, IR systems should be designed such that to be of high performance and can also operate in limited resources devices. Indeed, the computation complexity of images retrieval systems hence must be reduced in terms of various parameters used. So, enhanced PSO algorithm is used to generate templates for reducing TRVs in datasets and new images for classifying.

1.6 Aim of Thesis

The main objectives of this work are as follows:

- Designing images classifiers with high performance used for classifying images in efficient way and can be applied several applications such as autonomous car driving systems.

1.7 Contributions

- Making experiments to test best configuration of the sequential model for classifying images, beginning with four layers and ending with two layers, the best performance was obtained in two layers the first layer consists of 128 nodes and the second layer is 32 nodes, the accuracy reached to 0.9012.

- Using the Particle Swarm Optimization (PSO) algorithm to reduce (TRVs) - Output of the Inception Model (IM) [54]- and use these values to build and train models of classification in order to get a better performance. The common work in this field is to reduce the features of images but because the features of CIFAR-10 dataset [55]. The focusing of this thesis is on TRVs to reduce it and obtain high performance images classifier models.

- Enhance the PSO algorithm using crossover technique from genetic algorithm to obtain high accuracy. This result reduces the computational complexity of models in terms of number of parameters used and the execution time.

1.8 Thesis Structure

The next chapters of this thesis are organized as follows:

Chapter Two provides the background of IR systems that use CNN based hash encoding. These can be classified into three categories: Supervised, unsupervised, and semi-supervised techniques according to each technique's learning method.

Chapter Three contains a full description of the proposed system in terms of the algorithms and measures used to implement the approaches of IR system and achieve the desired goal.

Chapter Four presents the results that are obtained through experimentation with the proposed approaches. Indeed, those results are discussed.

Chapter Five presents the main conclusions of this research in addition to some suggestions for future works.

Chapter Two
Theoretical Background

Chapter Two: Theoretical Background

2.1 Introduction

An IR system is a computer system for browsing, searching and retrieving images from a large database of digital images. A lot of IR common approaches use methods for adding metadata such as captioning, keywords, title or descriptions to the images so that retrieval can be executed over the annotation words. Manual image annotation is needing a lot of time to be suitable for IR. To address this, there has been a large amount of research done on automatic image annotation. Additionally, increasing of social web applications and the semantic web have led to the development of several web-based image annotation tools.

Several techniques have appeared in this field. However, the most common of these techniques are using neural network-based hash encoding, which can be categorized into three main classes: Supervised, unsupervised, and semi-supervised techniques according to each technique's learning method. The most important related works appeared in the literature are reviewed and constructive comparisons have been done to show the strengths and limitations of various techniques.

2.2 Deep Learning

Deep learning is a machine learning research area that is based on a particular type of learning mechanism. It is characterized by the effort to create a learning model at several levels, in which the most profound levels take as input the outputs of previous levels, transforming them and always abstracting more. This insight on the levels of learning is inspired by the way the brain processes information and learns, responding to external effects [56]. There are two important parameters used in experiments: accuracy and loss.

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition

Equation 2.1: Calculating Accuracy[57]:

$$\boldsymbol{Accuracy} = \frac{\textbf{Number of correct predictions}}{\textbf{Total number of predictions}}$$

the loss function is one of the most important steps to ensure the model will work in the intended manner. The loss function can give a lot of practical flexibility to your neural networks and it will define how exactly the output of the network is connected with the rest of the network. The loss value can be calculated as shown below:

Equation 2.2: Calculating Loss [58]

$$\boldsymbol{loss = predicted\ value - true\ value}$$

2.3 CNN

(CNNs) in essence are neural networks that employ the convolution operation (instead of a fully connected layer) as one of its layers [59]. CNNs are an incredibly successful technology that has been applied to problems wherein the input data on which predictions are to be made has a known grid like topology like a time series (which is a 1-D grid) or an image (which is a 2-D grid) [60].

CNN have completely dominated the machine vision space nowadays. A CNN consists of an input layer, output layer, as well as multiple three hidden layers. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers (Rectified Linear Unit-ReLU). Additional layers can be used for more complex models.

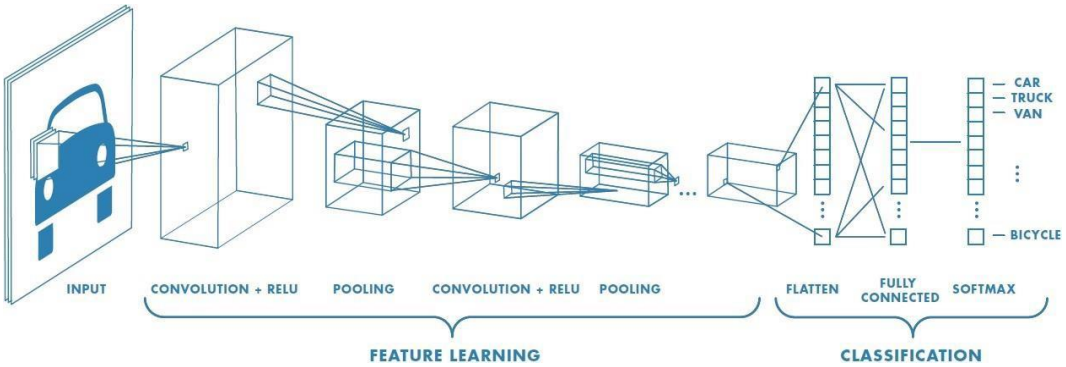

Figure 2.1: Typical CNN architecture [91]**.**

The CNN architecture has shown excellent performance in many Computer Vision and Machine Learning problems. CNN trains and predicts in an abstract level, with the details left out for later sections. This CNN model is used extensively in modern Machine Learning applications due to its ongoing record-breaking

effectiveness. Linear algebra is the basis for how these CNNs work. Matrix vector multiplication is at the heart of how data and weights are represented. Each of the layers contains a different set of characteristics for an image set. The CNN is the core of IM working as shown in Figure 2.2

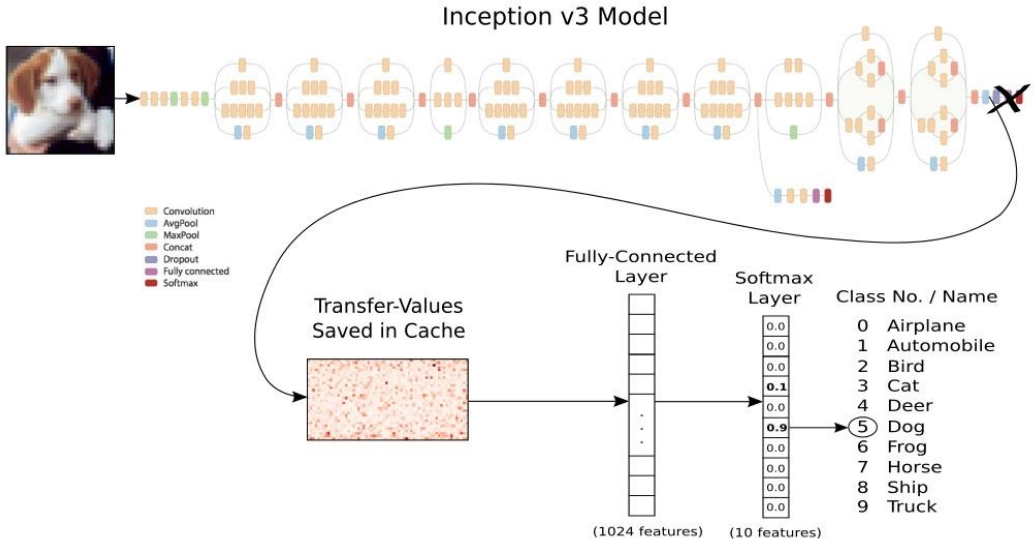

Figure 2.2: IM with TRVS [54]**.**

2.4 Particle Swarm Optimization (PSO)

PSO is a technique used for selecting the best solution from set of solutions (swarm) according to value called fitness, where this value may be maximum or minimum based on the type of the problem [61]–[65]. Each solution has two factors: position and velocity. The position represents how the values of zeros and ones are distributed in the solution. The position of each solution can be updated according to velocity value which can be calculated using **Equation 2.3**. where, if the value of velocity is greater than 0.5 then the position of the solution will change as follow: if the value in the template is zero then it will be one and vice versa. From the other hand, if the value of velocity is less than 0.5 then the position of the solution will not change.

Equation 2.3: Updating velocity of the solution[62]**.**

$$V(new) = (W * V(old)) + (C1 * R1) * (pbest - P(old) + (c2 * r2) * (gbest - P(old)$$

Where *V(new)* is the new velocity of the solution t, w is random number between (0.2,0.9), r1, r2 are random number between (0.0,1.0), P[t] is the position of the solution, c1 and c2=2.0, *pbest* is the the solution with best fitness in the iteration and gbest is the solution with the best fitness in all the swarm.

2.5 Supervised Learning Techniques

The approaches of supervised traffic classification must have a pre-labelled dataset for training. A classifier is trained in the feature space via the training dataset and will be applied for classifying new network traffic. Numerous researches were performed for solving a variety of traffic classification issues with the use of supervised approaches [1]. In this section, the common architectures for supervised learning methods of IR will be described with examples for each architecture.

2.5.1 One-Stage Supervised Deep Hashing (SDHP)

Qi Li *et al.* suggested a technique that includes developing a deep supervised discrete hashing approach which is modelled on the basis of the hypothesis that the learned binary codes must be optimal for classification. Each of the classification information and the pairwise label information are utilized for learning the hashing codes within one stream model [3].

This was achieved by constraining the results of the past layer to be directly binary codes, and that has been seldom researched in the deep hashing approach. Due to the fact that hash codes have discrete nature, an alternating minimizing approach is utilized for optimizing the objective function. This might be quite a useful approach for applications of image or video search because it is computationally inexpensive and storage efficient.

The Mean Average Precision (MAP) outputs of SDH+CNN and FastH+CNN on CIFAR-10 dataset are 0.553 and 0.604, respectively. The average MAP output of this approach on CIFAR10 dataset is 0.787, and that performs better than previous conventional techniques of hashing with deeply learned features. Indeed, the suggested approach accomplishes a comparable efficiency to the optimal

conventional approaches of hashing on NUS-WIDE dataset under the first experimental setting.

Another method proposed by Dongbao Yang *et al.* called a one-stage supervised hash approach for learning high-quality binary codes. Those researchers executed a deep CNN and enforced the learned codes for satisfying the following points: (a) the binary codes must be distributed uniformly; (b) images that are similar must be encoded to similar binary codes, and the other way around; (c) quantization loss must be minimized. Experimental comparisons between this approach and previous algorithms had been conducted on NUS-WIDE and CIFAR-I0 datasets [4].

The MAP of this approach reached to 87.67% and 77.48% on CIFAR-I0 and NUS-WIDE datasets respectively with 48-bit. Thus, this method improved the search accuracy. The contribution of this work mainly focused on four aspects: At first, the GoogleNet architecture gives to the developers the ability to put their own equations in the classification layer because of its highly efficient feature representation power. Secondly, a pairwise loss function has been devised to maintain the semantic similarity of the original data. Thirdly, enforcing the binary codes uniformly spread for carrying more information. Fourth, the loss of quantizing from Euclidean to Hamming space has been diminished.

Also, Chenggang Yan *et al.* developed the previous architecture for learning high-quality binary codes, and in the same framework, the method was extended into what they called SDHP+ for improving the discriminative power of binary codes. Implementation of this approach used a deep CNN, and the learned codes were enforced for satisfying the following criteria: Similar binary codes must result from similar images, and vice versa, minimizing the loss of quantization from Euclidean to Hamming spaces, and the learned codes must be distributed in a uniform manner. The application of this approach was for the efficient recognition of the on-road environment based on the analysis of the contents of the images from the large-scale scene database [5].

Experimental comparisons of this method with some previous hashing algorithms had been performed on NUS-WIDE and CIFAR-10. The MAP of SDHP reached 87.67% and 77.48% with 48 b (binary code), respectively, and the MAP of SDHP+ reaches 91.16%, 81.08% with 12 b, 48 b on CIFAR10 and NUS-WIDE,

respectively. It can be concluded the fact that the latter approach described in [5]outperformed the others because the enhancement in this method called (SDPH+) had improved the search accuracy.

2.5.2 Nearest Neighbour Search

For the majority of the available hashing approaches, an image is initially encoded in a form of a vector of hand-engineering visual characteristics, which is followed by one more step of separate projection or quantizing which produces binary codes.

Hanjiang Lai *et al.* [6] proposed a single-stage supervised hash approach for retrieving images that generated bitwise hash codes for images by a thoroughly designed deep model. The suggested deep architecture used a triplet ranking loss designed in order to keep relative similarities. In this approach, the input images were converted to unified representations of image through a shared subnetwork of stacked convolution layers. After that, those intermediate image representations were encoded to hash codes via "divide-and-encode" modules, as shown in Figure 2.3

Empirical evaluations in IR showed the fact that the suggested approach had achieved better performance gains compared to some previous supervised or unsupervised hashing approaches. The method achieved better search accuracies (accuracy within Hamming distance 2, MAP, accuracy with varying size of top returned samples, and precision-recall) than the ones that are considered baseline approaches utilizing conventional hand-crafted visual features.

For instance, in comparison with the optimal competitor KSH, the MAP results of the suggested approach refer to a relative increasing of 58.8%-90.6 %, 61.3%-82.2 %, and 21.2%-22.7% on SVHN, CIFAR10, and NUS-WIDE datasets, respectively.

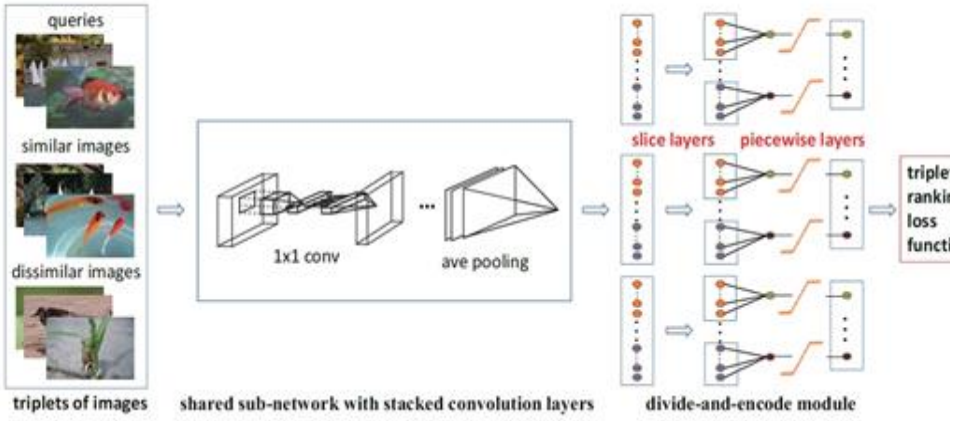

Figure 2.3: One-Stage Supervised Hashing method Architecture [6].

Manish Sapkota *et al.* designed a Deep Convolutional Hashing (DCH) approach which may be trained "point-wise" for a simultaneous learning of each of binary and semantic representations of histo-pathological images. Particularly, the researchers proposed a CNN which introduced a latent binary encoding (LBE) layer for low dimensional embedding of features for learning binary codes. This method included designing a joint optimizing objective function which encourages the network for learning discriminative representations from the label data, and diminish the gap between the desired binary values and the real-valued low dimensional embedded features [66].

The binary encoding for new images could be obtained through forward propagation via the NN and the quantization of the result of the LBE layer. Results of experimentations on a large-scale histo-pathological image dataset demonstrated the efficiency of the suggested approach. This architecture provided a fast image query and retrieval of related cases, this way could be helpful for specialists in the evidence-based research of the diseases for diagnosing. In addition to that, they could research the retrieved similar instances for understanding the biological and morphological properties of an illness.

A better ranking performance has been noticed for DCH with MAP that ranges between 0.94 and 0.96 utilizing various numbers of bits, and performing batter that the other considered approaches by 2%-4%. However, the results can be further enhanced by enhancing the loss of quantization with a more sufficient

optimization formulation. In addition to that, this work had the assumption that image labels have no noise, in other words, data annotation has been consistent. Label noise would negatively influence the learning of the suggested model and after that in the final diagnose of the disease. This is why, the research can be enhanced by developing noise-insensitive learning algorithms for training the network in addition to the statistical evaluation of the robustness of the suggested approach on the diagnosis of the disease.

Approximate nearest neighbour search could be an effective approach for large-scale retrieval of images. Jun-Yi Li and Jian-hua Li developed a sufficient DL framework approach for generating binary hash codes for time efficient IR based on CNN. The adopted concept was that it is possible learning binary codes with the use of a hidden layer for presenting the latent concepts that dominate the class labels in which the data labels can be used. In addition to that, CNN may be utilized for learning image representations [67].

Other supervised approaches need pair-wised binary code learning inputs. Nevertheless, this approach may be utilized for learning image representations and hash codes in a point-by-point way, for this reason, it is appropriate for large-scale datasets. Experimental results have shown the fact that this method is more efficient than a number of other hashing approaches on the MNIST and CIFAR10 datasets. The work can be extended by investigating its efficiency and scalability on a dataset of a larger scale.

The experimental results which indicated that this approach can enhance some earlier best retrieval results with 30% and 1% retrieval accuracy on the CIFAR-10 and MNIST datasets respectively with only a simple alteration of the deep CNN. This method provided 83.75% precision (which has been obtained by the last layer) on the task of 116 classes of clothing classification.

Another structure proposed by Siying Zhu *et al.* had merge the process of generating binary codes within deep NNs for sufficient retrieval of images. The suggested model included two main blocks. The stacked convolution layers of Network-In-Network with global average pooling for the calculation of the sufficient representation of images and the embedded latent layer with binary activation functions learn binary hash codes in a simultaneous manner [68].

Experiments have shown that the suggested approach has gained improvements over a number of earlier hashing approaches on two well-known DL datasets, which are MNIST and CIFAR-10. These researchers had implemented their suggested structure on the NN toolkit Caffe1. In all of the experiments, the networks have been trained via stochastic gradient descent with a learning rate of 0.01 and weight decay of 0.0005. The training iterations were 10,000 for each one of the datasets. The experimental results showed that the precision of the approach while using 48 bits is 0.98 and 0.587 on MNIST and CIFAR10 respectively.

Another joint binary code learning approach has been suggested by Xuelong Li *et al.* for combining image feature to latent semantic feature with as little encoding loss as possible. This was known as Latent Semantic Minimal Hashing. The latent semantic feature has been learned according to matrix decomposition for refining the initial feature, this way it makes the learned feature more discriminative. In addition to that, a minimum encoding loss has been combined with the process of latent semantic feature learning in a simultaneous way, in order to ensure that the obtained binary codes are also discriminative [69].

Extensive experiments on numerous common large databases have shown that the suggested method performed more efficiently compared to numerous other approaches of hashing. The conclusion is that this latter approach that has been described in [69] is, in general, more efficient compared to the rest of the approaches of this sub-section due to the fact that its retrieval precision has reached up to 0.98%.

2.5.3 Content-Based Image Retrieval (CBIR)

Huafeng Wang *et al.* have suggested an architecture based on content-based IR systems (CBIRs) which consists of three parts: feature extraction, processing and indexing, as shown in Figure 2.4. Via selecting the intermediate model layers like feature representation, and pre-processing the data with some important approaches, the CBIR task based on CNN could noticeably be enhanced. Retrieval performance was between 34.40% and 53.41% [70].

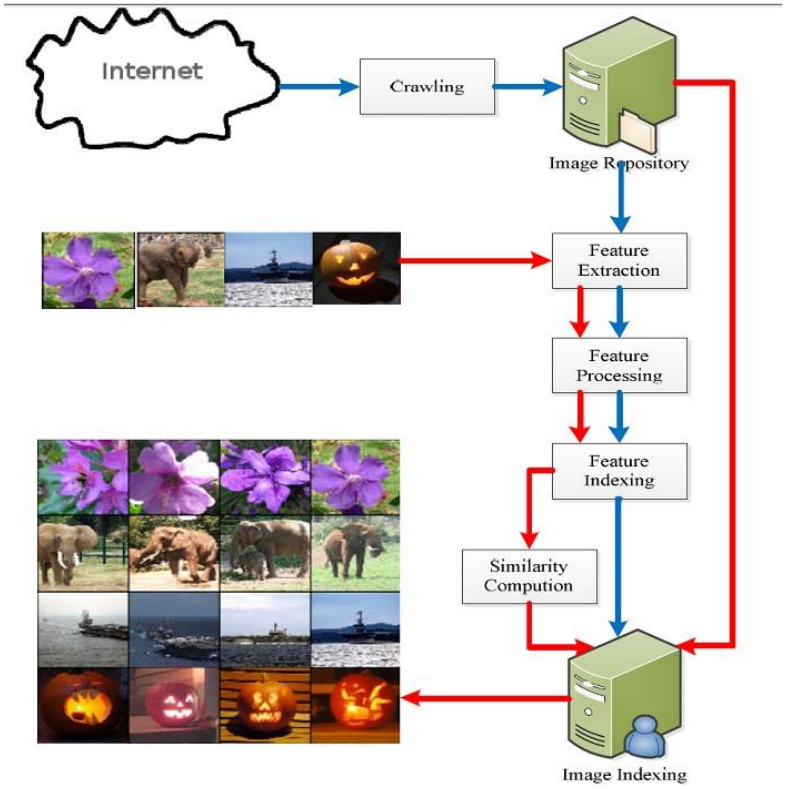

Figure 2.4: Simplified Industrial CBIR Architecture [70]**.**

A second method designed by Yang Li *et al.* called a deep feature hash codes model for CBIRs, where they initially extracted features of the image by a pretrained CNN model. Then, they used various hashing approaches for binary feature extraction. Lastly, they used the optimal binary encoding features for building a CBIRs [71].

The experimental results demonstrated that with decreasing feature dimension, the approach did not decrease the accuracy of retrieval and could as well improve the precision of retrieval in some of the cases. The precision of retrieval of 256 bits binary characteristics might exceed the conventional approach of 256 dimensional (4096 bits) features.

As soon as the feature bits were 16 times lower, the space of the storage had decreased 16 times and the performance of retrieval has increased. Which is why, this approach could sufficiently enhance the speed and accuracy of CBIRs. The efficiency of retrieval was in the range from 50.08 to 77.55. The work can be additionally improved with the addition of the approach of matrix learning in this framework and enhance the prcision of the CBIR retrieval.

A third method is large-scale remote sensing (RS) retrieval of images that had become a valuable research problem in geo-sciences. This method was proposed by Peng Li and Peng Ren [72]. It is compatible with the fast evolution of the technologies of satellite and aerial vehicle. Methods of Hashing-based searching have been commonly utilized in the tasks of content-based retrieval of images.

On the other hand, the majority of hash approaches make a trade-off between the precision of retrieval accuracy and the effectiveness of learning, and therefore can hardly meet the exact RS data analysis requirements. For the sake of addressing those drawbacks, they have proposed an approach for partial randomness in order to learn hash functions, which has been called as partial randomness hashing (PRH). Particularly, for the construction of hash functions, a part of model parameter values have been arbitrarily produced and the rest of them have been trained on the basis of the RS images.

Experiments on two large public RS image datasets had proven the fact that this PRH approach had outperformed several other related algorithms according to each of precision of retrieval and effectiveness of learning. Retrieval performance was between 0.4138 and 0.5202 using MAP. Finally, comparing these three methods to each other, one generally may conclude that **t**he second method [71] is superior to the other methods because it produced the best retrieval performance that reached to 77.55.

2.5.4 Neural Network based Hash Function Method for Authentication and Retrieval of Color Images

In this approach, two interesting methods were noticed. The first method was proposed by Yakup Kutlu and Abdullah Yayik [73]. The main idea of this method was re-sizing input colour image to a constant size, after that, generating hash values with the use of NN one-way feature and nonlinear approaches, for three dimensions respectively. This work proposed a NN-based hash function architecture for colour image authentication. The presented system presented binary and hexadecimal sensitivity of hash value. The binary sensitivity was nearly 50% that satisfies diffusion of the hash value. Also, almost 100% hexadecimal sensitivity obtained meaning that the algorithm has very high ability of robustness. The binary sensevity reached 51.36%

The second method designed by Yang Li and Zhuang Miao [74], which included combining non-linear reduction of dimension and hashing approach for effective retrieval of images. They firstly extracted 4096-dimension characteristics via a pretrained CNN model. After that, they have used t-Distributed Stochastic Neighbour Embedding (t-SNE) for the reduction of the deep characteristics to 1024-dimension. Lastly, they have utilized Sparse Projection (SP) for building 256 bits binary encoding characteristics for retrieving images. They assessed the efficiency of their approach with the Oxford-5k, Paris-6k and Holidays datasets. Experiments on those three datasets showed that the performance is reached to 81.35 using MAP. The retrieval performance of this latter method was higher than the previous one as it reached to 81.35.

2.5.5 Other Supervised Learning Techniques

In this subsection, some other interesting techniques for supervised learning will be describe.

A. Siamese Architecture

This approach was proposed by Abin Jose *et al.* for learning binary codes for fast retrieval of images. A Siamese model was used with 2 parallel feed-forward branches but with a shared weight for generating binary codes. The training data has been split to similar and different pairs. The NN attempts learning the weights in a way that it decreases the distance between similar pairs of images and increases the distance between pairs of images which are not similar. The binary codes have been formed via squashing the output of the NN through a sigmoid function of activation, as shown in Figure 2.5 [75].

The training with sigmoid hash constrained the result of every node in the final fully connected layer into either 1 or 0. The input has been fed to the NN in the form of image pairs. The result of the fully connected layer has been transmitted to the sigmoid function. The distance between the result feature vectors has been computed inside the function of loss. Retrieval performance reached to 67.19 on CIFAR-10.

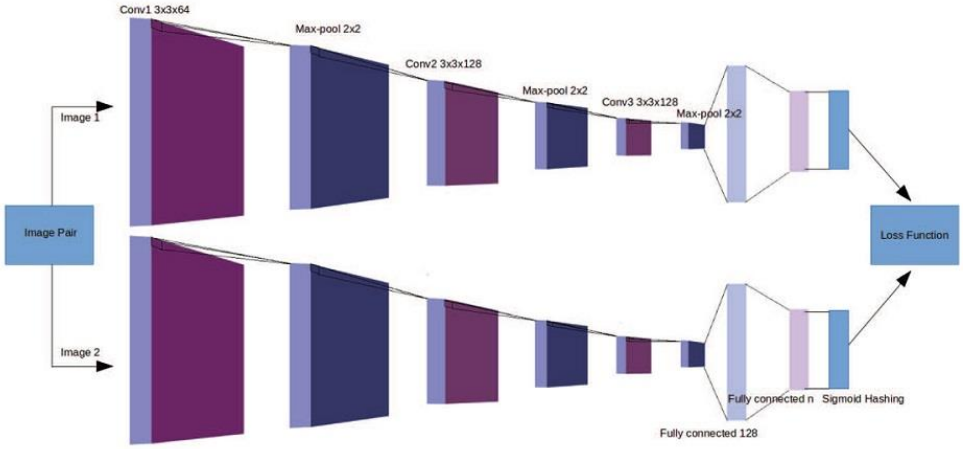

Figure 2.5: Architecture of the Siamese NN [75]**.**

B. Deep-Networks based Hashing for Multi-label Images

Another approach designed by Hanjiang Lai *et al.* focused on deep-networks-based hashing for images that have multi-label; every one of these images may contain objects of multiple categories, as shown in Figure 2.6. Also, in the most common hashing methods, the representation of each image consists of one piece of hash code, which is called semantic hashing. The performance of this approach reached to 0.8830 using MAP [76].

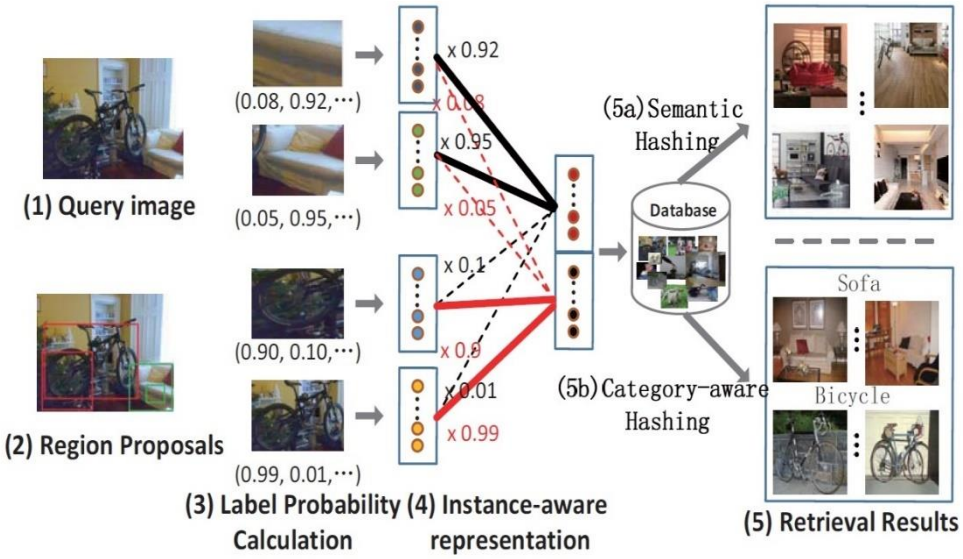

Figure 2.6: Illustration of Instance-Aware IR [76]**.**

C. The Model of Bag-of-Features (BoFs) & A Symmetry-Aware Spatial Segmentation Approach

Nikolaos Passalis and Anastasios Tefas [77] proposed the bag-of-features (BoFs) approach which consists of 3 layers: A radial basis function (RBF) layer, fully connected layer, and accumulation layer, as shown in Figure 2.7.

This module allows for split the size of the representation from the number of utilized code-words and also for a more sufficient formulation the scattering of features with the use of an independent trainable for every one of the RBF neurons.

The final network, referred to as the "retrieval oriented neural BoF" (RN-BoF) was trained by the use of the regular backpropagation and takes into consideration the quick extraction of conservative picture representations. The RN-BoF approach was capable of accelerating the speed of retrieval and encoding of objects, minimizing the size of obtained representation, and increasing the accuracy of retrieving.

Another technique called "A symmetry-aware spatial segmentation" was also proposed for minimizing the storage needing and time of encoding which make this approach capable of scaling large datasets in an efficient way.

The performance of this latter approach reached to 97.87 % using MAP. This work can be further enhanced by combining the RN-BoF model with extreme learning techniques for reducing the time of training.

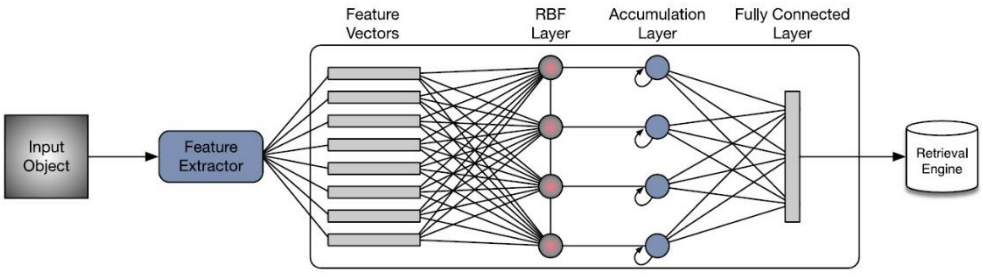

Figure 2.7: Proposed RN-BoF Model [77]**.**

D. Supervised Semantics-Preserving Deep Hashing (SSPDH)

Huei-Fang Yang *et al.* designed an approach called effective supervised deep hash approach, where the constructing of the binary hash codes was from labelled data for image search which is of large-scale. Those researchers assumed

the fact that the semantic labels are administrated via multiple interior specifications with every specification On or Off, and that the classification depends on those specifications [78].

Constructing hash functions was done as a latent layer in a deep NN. The binary codes were trained by reducing an objective function which has been defined over classification error and other wanted hash codes characteristics, as shown in Figure 2.8.

With this model, SSPDH achieved a good attribute where a single learning model contains both classification and retrieval parts [78]. The performance of this approach reached to 91.45 % and 99.39% on MNIST and CIFAR10 datasets respectively using MAP.

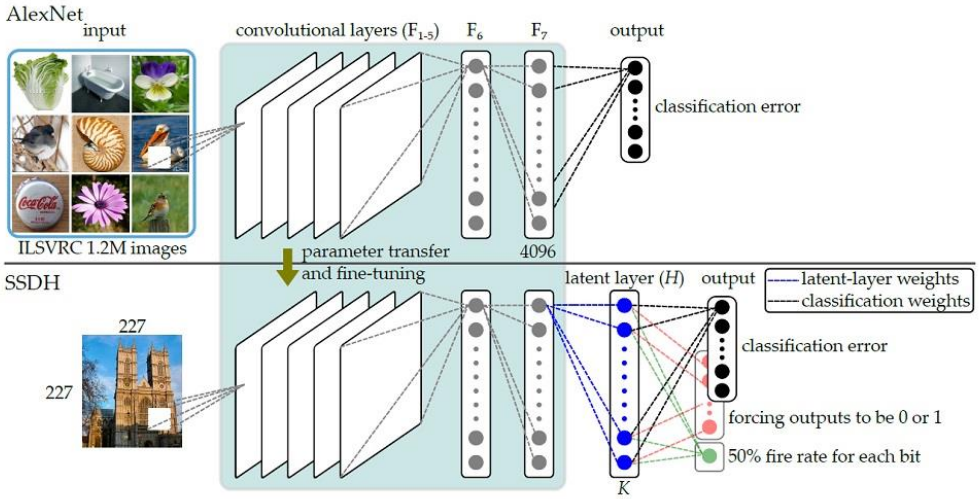

Figure 2.8: Supervised Semantic-Preserving Deep Hashing (SSPDH) [78]**.**

E. Bit-Scalable Hashing Approach

Another supervised learning architecture was proposed by Ruimao Zhang *et al.* [47], where the raw images used to directly create compact and bit-scalable hashing codes, as shown in Figure 2.9. Firstly, the triplet samples should be generated by organising the training images into a batch, and every sample includes 3 images where two of those images are in the same category while the third one is different. Then in these samples, the margin between the identical pairs and mismatch pairs should be maximized in the Hamming space.

The images of similar components must have similar codes, also in the resulting hashing code, each bit should be weighted individually. The performance of this approach reached to 63.26 %, 98.09% and 64.14 on MNIST, CIFAR10 and NUS-WIDE datasets respectively using MAP. This framework can be further enhanced via leveraging more semantics (*for instance,* numerous attributes) of images and/or introducing feedback learning in the framework such as to make it more powerful in practice.

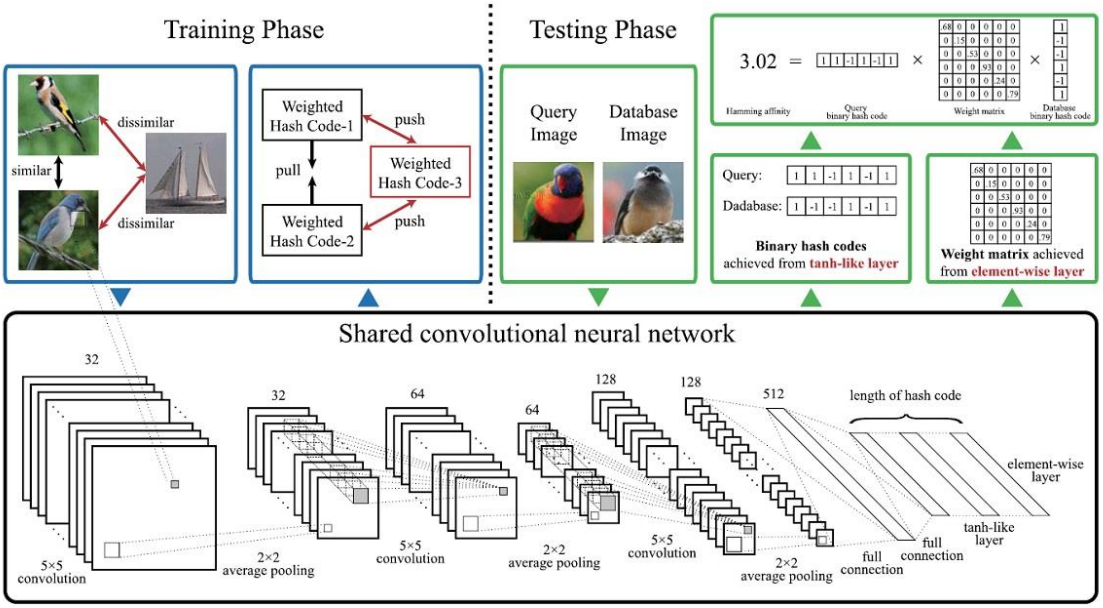

Figure 2.9: The Bit-Scalable Deep Hashing Learning Framework [47]**.**

F. Deep Hashing (DH) Method for Learning Compact Binary Codes for Scalable Image Search

This architecture was designed by Jiwen Lu *et al.* [79]. It is differed from most other learning techniques that it has multiple linear projection of mapping each image into binary feature vector, as shown in Figure 2.10: The Main Idea of Deep Hashing Method for Compact Binary Codes Learning [79].

Those researchers built up a deep NN to look for numerous hierarchical nonlinear transformations for learning the binary codes, with the goal that the non-linear correlation of tests may be all around misused.

They pointed out two main headings for future work: The first one is to stretch out this approach to deal with adaptable video search. And the second one is to stretch out this approach to deal with the cross-modular search for scalable

multimedia search. The Hamming ranking of this approach using MAP reached to 0.72 on CIFAR10 dataset.

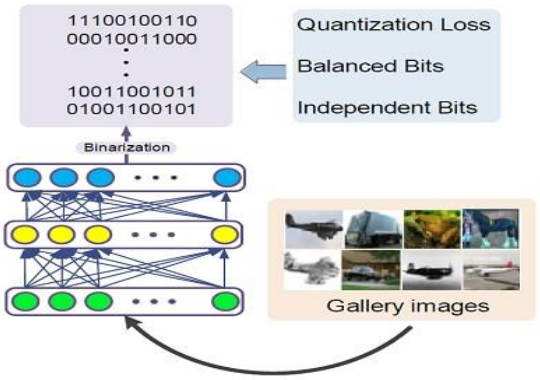

Figure 2.10: The Main Idea of Deep Hashing Method for Compact Binary Codes Learning [79]**.**

G. The Method of Query-adaptive Deep Weighted Hashing (QaDWH)

The query-adaptive deep weighted hashing (QaDWH) approach was proposed by Jian Zhang and Yuxin Peng [80]. This approach is capable of performing fine-grained ranking for various queries by weighted Hamming distance (See Figure 2.11). Here the researchers used an example of 6-bits hashing codes. Supposing that the query image has a hash code of 000000, there are six images within Hamming radius 1 with query image, however, they are different in various bits.

Conventional Hamming distance is not capable of performing fine-grained ranking amongst them. The proposed approach consists of two parts: First, designing a new deep hash network, consisting of 2 streams: the hashing stream learns the compact hashing codes and corresponding class wise hash bit weights in a simultaneous manner, whereas the classification stream maintains the semantic info and enhances the efficiency of the hash.

After that, designing a sufficient method of query-adaptive retrieval of images that initially rapidly creates the query-adaptive hashing weights based on the class-wise weights and the projected semantic probability of the query, and after that performs sufficient IR via weighted Hamming distance. As a future work, the researchers suggested extending the deep weighted hashing approach to a multi-table framework of deep hash, where various weights have been learned for various

functions of hashing map. The Hamming ranking of this approach using MAP reached to 0.884 on CIFAR-10 dataset.

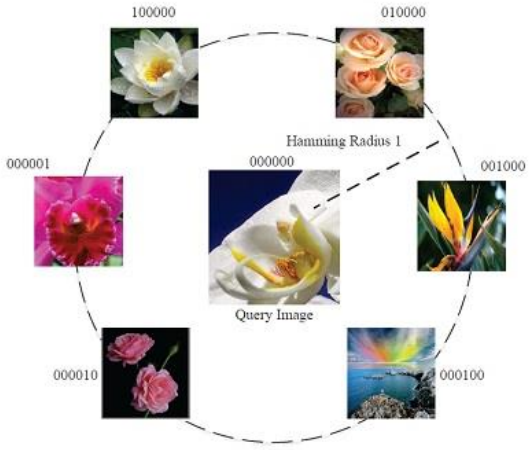

Figure 2.11: The Discrete Hamming Distance [80]**.**

H. Jointly Sparse Regression (JSH) Model

The purpose of this IR model was minimizing the loss of locality information and get jointly sparse hash approach. The presented approach by Zhihui Lai *et al.* integrated joint scarcity, locality and process of rotation together in a smooth construction [81].

The presented JSH could learn the best jointly sparse matrix of projection for low-dimensional feature extracting and selecting. With the orthogonal constraints in the form of a rotational operation between the binary codes and the low-dimensional characteristics, the suggested model could additionally reduce the loss of information and get the binary solutions in a direct manner The Hamming ranking of this approach using MAP reached to 0.1913 on CIFAR-10 dataset.

Comparing the eight latter techniques mentioned in this subsection, it can be noticed that the approach proposed by Nikolaos Passalis and Anastasios Tefas [77] achieved the best IR ration of 97.87 %.

2.6 Semi-Supervised Techniques

Usually, is not easy to obtain labelled data most of the time. In the case of using the label data alone for training, the training dataset would be excessively small for entirely reflecting the characteristics of the traffic. New applications of network are continuously developed, which results in traffic data with no label.

Each one of the situations results in issues for the conventional supervised ML approaches.

For addressing this issue, the semi-supervised ML approach has emerged, because of its capability of combining unsupervised and supervised learning. In semi-supervised learning, the training dataset includes each of samples that are labelled or unlabelled [1]. In this section, some of the most interesting semi-supervised learning proposals are reviewed.

2.6.1 Semi-Supervised Hashing for Scalable Image Retrieval

J. Wang *et al.* have suggested a semi-supervised hashing method which was formulated in a form of a minimization of empirical error on the labeled data whereas increasing variance and independence of hash bits over the labeled and unlabeled data. The suggested approach was capable of handling both metric in addition to the semantic similarity. This method can be enhanced via relaxing the widely utilized constraints of orthogonality in a way that one is capable of achieving better results, particularly for larger number of bits [82].

2.6.2 Semi-supervised Kernel Hyperplane Learning (SKHL)

Meina Kan *et al.* proposed a hashing approach that has been referred to as the SKHL for semantic image retrieving via the modeling of every hashing function as a non-linear kernel hyper-plane that has been constructed from an unlabeled dataset. In addition to that, a Fisher-like criterion has been suggested for learning the best kernel hyperplane and hashing function, with the use of only weakly labeled samples of training with side information [83].

For the sake of additionally integrating various feature types, they have incorporated multiple kernel learning (MKL) in the suggested SKHL (referred to as the SKHL-MKL), that led to a more sufficient hashing function. The work can be extended by studying how to update them jointly. It is also justifiable to apply this method for the retrieval of videos with the use of the video key frames as the input.

2.6.3 Semi-Supervised composite Multi-view Discrete Hash (SSMDH) Model

At the time where the majority of researches on the subject of hash models have been focused on single-view data, lately the multiview methods with a majority of unsupervised multi-view hashing models were taken under consideration.

In addition to the incorporation of a part of label data in the model, the suggested multi-view model that has been designed by Wei-Shi Zheng and C. Zhang [84] is different from the available multi-view hashing models in 3-fold: 1) a composite discrete hashing learning modelling which is capable of jointly minimizing the loss on multi-view characteristics in the case of utilizing relaxation on learning hash codes, 2) a composite locality preserving modelling for locally compact coding, 3) the exploration of statistically uncorrelated multi-view characteristics for the generation of hashing codes.

A future extension of this work can be investigating if an entirely end-to-end multi-view semi-supervised architecture would get considerably more enhancement.

2.6.4 Semi-Supervised Deep Hashing (SSDH) Method

This approach has been presented by Jian Zhang and Yuxin Peng [85] for performing better hash function learning via a simultaneous preservation of the semantic similarity and underlying data structures. The basic contributions were the following: 1) a semi-supervised loss for jointly minimizing the empirical error on labeled data, in addition to the embedding error on each of the unlabeled and labeled data that might maintain the semantic similarity and record the meaningful neighbors on the underlying data structures for sufficient hashing. 2) a SSDH network has been proposed for broadly exploiting each of the labeled and unlabeled data.

The suggested deep NN performed learning of hashing code and learning of features in a simultaneous manner in a semi-supervised manner. As a future work, it is possible to discover a variety of semi-supervised embedding methods which might take more advantage of the unlabeled data, and more improved strategies of

graph construction may be used. Indeed, this framework can be extended to an unsupervised scenario, in which clustering methods are used for obtaining virtual image labels.

2.6.5 The Approach of Semi-Supervised Metric Learning-based Anchor Graph Hashing (MLAGH)

Another approach was presented by Haifeng Hu *et al.* [86]. This method may be split to 3 parts. 1) utilizing a transform matrix for the construction of the anchor-based graph of similarity of the training dataset. 2) proposing the objective function which is based on the triplet correlation, where the best transform matrix may be learnt with the use of the label smoothness and the margin hinge loss which is incurred by the triplet constraint.

In addition to that, the approach of stochastic gradient descent (SGD) can leverage the gradient on every one of the triplets for updating the transform matrix. 3) designing a penalty factor for accelerating the speed of execution for the SGD.

2.6.6 Comparison of the Semi-Supervised Methods

Comparing the semi-supervised learning based IR methods mentioned in the present section, a preference can be assigned to the second method (SKHL) [83]in general broad terms. This can be for the following reasons: First, every one of the hashing functions is modeled in a form of a nonlinear kernel hyper-plane that has been constructed from an unlabeled dataset. By the maximization of a Fisher-like criterion on a weakly labeled dataset only with side information, one can get a collection of hashing functions and optimal kernel hyper-planes.

In addition to that, every one of the hashing functions is independently updated in every one of the iterations. This approach is applicable as well for the retrieval of videos with the use of video key frames as the input.

2.7 Unsupervised Learning Techniques

Unsupervised approaches detect internal correlations in the unlabeled input data. One of the main unsupervised approaches is the clustering. Even though clustering requires no class labels, classifiers may be derived in the case where the

traffic clusters are corresponding to various applications of the network [1]. In this section, some of the most interesting unsupervised learning proposals are considered.

2.7.1 Cascaded Principal Component Analysis (PCA)

This architecture was proposed by Tsung-Han *et al*. The main segment is cascaded principal component analysis (PCA), the second one is binary hashing, and the last one is block-wise histograms. For comparing and to present a good understanding, the paper also introduced and studied two simple PCA-Net variations, which are: Rand-Net and LDA-Net [87].

Experimentations on other public datasets also demonstrated the possibility of PCA-Net in serving as a simple yet very competitive base-line for classification of textures and recognition of objects. As soon as the parameters get fixed, the training of the PCA-Net is very simple and sufficient due to the fact that the filter learning in the PCA-Net involves no regularized parameters or requires numerical optimization solvers.

In addition to that, the construction of the PCA-Net includes only a cascaded linear map, which is followed by a non-linear output phase. This level of simplicity presents an alternative, but at the same time refreshing potential on CNNs and might additionally simplify the mathematical justification and analysis of their efficiency. The bottleneck which might prevent PCA-Net from getting deeper (for instance to more than two stages) is the fact that the resultant feature dimension would exponentially maximize with the number of stages. Which can probably be fixed via the replacement of 2-D convolutional filters with tensor-like one as a future work.

2.7.2 Quaternion Orthogonal Matching Pursuit (Q-OMP)

V. Risojevic and Z. Babic [88] proposed unsupervised learning of quaternion feature filters using quaternion representation for color images, in addition to feature encoding with the use of a Q-OMP. With the use of quaternion representation, there was a possibility of jointly encoding color information and intensity in an image.

Local descriptors have been obtained with the use of soft thresholding and the calculation of the absolute values of scalar and 3 vector parts of the quaternion valued distributed code. Local descriptors have been pooled, normalized, and power-law transformed, thereby resulting in the output image representation.

The suggested approach for quaternion feature learning has been capable of adapting to the properties of the available data, and being entirely unsupervised It has appeared as a suitable substitute to each of convolutional NNs and hand-crafted representations, particularly in application scenarios that include scarce-labeled training data. This approach might as well be extended to hierarchical classification structure.

2.7.3 Siamese-Twin Random Projection Neural Network (ST-RPNN)

Mohamed Fahkr *et al.* designed ST-RPNN approach that comprises two similar random projection NNs with hard thresholding neurons in which the binary code is fed as the results for the neuron. The learning goal was producing similar binary codes for similar pairs of input image and dissimilar binary codes in the opposite case [89].

The procedure of learning has been split to 2 stages. Initially, over-complete random projection has been utilized for producing a suitably long code, which has been followed by a fast sparse technique for neurons selection (FSNS). Bootstrap Aggregation Trees or Bagging Trees (BT) has been after that, utilized for making an enhanced compact code section. BT has as well been utilized as a fast retrieval tool which was used for ranking the database in terms of a query with no calculations of distance and with a considerably lower level of complexity compared to the method of Hamming distance.

2.7.4 Converting Unsupervised Hashing to Supervised Hashing using Pseudo Labels of Images

Haofeng Zhang *et al.* proposed an unsupervised model which had two main contributions: The first one is converting the unsupervised DH architecture to supervised via learning pseudo labels and the second one is the framework unifies probability maximizing, quantization error minimizing, and mutual information maximizing, in a way that the pseudo labels may maximally be preserved [90].

2.7.5 Comparison of Unsupervised Methods

Comparing the unsupervised learning based IR methods mentioned in this section, the first method [87] seems to be the best one in general terms. It is the simplest unsupervised convolutional DL network that called cascaded principal component analysis due to the fact that the filter learning in the PCA-Net involves no regularized parameters or requires numerical enhancement solvers.

In addition to that, constructing the PCA-Net comprises only a cascaded linear map, which is followed by a non-linear output phase. Also, this architecture exceptionally straightforward DL network for classifying images which depends on extremely basic data processing parts.

2.8 Summary

In this chapter the most common techniques of IR are explained and classified into three categories based on the learning method: Supervised, Semi-Supervised and Unsupervised techniques. The direction of thesis is with the first category (Supervised Learning Techniques). Because it uses the labeled data for training models of classification.

The methods explained in section (2.5.1 above) are the significant work of this approach, In which IM and Sequential Model are used in images classification, where the labeled data represent the output of IM which is a pretrained model works as features extraction. This pretrained model trained on ImageNet dataset that contains 1000 category with 1.2 million images.

The sequential model represents the classification part of the whole architecture which consists of fully connected layers and learned using TRVS of IM to classify new images in another time.

Chapter Three

Proposed Images Classification System

Chapter Three: The Proposed Image Classification System

3.1 Introduction

This thesis focuses on the design and implementation of DL-based image classification system with high performance, where CIFAR-10 dataset is used to train the models. The IM is the beginning of the classification work and the main aim of this model is converting dataset to TRVs that are suitable for training and testing classification models. The original classification part of IM is removed and the output of IM in this case is the TRVs of CIFAR-10 images. These TRVs are the input of the classification part (Fully connected layers) which has ten categories (number of categories in CIFAR-10 dataset) and the output is what the model predicts of any input image. The core of this work is the use of PSO to reduce TRVs size and building new models compatible with this size. Later on, when there is a new image to classify, it would be possible to use the same template of reducing TRVS to modify the new image. PSO is used in two cases: The first case is returning the minimum TRVS and the second one is returning the template of TRVS with high accuracy in addition to reducing TRVS. The crossover technique is applied inside PSO for getting new solutions some of them have high accuracy with high number of TRVs and others have lower accuracy with lower number of TRVs, where the selection of model based on the type of the application.

This chapter contains the following: explaining CIFAR-10 dataset and reasons of using it. Followed by the architecture of CNN. Also clarify the main parts of IM and how the TRVs have been resulted. After that explaining the method of building classifier part. Finally explaining the enhanced PSO with the whole architecture of classifying. These steps are shown in Figure 3.1.

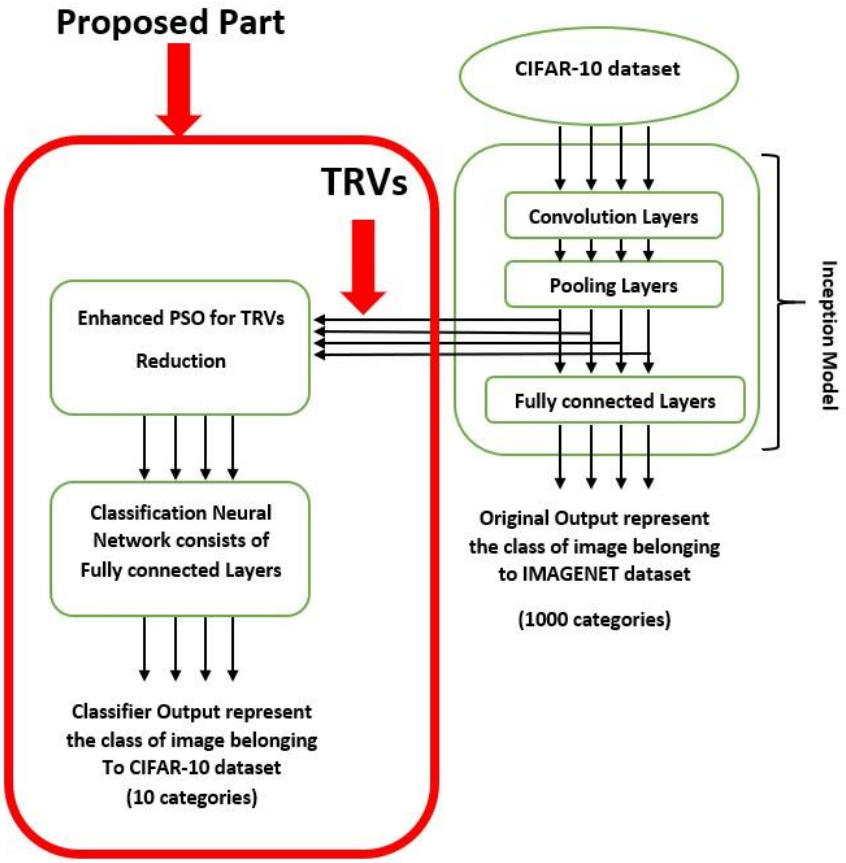

Figure 3.1: Main Architecture of the Proposed Image Classifier.

In the Figure 3.1 there are two parts of image classifier. The first part is TRVs extraction using IM and the second part is the proposed approach that use these TRVs and reducing it using enhanced PSO to build new classification model suitable for new dataset.

3.2 CIFAR-10 Dataset

The CIFAR-10 dataset is made up of 60000 color images, each image has dimensions of 32x32 pixel. The dataset is divided into ten classes: **Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship and Truck**. Each class contains 6000 images. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some

training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class. The reason of using CIFAR-10 dataset is this dataset has a small number of images, also the size of images is too small(32*32) pixel.

3.3 CNN

To clarify how CNN deals with the image, here is an example, When using a face image as input into a CNN, the network will learn some basic characteristics such as edges, bright spots, dark spots, shapes etc., in its initial layers. The next set of layers will consist of shapes and objects relating to the image which are recognizable such as: eyes, nose and mouth. The subsequent layer consists of aspects that look like actual faces, in other words, shapes and objects which the network can use to define a human face. CNN matches parts rather than the whole image, therefore breaking the image classification process down into smaller parts (features).

A 3x3 grid is defined to represent the features extraction by the CNN for evaluation. The following process, known as filtering, involves lining the feature with the image patch. One-by-one, each pixel is multiplied by the corresponding feature pixel and once completed, all the values are summed and divided by the total number of pixels in the feature space. The final value for the feature is then placed into the feature patch. This process is repeated for the remaining feature patches followed by trying every possible match- repeated application of this filter, which is known as a convolution.

The next layer of a CNN is referred to as “max pooling”, which involves shrinking the image stack. In order to pool an image, the window size must be defined, the stride must also be defined. The window is then filtered across the image in strides, with the max value being recorded for each window. Max pooling reduces the dimensionality of each feature map whilst retaining the most important information.

The normalization layer of a CNN, also referred to as the process of ReLU, involves changing all negative values within the filtered image to zero. This step is then repeated on all the filtered images, the ReLU layer increases the non-linear

properties of the model. The subsequent step by the CNN is to stack the layers (convolution, pooling, ReLU), so that the output of one layer becomes the input of the next. Layers can be repeated resulting in a “deep stacking”.

The final layer within the CNN architecture is called the fully connected layer also known as the classifier. Within this layer every value gets a vote on determining the image classification. Fully connected layers are often stacked together, with each intermediate layer voting on phantom “hidden” categories. The final layer of these layers called soft-max layer that give the classification category. In effect, each additional layer allows the network to learn even more sophisticated combinations of features towards better decision making.

The values used for the convolution layer as well as the weights for the fully connected layers are obtained through backpropagation, which is done by the deep neural network. Backpropagation is whereby the neural network uses the error in the final answer to determine how much the network adjusts and changes.

3.4 Transfer Learning

The pre-trained IM is used for classifying images. It is actually capable of extracting useful information from an image. So, it can be instead train the Inception model using another dataset. But it takes several weeks using a very powerful and expensive computer to fully train the IM on a new dataset. It can be instead re-use the pre-trained inception model and merely replace the layer that does the final classification. This is called Transfer Learning.

3.5 Inception Model

The Inception-v3 model is an architecture of convolutional networks. The use of IM v3 because the dataset is small and no need for more processing. Also this model has more accurate than v2 and v1. It is one of the most accurate models in its field for image classification having been trained on the ImageNet dataset. Originally created by the Google Brain team, this model has been used for different tasks such as object detection as well as other domains through Transfer Learning as shown in Algorithm 3.1.

The Inception v3 model takes weeks to train on a monster computer with eight Tesla K40 GPUs and probably costing $30,000 so it is impossible to train it on an ordinary PC. We will instead download the pre-trained IM and use it to classify images. This model has nearly 25 million parameters and uses five billion multiply-add operations for classifying a single image. On a modern PC without a GPU this can be done in a fraction of a second per image.

Algorithm 3.1: Steps of IM.

```
Input: CIFAR-10 dataset

Output: Transfer Values of CIFAR-10 dataset

Step1: Define images input shape entering to IM

Step2: define the number of Convolution Layers

Step3: if using IM itself to classify image then:
         a- define the number of fully connected Layers and size of
            each layer
         b- define the size of soft-max layer (equal to number of
            categories in the dataset).

Step4: define TRVs array

Step5: For I=1 to number of convolution layers
       Execute Steps 6 to 8

Step6: execute three convolution operations on the input image:
       A=Convolution (1*1), B= Convolution (3*3), C= Convolution
       (5*5)

Step7: execute two max polling operations on B and C:
       B1=max polling(B)
       C1= max polling(C)

Step8: TRVs=concatenate (TRVs, A, B1, C1)

Step9: return (TRVs)
```

In Algorithm 3.1 above, the input to this algorithm is the image and the output is the TRVs of this image. The first step in this algorithm is defining the input shape of the image to be compatible with the IM. After that, there are two cases to use IM: the first case is classifying images using original fully connected layers of the IM, this can be done by sending the image to the IM and IM will return the class of this image. The second case is using IM to generate TRVs of a given image, these values later will use to build and train new classifier of a given dataset.

Also, there are three types of convolution layers:(1*1), (3*3) and (5*5). The convolution layers (3*3) and (5*5) enter to the max-polling to select the best values in the layer. Then linking (1*1) convolution layer with max-polling layer of (3*3) and (5*5) layers to generate the TRVs of the image.

3.6 Generating TRVs

The first step of creating image classifier is to generate TRVs of CIFAR-10 dataset using IM v3 as shown in **Figure 3.2** below:

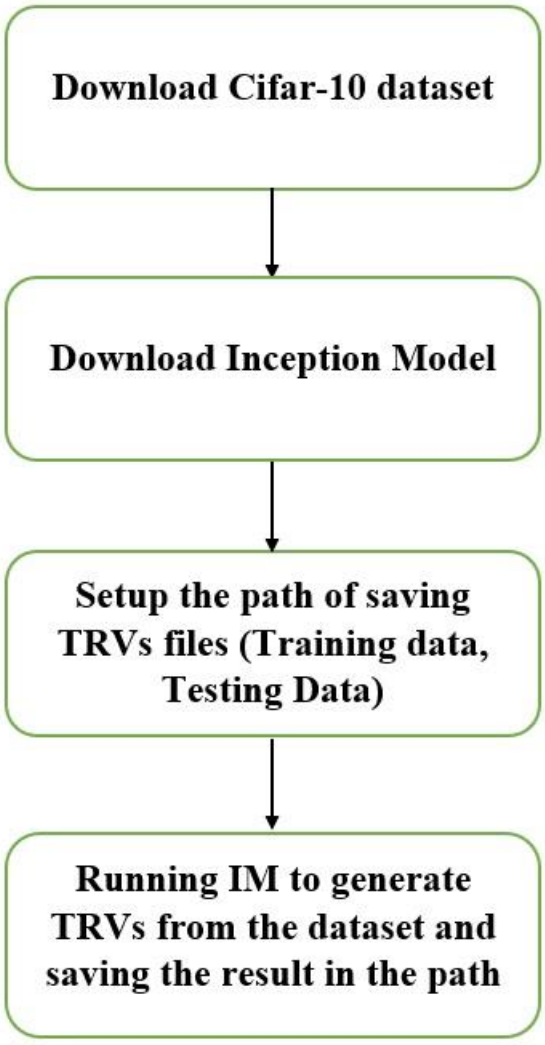

Figure 3.2: Generating TRVs of CIFAR-10 Dataset Using IM.

3.7 Building Classifier Part

The input to this part is TRVs of reduced image and the output is the result of classifying image, where this part consists of a set of fully connected layers and soft-max layer that are trained and tested using the TRVs. These layers are used for final classification of new images as shown in **Algorithm 3.2.** The output length of soft-max layer consists of ten values which is equal to the number of CIFAR-10 categories and the sequence of the soft-max output value represent the sequence of the category in CIFAR-10 categories from zero to nine. For example: if the soft-max value has a sequence zero then the class of the input image is Airplane and so on**.**

Algorithm 3.2: Building Classification Part using TRVs as input to Fully connected layers.

```
Input: TRVs of CIFAR-10 dataset
Output: The accuracy of the built model

Step1: define a new sequential model.
Step2: define the size of the input layer (at the beginning using
       full TRVS of IM-2048 TRVS-)
Step3: define the dimension of each layer say(32-128).
Step4: create the layers using dimensions in step3 and each lower
       layer dimension should be less than or equal to the dimension
       of the layer above it.
Step5: define the dimension of output layer (equal to 10 that
       represent the number of categories CIFAR-10 dataset.
Step6: define the optimizer used for reducing loss.
Step7: train the model and validate it using training and testing
       values of CIFAR-10 dataset.
Step8: evaluate the accuracy of the model.
```

In Algorithm 3.2, the input is TRVs of CIFAR-10 dataset and the output is the classifier model with its accuracy. Where this algorithm is used to test the best configuration of neural in term of number of layers and parameters, beginning with four layers and ending with one layer. The maximum dimension of layer is 2048 nodes and the minimum one is 16 nodes. The best architecture was obtained with

two layers, where the first layer contains 128 nodes and the second layer contains 32 nodes as will justify in Chapter four.

3.8 Particle Swarm Optimization (PSO) with Enhancement

In this thesis, two cases are used. The first case is select the solution with minimum TRVs to train and test model. The second case is select the solution maximum accuracy. PSO represented as a function to reply the solution of best fitness, where the input to this function is a swarm. The swarm is a set of solutions each one has random values of zeros and ones, where the value of one represents the existence of the corresponding transfer value in the dataset or in the new image, while the value of zero refers to the transfer values that will be excluded from the dataset and new image as shown in **Figure 3.3**. The algorithm below describes the steps of PSO.

Algorithm 3.3: Steps of PSO algorithm with crossover Technique.

```
Input: TRVs template.

Output: best template of TRVs.

Step1: generate random templates.

Step2: Update velocity and position for each solution

Step3: evaluate fitness for each solution

Step4: select best solution according to the value of fitness.

Step5: make the best solution the leader of the swarm.

Step6: set the value of iterations

Step7: For I=1 to the number of iterations
       Execute Steps (8 to 11) for each solution (Particle)
Step8: update velocity, position
Step9: calculate fitness for the particle
```

```
Step10: if the fitness of particle is better than the fitness of
        the leader then the leader = particle

Step11: Running crossover function between the bad half and good
half after sorting solutions from worst to best to get new
solutions

Step12: return the leader (Best solution in the swarm)
```

In **Algorithm 3.3**, the input to this algorithm is a set of solutions each one represents the template of reducing the dataset and the image. The number of solutions used are 40 solutions because increasing of solutions need more time for running, also the number of features in the data set is small so there is no need for using high number of solutions. Now each solution has length of 2048 values generated randomly using zeros and ones which represent the position of the solution. The velocity and position can be calculated as mentioned in section 2.4.

Now there is a template coming from PSO which has length of 2048 values (0 or 1). This template will be used in two steps: the first step is to reduce the dataset according to modify function, where the input to this function is the template from PSO and original dataset (in case of building model) or (image in case of classify new image), the output is a new dataset which is less than original dataset or image data which is less than the original image data. Then building model that has input shape equal to the size on new dataset. The second used is when there is a new image to classify, it should reduce the values of image according to the same template, so the image can be classified using this model, as illustrated in Figure 3.3 and Figure 3.4

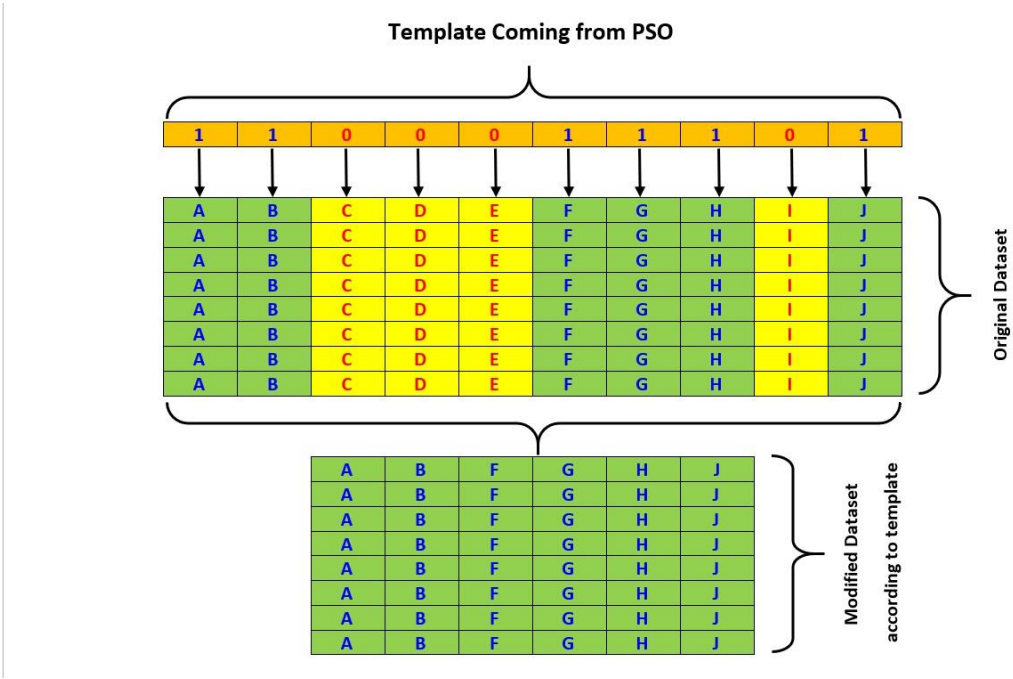

Figure 3.3: Architecture of Reducing Dataset According to PSO Template.

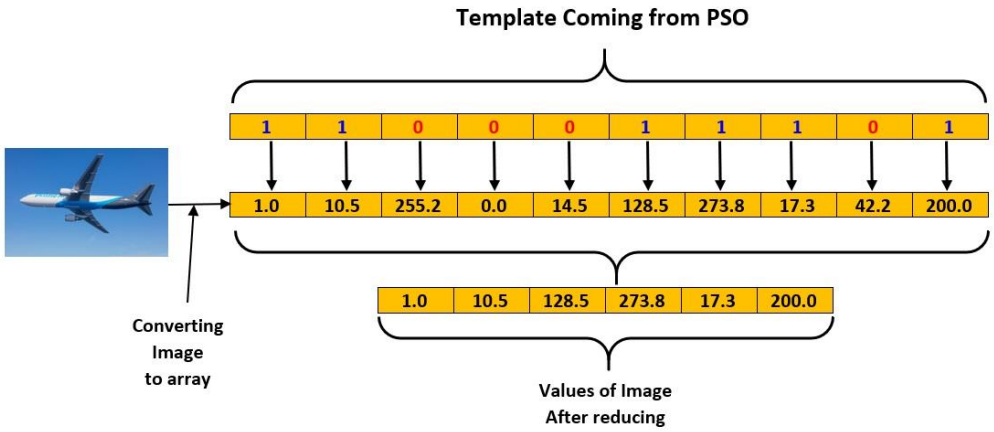

Figure 3.4: Reducing the Values of Image Using the Template of PSO.

PSO algorithm can be enhanced using the crossover technique and applying it on the swarm at the ending of each iteration in Algorithm 3.3, Two steps were added in order to achieve this enhancement. The first step is sorting the solutions in an ascending order from worst to best according to the value of model accuracy and the second step is performing crossover operation on the swarm resulted from the step1 as shown in sections 3.8.1 and 3.8.2.

PSO algorithm can be enhanced using the crossover technique and applying it on the swarm at the ending of each iteration in Algorithm 3.3: Steps of PSO algorithm with crossover Technique.

3.8.1 Sorting Solutions from Worst to Best

The swarm resulted from each iteration in (Algorithm 3.3: Steps of PSO algorithm with crossover Technique), will be sent to another function called sorting function. This function will arrange the solutions inside the swarm in ascending order based on the value of accuracy for each solution, then the crossover operation will be applied on the sorted swarm.

3.8.2 Performing Crossover Operation

The crossover technique is a one of genetic algorithm techniques, where the goal of this technique is to obtain good solutions from bad solutions by overlapping bad solutions with good solutions. There are several types of crossover: single-point crossover, two-point crossover and etc. In this work, single-point cross over was used between the solutions as shown in Figure 3.5.

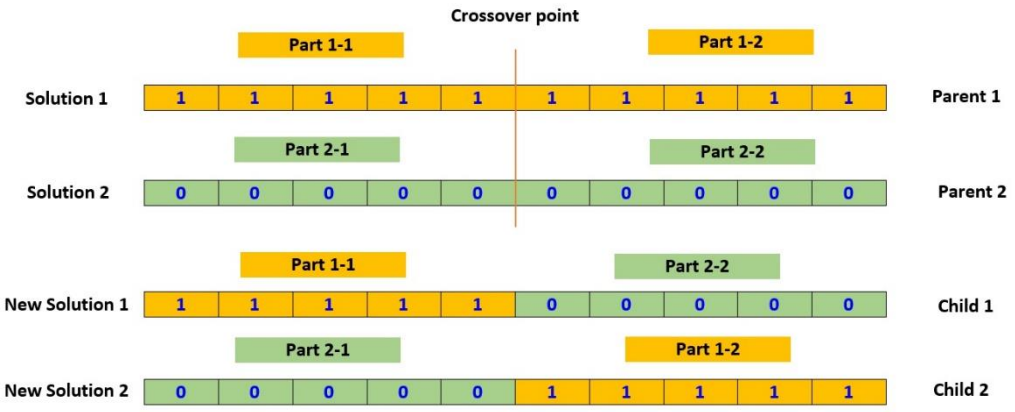

Figure 3.5: Crossover Operation.

3.9 Approach of Creating Images Classifier and Classifying new Images

The approach of creating image classifier is made up of two parts: Part one includes the steps of creating classification model, while part two is consists of the steps of classifying new image that enter to the system as shown in **Figure 3.6**.

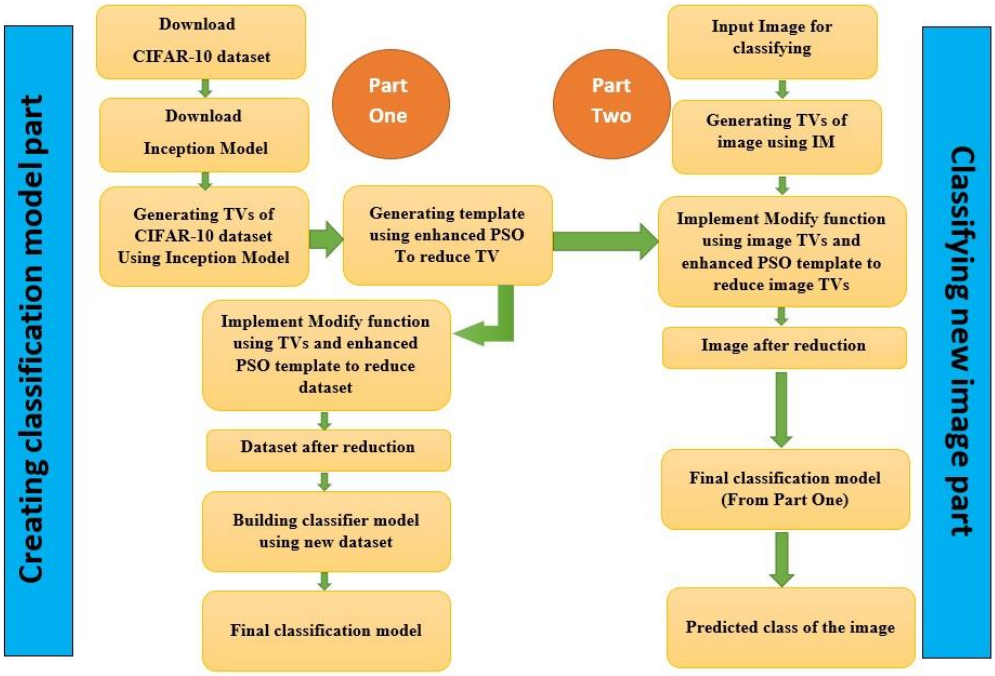

Figure 3.6: Approach of Creating Image Classifier and Classifying New Image.

As shown in the Figure 3.6, the part one on the left side contains the following: downloading CIFAR-10 dataset and IM, where IM is used for extracting TRVs of the dataset. From the other hand the enhanced PSO is working to generate the template of TRVs reduction. Now the modify function is working for generating reduced dataset to build new model that has input shape equal to the size of the reduced dataset. The final step in part one is to build classification model by calling the build function with two parameters: the first one is reduced dataset and the second one is the template.

After building the classification model, now it is possible to classify a new image by implementing the steps of part two as follows: there are two things

resulted from part one and will be necessary in part two, the first thing is the template and the second thing is the final classification model. To classify a new image, firstly the image is entering to part two, then this image is converting to TRVs using IM. The modify function is used to reduced TRVs using the template and original TRVs of image. The resulted TRVs (TRVs of image after reduction), are the input to the final classification model. The output of this model represents the predicted class of the image which belonging to CIFAR-10 categories.

3.10 Summary

After explaining all parts of classification architecture, now it is possible to summarize how the whole approach is working.

The first step is to generate TRVs as shown in **Figure 3.2** using **Algorithm 3.1**. Then before building classification model it has two cases to manipulate TRVs: the first case is using all TRVs (2048) to build classification model, this can be done by making all values in the template equal to one. The second case is using the template of enhanced PSO algorithm see **Algorithm 3.3**. The TRVs and the template are sent to modify function to generate new dataset from original dataset. Now the model can be built after preparing the data using **Algorithm 3.3**. The built model is now able to classify new images.

For classifying a new image, the image at first is converted to array, then reducing the data of this array using the same template used in building the classification model by sending image array and template to modify function. The output of modify function is the image after reducing its TRVs according to the template. Finally use the model to predict the class of this image.

Chapter Four
Results and Discussion

Chapter Four: Results and Discussion

4.1 Introduction

Image classification is one of the core problems in computer vision field with a large variety of practical applications. Examples include object recognition for robotic manipulation, pedestrian or obstacle detection, autonomous car driving, personal identification systems in airports, marketing and aircraft navigation systems among others. In this chapter, the proposed system is tested in order to obtain and discuss the results in order to indicate the effectiveness of this system. There are four parts of experiments were done using CIFAR-10 dataset and getting the results: the first part is the results of testing images classifier configuration, the second part represents the testing results of two layers (selected from the first part) on all possible number of nodes and selecting the best configuration, the third part represents the results of three experiments: the first one is testing the accuracy of image classifier using all TRVs to build the classifier, the second one is results of using minimum TRVs resulted from PSO and checking the accuracy of the image classifier and the third one is results of using maximum accuracy resulted from model built using TRVs coming from enhanced PSO. The final part of experiments represents using all models built in three experiments above to classify real images from CIFAR-10 dataset, images from google that has different categories other than that of CIFAR-10 dataset.

4.2 Configuration of Classifier Part Experiments

The configuration of fully connected layers in term of number of layers and nodes, was tested on several models beginning with four layers and ending with two layers. There is no using to one layer because one layer is not suitable for large datasets that have a big number of features. The maximum dimension of each layer is 2048 nodes and the minimum one is 16 nodes. In **Table 4.1**, each row in these tables represents the complete model configurations and results obtained from this model, while the columns of these tables represent the following: the first column is model no., which represent the sequence of the model in the experiment, the second column represents the number of fully connected layers in the model, which

is between one to four layers, while the third column represents the number of nodes in each layer and is between (16-2048). The fourth column contains the loss value of training, where this value equal to the difference between true label and predicted label of the image inside training, which should be minimize as possible.

The five column represents the accuracy of the model and it is equal to the number of true predicted values divide on the number of all images entering to the model while training multiply by 100. The last column represents the time consumed for training using the following parameters: batch size equal to 32, which represents the number of images that send to the model in each time.

Table 4.1: Settings and Results of Network Consists of Two - Four Layers.

Model No.	No. of Layers	No. of Nodes	Loss	Accuracy	Time (Second)
1.	2	2048-2048	0.3551	0.8992	326
2.	2	1024-1024	0.3512	0.8989	174
3.	2	512-512	0.3346	0.8965	119
4.	2	256-256	0.3488	0.892	62
5.	2	128-128	0.345	0.8934	52
6.	2	64-64	0.3541	0.8949	52
7.	2	32-32	0.3411	0.8917	50
8.	2	16-16	0.3747	0.8823	51
9.	2	2048-1024	0.3468	0.9026	288
10.	2	1024-512	0.3563	0.8957	189
11.	2	512-256	0.3478	0.894	114
12.	2	256-128	0.3461	0.894	62
13.	2	128-64	0.3671	0.8928	52
14.	2	64-32	0.3563	0.8927	52
15.	2	32-16	0.3478	0.8868	50
16.	3	2048-2048-2048	0.3403	0.8945	384
17.	3	1024-1024-1024	0.366	0.8953	164
18.	3	512-512-512	0.3629	0.8958	81
19.	3	256-256-256	0.3446	0.8916	61
20.	3	128-128-128	0.3395	0.8924	51
21.	3	64-64-64	0.3382	0.8886	51
22.	3	32-32-32	0.3438	0.8848	51
23.	3	16-16-16	0.3636	0.8875	51

24.	**3**	**2048-1024-512**	**0.3591**	**0.8963**	**299**
25.	**3**	**1024-512-256**	**0.3624**	**0.8937**	**194**
26.	**3**	**512-256-128**	**0.3495**	**0.8924**	**128**
27.	**3**	**256-128-64**	**0.3385**	**0.8943**	**79**
28.	**3**	**128-64-32**	**0.3486**	**0.8946**	**55**
29.	**3**	**64-32-16**	**0.3447**	**0.8929**	**57**
30.	**4**	**2048-2048-2048-2048**	**0.3488**	**0.8897**	**554**
31.	**4**	**1024-1024-1024-1024**	**0.357**	**0.8897**	**240**
32.	**4**	**512-512-512-512**	**0.3925**	**0.889**	**178**
33.	**4**	**256-256-256-256**	**0.3794**	**0.887**	**91**
34.	**4**	**128-128-128-128**	**0.3551**	**0.8919**	**69**
35.	**4**	**64-64-64-64**	**0.3249**	**0.9**	**65**
36.	**4**	**32-32-32-32**	**0.3597**	**0.8818**	**64**
37.	**4**	**16-16-16-16**	**0.3657**	**0.885**	**78**
38.	**4**	**2048-1024-512-256**	**0.3447**	**0.8929**	**321**
39.	**4**	**1024-512-256-128**	**0.3413**	**0.896**	**216**
40.	**4**	**512-256-128-64**	**0.3348**	**0.8948**	**158**
41.	**4**	**256-128-64-32**	**0.3658**	**0.892**	**97**
42.	**4**	**128-64-32-16**	**0.3418**	**0.8955**	**70**

In the **Table 4.1**, the selected architecture is the architecture that has two layers for several reasons: the best accuracy was in the two layers. Also, the time consumed to build neural with a smaller number of layers and smaller number of nodes is less than it to build neural with more layers and more nodes in each layer. Another reason is the complexity of the models is less with a smaller number of layers and nodes.

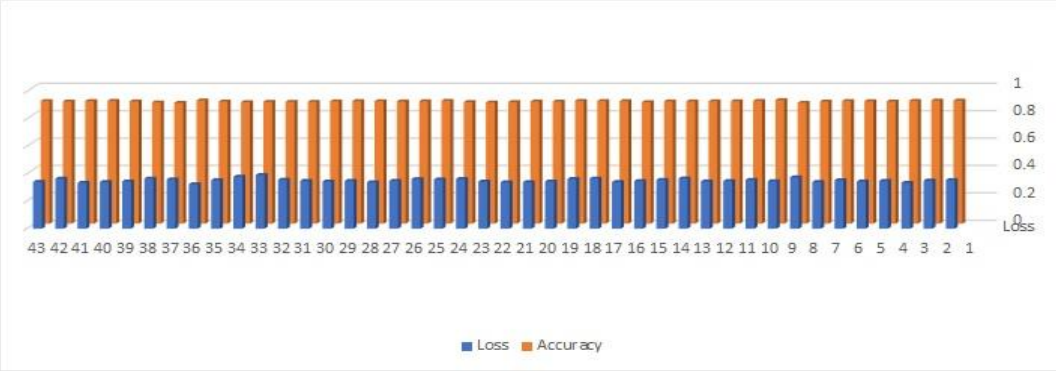

Figure 4.1: Relationship between Accuracy and Loss in Images Classifiers

4.3 Two Layers Experiments

After selecting the two layers setting from experiments illustrated in Section 4.2, this setting was further tested on all possible number of nodes using CIFAR10 dataset to train and test the models as shown below:

In the **Table 4.2**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 128 and 128 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.33787**, the accuracy average is equal to **0.89559** and the time average is equal to **51.1** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8922** and the best accuracy is equal to **0.9003**.

Table 4.2: Two Layers Model (128-128).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-128	2	Train 1	0.3427	0.8927	51	0.33787	0.89559	51.1
		Train 2	0.3408	0.8972	51			
		Train 3	0.3572	0.8927	51			
		Train 4	0.3262	0.8999	50	Worst Accuracy	Best Accuracy	
		Train 5	0.329	0.8932	51			
		Train 6	0.3248	0.9003	53			
		Train 7	0.3514	0.8942	51	0.8922	0.9003	
		Train 8	0.326	0.897	51			
		Train 9	0.3393	0.8922	51			
		Train 10	0.3413	0.8965	51			

In the Table 4.2, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the **Table 4.3**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 64 and 64 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.34016**, the accuracy average is equal to **0.8941** and the time average is equal to **51.4** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8881** and the best accuracy is equal to **0.8996**.

Table 4.3: Two Layers Model (64-64).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
64-64	2	Train 1	0.3428	0.8915	44	0.34016	0.8941	51.4
		Train 2	0.3397	0.8909	43			
		Train 3	0.3539	0.8881	52			
		Train 4	0.3395	0.8954	51	Worst Accuracy	Best Accuracy	
		Train 5	0.3395	0.8987	51			
		Train 6	0.3391	0.8953	52			
		Train 7	0.3337	0.8996	57	0.8881	0.8996	
		Train 8	0.3521	0.891	53			
		Train 9	0.3267	0.8984	55			
		Train 10	0.3346	0.8921	56			

In the Table 4.3, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the **Table 4.4**, the rows in this table shown the ten training rounds results of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 32 and 32 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each

model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.34189**, the accuracy average is equal to **0.89035** and the time average is equal to **50.4** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.885** and the best accuracy is equal to **0.895**.

Table 4.4: Two Layers Model (32-32).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
32-32	2	Train 1	0.3359	0.8878	43	0.34189	0.89035	50.4
		Train 2	0.35	0.8894	52			
		Train 3	0.3283	0.8937	51			
		Train 4	0.3505	0.885	49	Worst Accuracy	Best Accuracy	
		Train 5	0.3424	0.8905	50			
		Train 6	0.3526	0.8871	50			
		Train 7	0.3383	0.893	51	0.885	0.895	
		Train 8	0.3282	0.8927	52			
		Train 9	0.3348	0.895	55			
		Train 10	0.3579	0.8893	51			

In the **Table 4.4**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the **Table 4.5**, the rows in this table shows ten training rounds of the model. The columns of this table are as follow: the first column shows the number of nodes in each layer which are 16 and 16 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.39474**, the accuracy average is equal to **0.87457** and the time average is equal to **80.8** seconds. The last two

columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8565** and the best accuracy is equal to **0.8871**.

Table 4.5: Two Layers Model (16-16).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
16-16	2	Train 1	0.3565	0.8871	51	0.39474	0.87457	80.8
		Train 2	0.378	0.8813	131			
		Train 3	0.3563	0.8829	129			
		Train 4	0.3881	0.8771	56	Worst Accuracy	Best Accuracy	
		Train 5	0.3602	0.8796	51			
		Train 6	0.3892	0.8799	51			
		Train 7	0.4748	0.8571	61	0.8565	0.8871	
		Train 8	0.3819	0.8746	51			
		Train 9	0.4529	0.8565	55			
		Train 10	0.4095	0.8696	172			

In Table 4.5, it can be noticed that the accuracy is decreased comparing to the results in tables 4.2 – 4.4 and tables 4.6 – 4.11 of this experiments. The reason is that the size of the layers cannot cover the important features in the images, or the features are expanded on the size more than 16 nodes.

In the **Table 4.6**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 128 and 64 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.33716**, the accuracy average is equal to **0.89597** and the time average is equal to **51.1** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8923** and the best accuracy is equal to **0.9004**.

Table 4.6: Two Layers Model (128-64).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-64	2	Train 1	0.3275	0.8943	53	0.33716	0.89597	51.1
		Train 2	0.3377	0.8939	50			
		Train 3	0.3411	0.8976	51			
		Train 4	0.3373	0.8992	51	Worst Accuracy	Best Accuracy	
		Train 5	0.3595	0.8923	51			
		Train 6	0.3451	0.8934	51			
		Train 7	0.3385	0.8969	51	0.8923	0.9004	
		Train 8	0.3404	0.8981	51			
		Train 9	0.3234	0.8936	51			
		Train 10	0.3211	0.9004	51			

In the **Table 4.6**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the table 4.7, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 128 and 32 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to 0.33473, the accuracy average is equal to 0.89586 and the time average is equal to 60.1 seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to 0.8899 and the best accuracy is equal to 0.9012.

Table 4.7: Two Layers Model (128-32).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-32	2	Train 1	0.3464	0.891	52	0.33473	0.89586	60.1
		Train 2	0.3411	0.8938	64			
		Train 3	0.3378	0.8942	58			
		Train 4	0.3554	0.8899	55	Worst Accuracy	Best Accuracy	
		Train 5	0.3189	0.8984	56			
		Train 6	0.3288	0.8961	61			
		Train 7	0.3344	0.8978	52	0.8899	0.9012	
		Train 8	0.3169	0.9012	63			
		Train 9	0.3344	0.8996	68			
		Train 10	0.3332	0.8966	72			

In the table 4.7, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the Table 4.8, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 128 and 16 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.33007**, the accuracy average is equal to **0.89626** and the time average is equal to **53.9** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8916** and the best accuracy is equal to **0.9004**.

Table 4.8: Two Layers Model (128-16).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-16	2	Train 1	0.3286	0.8967	52	0.33007	0.89626	53.9
		Train 2	0.3549	0.8916	50			
		Train 3	0.3269	0.8986	53			
		Train 4	0.3308	0.8956	56	Worst Accuracy	Best Accuracy	
		Train 5	0.3231	0.8957	58			
		Train 6	0.3189	0.8985	57			
		Train 7	0.3171	0.9004	52	0.8916	0.9004	
		Train 8	0.3402	0.8949	56			
		Train 9	0.3324	0.8941	53			
		Train 10	0.3278	0.8965	52			

In the **Table 4.8**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the **Table 4.9**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 64 and 32 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.33503**, the accuracy average is equal to **0.89252** and the time average is equal to **61.1** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8844** and the best accuracy is equal to **0.8971**.

Table 4.9: Two Layers Model (64-32).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
64-32	2	Train 1	0.3383	0.8934	60	0.33503	0.89252	61.1
		Train 2	0.3292	0.895	51			
		Train 3	0.3231	0.8907	59			
		Train 4	0.3541	0.8877	53	Worst Accuracy	Best Accuracy	
		Train 5	0.3327	0.8955	62			
		Train 6	0.3342	0.8917	67			
		Train 7	0.3323	0.893	61	0.8844	0.8971	
		Train 8	0.3184	0.8967	67			
		Train 9	0.3247	0.8971	66			
		Train 10	0.3633	0.8844	65			

In the **Table 4.9**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the **Table 4.10**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 64 and 16 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.33056**, the accuracy average is equal to **0.89412** and the time average is equal to **67** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8913** and the best accuracy is equal to **0.8971**.

Table 4.10: Two Layers Model (64-16).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
64-16	2	Train 1	0.3191	0.8927	64	0.33056	0.89412	67
		Train 2	0.3288	0.8971	69			
		Train 3	0.3309	0.8948	69			
		Train 4	0.3325	0.8967	65	Worst Accuracy	Best Accuracy	
		Train 5	0.3379	0.8913	76			
		Train 6	0.3497	0.8918	71			
		Train 7	0.3197	0.8939	69	0.8913	0.8971	
		Train 8	0.3311	0.8939	59			
		Train 9	0.3249	0.8944	57			
		Train 10	0.331	0.8946	71			

In the **Table 4.10**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the table 4.11, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are 32 and 16 nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the sequence of each training round. The fourth, fifth and sixth columns represents the loss, accuracy and time of each model respectively (as mentioned in section 4.2). While the seventh, eighth and ninth columns represents the average of loss, accuracy and time for ten training rounds respectively, where the loss average is equal to **0.34298**, the accuracy average is equal to **0.88982** and the time average is equal to **69.6** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8832** and the best accuracy is equal to **0.8941**.

Table 4.11: Two Layers Model (32-16).

No. of Nodes	No. of Layers	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
32-16	2	Train 1	0.3618	0.8832	81	0.34298	0.88982	69.6
		Train 2	0.3356	0.8913	66			
		Train 3	0.3456	0.8908	67			
		Train 4	0.3478	0.8896	96	Worst Accuracy	Best Accuracy	
		Train 5	0.3499	0.89	84			
		Train 6	0.3464	0.8877	72			
		Train 7	0.3455	0.8907	69	0.8832	0.8941	
		Train 8	0.3322	0.8924	53			
		Train 9	0.3286	0.8941	54			
		Train 10	0.3364	0.8884	54			

It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In tables 4.2 - 4.11, it can be noticed that when the sizes of layers are less than 32 the average of the accuracy is decreased because this size cannot cover all possible features. On the other hand, when the TRVs are reduced, the complexity of the models will be less in term of total parameters and time.

The results showed that the best configurations in two layers was as follows: the first layer consists of 128 nodes and the second layer is made up of 32 nodes, because the model has highest accuracy using these configurations. So, there is no need to select the layers with high number of nodes because, selecting layers with a smaller number of nodes reduces the complexity of models in term number of parameters.

4.4 Applying PSO with two cases on TRVs Experiments

After testing all configuration of classifier part (Fully Connected Layers), it has been found that the best configuration is the two fully connected layers, where the first layer has 128 nodes and the second layer has 32 nodes. This configuration has been selected to build the proposed classifiers with less TRVs using PSO in two cases: The first case is PSO with minimum features and the second case is enhanced

PSO with maximum accuracy. Indeed, these cases have been compared with the model of using full TRVs.

In the **Table 4.12**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are **128** and **32** nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the number of TRVs used to build and train the images classifier, in this case using full TRVs (2048 values).

The fourth column is the total parameters used to build the model, in this case its equal to **266730** parameters. The fifth, sixth, seventh and eighth columns represents the sequence, loss, accuracy and time of the model respectively for each training round.

While the ninth, tenth and eleventh columns represents the average of loss, average of accuracy and average of time for ten training rounds respectively, where the loss average is equal to **0.33473**, the accuracy average is equal to **0.89586** and the time average is equal to **60.1** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8899** and the best accuracy is equal to **0.9012**.

Table 4.12: Two Layers Model (128-32) using Full TRVs.

No. of Nodes	No. of Layers	No. of TRVs	Total Parameters	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-32	2	2048	266730	Train 1	0.3464	0.891	52	0.33473	0.89586	60.1
				Train 2	0.3411	0.8938	64			
				Train 3	0.3378	0.8942	58			
				Train 4	0.3554	0.8899	55	Worst Accuracy	Best Accuracy	
				Train 5	0.3189	0.8984	56			
				Train 6	0.3288	0.8961	61			
				Train 7	0.3344	0.8978	52	0.8899	0.9012	
				Train 8	0.3169	0.9012	63			
				Train 9	0.3344	0.8996	68			
				Train 10	0.3332	0.8966	72			

In the **Table 4.12**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network.

In the **Table 4.13**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are **128** and **32** nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the number of TRVs used to build and train the images classifier, in this case using reduced TRVs according to the template coming from PSO with minimum TRVs. in this case (1030 values).

The fourth column is the total parameters used to build the model, in this case its equal to **136426** parameters. The fifth, sixth, seventh and eighth columns represents the sequence, loss, accuracy and time of the model respectively for each training round.

While the ninth, tenth and eleventh columns represents the average of loss, average of accuracy and average of time for ten training rounds respectively, where the loss average is equal to **0.33969**, the accuracy average is equal to **0.89234** and the time average is equal to **50.7** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8872** and the best accuracy is equal to **0.8957**.

Table 4.13: Two Layers Model (128-32) using PSO Template with Min TRVs.

No. of Nodes	No. of Layers	No. of TRVs	Total Parameters	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-32	2	1030	136426	Train 1	0.3462	0.8918	51	0.33969	0.89234	50.7
				Train 2	0.3568	0.8872	54			
				Train 3	0.329	0.8942	50			
				Train 4	0.3421	0.8946	56	Worst Accuracy	Best Accuracy	
				Train 5	0.3422	0.8914	53			
				Train 6	0.3399	0.8911	48			
				Train 7	0.3302	0.8957	46	0.8872	0.8957	
				Train 8	0.3392	0.8912	49			
				Train 9	0.3432	0.8912	50			
				Train 10	0.3281	0.895	50			

In the **Table 4.13**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network. Also, the total

parameters are decreased when the number of TRVs decreased. The reason is that when the TRVs decreased the neural need a smaller number of nodes which reduce the complexity.

In the **Table 4.14**, the rows in this table represents ten training rounds of the model. The columns of this table are as follow: the first column represents the number of nodes in each layer which are **128** and **32** nodes for the first and the second layer respectively. The second column contains the number of layers which are two layers. The third column represents the number of TRVs used to build and train the images classifier, in this case using reduced TRVs according to the template coming from enhanced PSO with maximum accuracy, in this case the TRVs is equal to 1383 values.

The fourth column is the total parameters used to build the model, in this case its equal to **181,610** parameters. The fifth, sixth, seventh and eighth columns represents the sequence, loss, accuracy and time of the model respectively for each training round.

While the ninth, tenth and eleventh columns represents the average of loss, average of accuracy and average of time for ten training rounds respectively, where the loss average is equal to **0.33692**, the accuracy average is equal to **0.89328** and the time average is equal to **55.2** seconds. The last two columns represent the worst and best accuracy for ten training rounds in which the worst accuracy is equal to **0.8864** and the best accuracy is equal to **0.8975**.

Table 4.14: Two Layers Model (128-32) Using Enhanced PSO Template with Max Accuracy.

No. of Nodes	No. of Layers	No. of TRVs	Total Parameters	Training No.	Loss	Accuracy	Time	Average Loss	Average Accuracy	Average Time (Second)
128-32	2	1383	181,610	Train 1	0.3452	0.8907	57	0.33692	0.89328	55.2
				Train 2	0.3322	0.896	53			
				Train 3	0.3368	0.8915	51			
				Train 4	0.3337	0.894	55	Worst Accuracy	Best Accuracy	
				Train 5	0.3333	0.896	56			
				Train 6	0.3307	0.8975	51			
				Train 7	0.3299	0.8942	59	0.8864	0.8975	
				Train 8	0.333	0.8962	55			
				Train 9	0.344	0.8903	56			
				Train 10	0.3504	0.8864	59			

In the **Table 4.14**, It can be seen that the values of accuracy and time are convergent and this indicates the presence of stability in the network. Also, the total parameters are decreased when the number of TRVs decreased. The reason is that when the TRVs decreased the neural need a smaller number of nodes which reduce the complexity.

The results in tables **4.12 - 4.14** showed that when reducing TRVs to build image classifier and classify new image the accuracy of the models is still high because the number of features used in the dataset is small comparing to number of TRVs, so reducing of TRVs will not affect the accuracy of models.

Also, the benefit of reducing TRVs is to minimize the complexity of processing in term of number of parameters used, number of TRVs used and time of training and classifying, where in the **Table 4.12** the accuracy average is **0.89586** which is very close to the accuracy average after reducing TRVs as shown in **Table 4.13** and **Table 4.14**, where in **Table 4.13** the accuracy average was **0.89234** and in **Table 4.14** the accuracy average was **0.89328.** Also, the number of parameters used, time average and TRVs used are less in **Table 4.13** and **Table 4.14** comparing with **Table 4.12**, so the performance of the models after reducing TRVs is higher than it using full TRVs in term time of training. Finally, the **Table 4.15** shows a comparison of the proposed approach with two approaches (SDPH, SDPH+) using MAP.

Table 4.15: Comparing the Proposed Approach with SDPH and SDPH+ Approaches Using MAP.

	Approach	**Accuracy using MAP**
1	SDPH [4]	0.8767
2	**The Proposed approach**	**0.89328**
3	SDPH+ [5]	0.9116

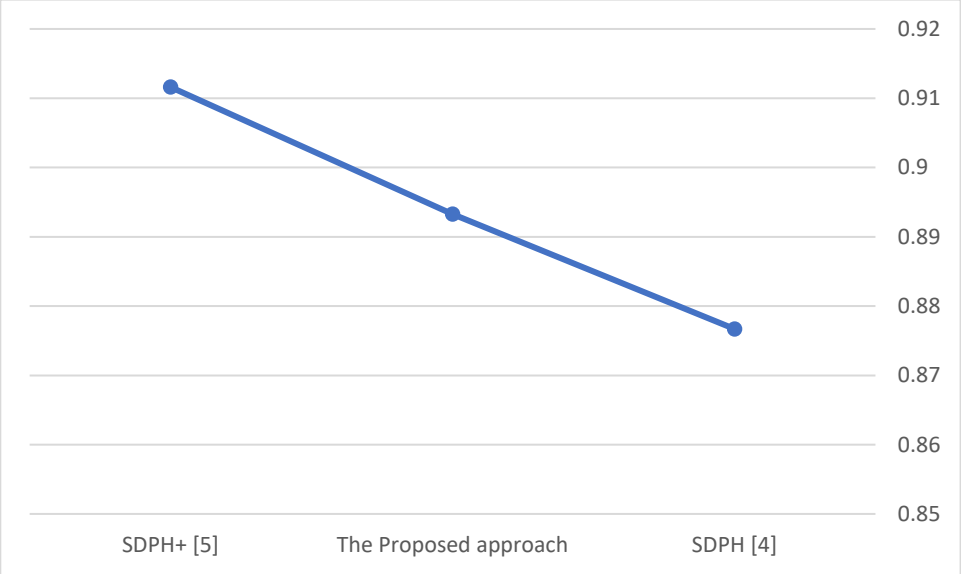

Figure 4.2: Comparing the Proposed Approach with SDPH and SDPH+ Approaches Using MAP.

4.5 Classifying Real Data Experiments

After completing all tests of the models, these models have been used to classify three types of images:

1- Selecting 100 images from the beginning of CIFAR-10 dataset itself.
2- Selecting 100 images downloaded from google that has same categories of the dataset.
3- Selecting 100 images downloaded from google that has different categories other than that of CIFAR-10.

The results of classifying images from CIFAR-10 dataset and Google are as shown in **Table 4.16**, where the rows of this table represent the three types of the tested models: model of using full TRVs, model of using reduced TRVs resulted from PSO with minimum TRVs and model of using reduced TRVs resulted from enhanced PSO with maximum accuracy, while the columns represent the following: two cases true and false predictions of 100 images selected from the beginning of CIFAR-10 dataset and 100 images downloaded from google.

Table 4.16: Classifying Real Images from CIFAR-10 Dataset and Google

Model Type	Images from CIFAR-10 Dataset		Images from Google	
	Correct Predication	Incorrect Predication	Correct Predication	Incorrect Predication
All TRVs used	93	7	98	2
PSO with Minimum TRVs	92	8	98	2
PSO with Maximum Accuracy	92	8	97	3

Table **4.16**, the ratio of true predictions of CFAR-10 images is less than it when classifying images from google because the resolution of CIFAR-10 images is very low comparing with google images that has high resolution. From the other hand, after reducing TRVs the true predictions still high because the feature of CIFAR-10 dataset is very little comparing to the number of TRVs.

The third case of this experiment is classifying 100 images downloaded from google that has different categories other than that of CIFAR-10. The model in some cases gives category from CIFAR-10 categories that is near to the type of the given image as shown Figure 4.3.

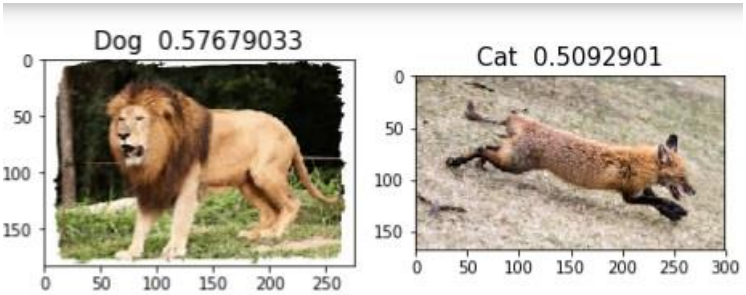

Figure 4.3: Sample Images with Succeed Predicted in Classification Model

In Figure 4.3, the model predicted that the image of lion as dog because it has the body that is nearest to dog in the original categories. Also, the fox image is predicted as cat because of the size and shape of fox is similar to cat.

From the other hand, the model sometimes fails to predict the image, so it gives random prediction to the image based on the maximum value of the neural network soft-max layer as shown in Figure 4.4.

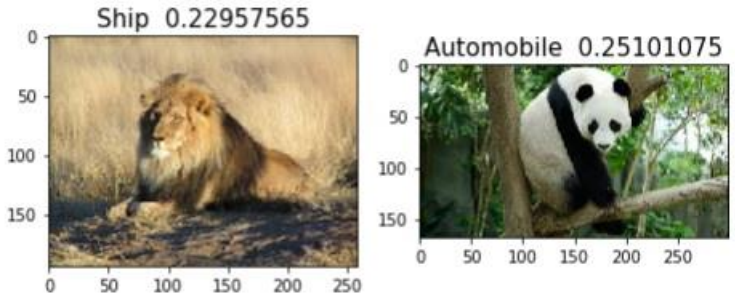

Figure 4.4: Sample Images with Fail Predicted in Classification Model

In figure Figure 4.4, the model fails to predict these images, because, there is no images that are close to these images in the dataset as shown in the prediction ratio, where it is too small, so there is no image near the given images.

Chapter Five
Conclusions and Future Works

Chapter Five: Conclusions and Future Works

5.1 Introduction

After designing image classifier that has high performance and making experiments for testing the results, there are some conclusions and future works that are mentioned in this chapter.

5.2 Conclusions

The most important conclusions of this thesis are:

1- Pre-trained models have high power in extracting TRVs because these models trained using big datasets and powerful computers and when extracting the data of dataset using normal CNN the accuracy is too low comparing to TRVs extracting using IM.

2- Reducing TRVs of dataset that has small number of features keeps the models of classifying with high accuracy as shown in Table 4.13 and Table 4.14 .

3- Using small number of TRVs makes classifier model has high performance in term of accuracy, number of used parameters and consumed time for training models.

4- Reducing TRVs before building the models increases the performance of building models and classifying new images because, it reduces the number of parameters used and time as shown in Table 4.12, Table 4.13 and Table 4.14.

5- Increasing number of fully connected layers and number of nodes in each layer lead to increasing the delay, without too much affecting the accuracy, in the models in term of training and classifying as explained in section 4.4 and section 4.5.

5.3 Future Works

There is a list of future works that can be applied in several directions. some of them are:

1- The developed system can be useful in application in class registration systems as well as being used as a biometric password. There is also the possibility to investigate different implementations of the system on a webserver. This would potentially allow users to use their face to verify their identity from any device and location in the world relating to the Internet of Things (IoT) as most devices are in one way or another connected online or moving towards.

2- There is still room to improve the accuracy results through implementation with the Inception-v4 model, increased epochs size and testing on larger datasets providing you have the time and resources.

3- Another possibility would be combining CNN with Long-Short Term Memory (LSTM). This may be multifaceted, but in theory should help achieving better results and efficiency.

4- Adding preprocessing step that remove the background of the picture and extracting the features of object only that make the accuracy of classifying better.

References

References

[1] G. Vargas-Solar, J. A. Espinosa-Oviedo, and J. L. Zechinelli-Martini, *Big continuous data: Dealing with velocity by composing event streams*. 2016.

[2] P. Tino, L. Benuskova, and A. Sperduti, "Artificial Neural Network Models," vol. 8, no. 3, pp. 455–472, 1997.

[3] Q. Li, Z. Sun, R. He, and T. Tan, "Deep Supervised Discrete Hashing," no. Nips, 2017.

[4] D. Yang, H. Xie, J. Yin, Y. Liu, and C. Yan, "Supervised deep quantization for efficient image search," *2017 IEEE Int. Conf. Multimed. Expo Work. ICMEW 2017*, no. July, pp. 525–530, 2017.

[5] C. Yan, H. Xie, D. Yang, J. Yin, Y. Zhang, and Q. Dai, "Supervised Hash Coding with Deep Neural Network for Environment Perception of Intelligent Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 284–295, 2018.

[6] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 3270–3278, 2015.

[7] N. Aslam, "Limitation and Challenges: Image/Video Search & Retrieval," *Int. J. Digit. Content Technol. its Appl.*, vol. 3, no. 1, 2009.

[8] M. Turcanik and M. Javurek, "Hash function generation by neural network," *NTSP 2016 - Proc. Int. Conf. New Trends Signal Process.*, no. December 2017, 2016.

[9] T. Komal, R. Ashutosh, R. Roshan, and A. M. Nalawade, "Encryption and Decryption using Artificial Neural Network," *Int. Adv. Res. J. Sci. Eng. Technol.*, vol. 2, no. 4, pp. 2393–2395, 2015.

[10] "Artificial Neural Networks (ANN) and their Types." [Online]. Available: https://www.elprocus.com/artificial-neural-networks-ann-and-their-types/. [Accessed: 20-Mar-2019].

[11] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 4, pp. 1773–1795, 2013.

[12] Z. Zha and T. Chua, "Oracle in Image Search : A Content-Based Approach to Performance Prediction," vol. 30, no. 2, pp. 1–23, 2012.

[13] H. Xie, Y. Zhang, J. Tan, L. Guo, and J. Li, "Contextual query expansion for image retrieval," *IEEE Trans. Multimed.*, vol. 16, no. 4, pp. 1104–1114, 2014.

[14] H. Jegou, M. Douze, and C. Schmid, "Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search – Extended version – ," *Eur. Conf. Comput. Vis.*, pp. 304–317, 2008.

[15] J. Wang *et al.*, "Learning fine-grained image similarity with deep ranking," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1386–1393, 2014.

[16] J. Wang, H. T. Shen, J. Song, and J. Ji, “Hashing for Similarity Search: A Survey,” pp. 1–29, 2014.

[17] Q. Wang, G. Zhu, and Y. Yuan, “Statistical quantization for similarity search,” *Comput. Vis. Image Underst.*, vol. 124, pp. 22–30, 2014.

[18] J. Fang, H. Xu, Q. Wang, and T. Wu, “Online hash tracking with spatio-temporal saliency auxiliary,” *Comput. Vis. Image Underst.*, vol. 160, pp. 57–72, 2017.

[19] A. Gionis, P. Indyk, and R. Motwani, “Similarity Search in High Dimensions via Hashing,” *25th Int. Conf. Very Large Data Bases*, vol. 99, no. 1, pp. 518–529, 1999.

[20] A. Andoni and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” *Proc. - Annu. IEEE Symp. Found. Comput. Sci. FOCS*, vol. 51, no. 1, pp. 459–468, 2006.

[21] A. Z. Ondrej Chum, James Philbin, “Near duplicate image detection: Min-Hash and tf-idf weighting,” *Aircr. Eng. Aerosp. Technol.*, vol. 14, no. 6, pp. 169–178, 1942.

[22] P. Jain, B. Kulis, and K. Grauman, “Fast image search for learned metrics,” *26th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR*, no. June, 2008.

[23] M. Raginsky and S. Lazebnik, “Locality-sensitive Bbinary Codes from Shift-invariant Kernels,” *Conf. Neural Inf. Process. Syst.*, pp. 1509–1519, 2009.

[24] Y. Weiss, A. Torralba, and R. Fergus, “Spectral Hashing,” *Nips*, vol. 21, no. 1, pp. 1753–1760, 2009.

[25] J. Wang, S. Kumar, and S.-F. Chang, “Sequential Projection Learning for Hashing with Compact Codes,” *Proc. 27th Int. Conf. Mach. Learn.*, pp. 1127–1134, 2010.

[26] Y. Gong, S. Lazebnik, C. Science, and U. N. C. C. Hill, “Iterative Quantization : A Procrustean Approach to Learning Binary Codes,” *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.(CVPR)*, pp. 817–824, 2011.

[27] H. T. S. Shen, Fumin, Chunhua Shen, Wei Liu, “Supervised Discrete Hashing,” *Comput. Vis. Pattern Recognit.*, pp. 37–45, 2015.

[28] W.-C. Kang, W. Li, and Z. Zhou, “Column Sampling based Discrete Supervised Hashing,” *Aaai*, pp. 1230–1236, 2016.

[29] A. T. AUDE OLIVA, “Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope,” *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, 2014.

[30] N. D. and B. Triggs, “Histogram of oriented gradients for human detection in video,” *Proc. 2018 5th Int. Conf. Bus. Ind. Res. Smart Technol. Next Gener. Information, Eng. Bus. Soc. Sci. ICBIR 2018*, pp. 172–176, 2018.

[31] H. Xie *et al.*, “Robust common visual pattern discovery using graph matching,” *J. Vis. Commun. Image Represent.*, vol. 24, no. 5, pp. 635–646, 2013.

[32] S. A.W.M., W. M., S. S., G. A., and J. R., “Content-based image retrieval at the end of the early years,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1349–1380, 2000.

[33] J. E. Camargo, J. C. Caicedo, and F. A. Gonzalez, “A kernel-based framework for image collection exploration,” *J. Vis. Lang. Comput.*, vol. 24, no. 1, pp. 53–67, 2013.

[34] L. Nie, M. Wang, L. Zhang, S. Yan, B. Zhang, and T. S. Chua, “Disease Inference from Health-Related Questions via Sparse Deep Learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 8, pp. 2107–2119, 2015.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.

[36] M. Lin, Q. Chen, and S. Yan, “Network In Network,” pp. 1–10, 2013.

[37] and P. V. Y. Bengio, A. Courville, “Representation learning: A review and new perspectives,” *Commun. Comput. Inf. Sci.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[38] S. R. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet and A. R. Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, “Going Deeper with Convolutions,” *Explor. Res. 9th ed.*, pp. 95–111, 2015.

[39] and J. S. K. He, X. Zhang, S. Ren, “Delving deep into rectifiers:Surpassing human-level performance on imagenet classification,” pp. 1026–1034, 2015.

[40] and D. E. C. Szegedy, A. Toshev, “Deep neural networks for object detection,” *Res. Gerontol. Nurs.*, pp. 2553–2561, 2013.

[41] Y. Sun, X. Wang, and X. Tang, “Deep Learning Face Representation by Joint Identification-Verification,” pp. 1988–1996, 2014.

[42] and Q. D. C. Yan, H. Xie, S. Liu, J. Yin, Y. Zhang, “Effective uyghur language text detection in complex background images for traffic prompt identification,” *Proc. - 2017 IEEE Int. Conf. Comput. Sci. Eng. IEEE/IFIP Int. Conf. Embed. Ubiquitous Comput. CSE EUC 2017*, vol. 1, pp. 315–320, 2017.

[43] Y. Zhang, Z. Qiu, T. Yao, D. Liu, and T. Mei, “Fully Convolutional Adaptation Networks for Semantic Segmentation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 6810–6818, 2018.

[44] L. Nie, S. Yan, M. Wang, R. Hong, and T.-S. Chua, “Harvesting visual concepts for image search with complex queries,” no. October 2012, p. 59, 2012.

[45] A. B. Badiru, “Supervised Hashing for Image Retrieval via Image Representation Learning,” *IEEE Trans. Eng. Manag.*, vol. 35, no. 3, pp. 186–190, 1988.

[46] V. E. Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, “Deep hashing for compact binary codes learning,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 2475–2483, 2015.

[47] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, “Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification,” vol. 24, no. 12, pp. 4766–4779, 2015.

[48] F. Zhao, Y. Huang, L. Wang, and T. Tan, “Deep semantic ranking based hashing for multi-label image retrieval,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 1556–1564, 2015.

[49] H. Liu, R. Wang, S. Shan, and X. Chen, “Deep Supervised Hashing for Fast Image Retrieval,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 2064–2072, 2016.

[50] W. J. Li, S. Wang, and W. C. Kang, “Feature learning based deep supervised hashing with pairwise labels,” *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2016-Janua, pp. 1711–1717, 2016.

[51] W. Liu, J. Wang, R. Ji, and Y. J. S. Chang, “Supervised Hashing with Kernels,” *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 2074–2081, 2012.

[52] G. Lin, C. Shen, Q. Shi, A. Van Den Hengel, and D. Suter, “Fast supervised hashing with decision trees for high-dimensional data,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. ii, pp. 1971–1978, 2014.

[53] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” pp. 1–14, 2014.

[54] “GitHub - Hvass-Labs/TensorFlow-Tutorials: TensorFlow Tutorials with YouTube Videos.” [Online]. Available: https://github.com/Hvass-Labs/TensorFlow-Tutorials. [Accessed: 07-Aug-2019].

[55] “CIFAR-10 and CIFAR-100 datasets.” [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Accessed: 06-Aug-2019].

[56] A. M. Giancarlo Zaccone, Md. Rezaul Karim, *Deep Learning with TensorFlow: Explore neural networks with Python*. 2017.

[57] “Classification: Accuracy | Machine Learning Crash Course.” [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy. [Accessed: 16-Jan-2020].

[58] “Loss Functions Explained - Deep Learning Demystified - Medium.” [Online]. Available: https://medium.com/deep-learning-demystified/loss-functions-explained-3098e8ff2b27. [Accessed: 17-Jan-2020].

[59] “A Study on CNN Transfer Learning for Image Classification,” *Adv. Intell. Syst. Comput.*, pp. 191–202, 2018.

[60] N. Ketkar, *Deep learning with python*. 2000.

[61] P. Senthil Kumar and D. Lopez, “A review on feature selection methods for high dimensional data,” *Int. J. Eng. Technol.*, vol. 8, no. 2, pp. 669–672, 2016.

[62] B. Chopard and M. Tomassini, “Particle swarm optimization,” *Nat. Comput. Ser.*, pp. 97–102, 2018.

[63] B. Sahu and D. Mishra, “A novel feature selection algorithm using particle swarm optimization for cancer microarray data,” *Procedia Eng.*, vol. 38, pp. 27–31, 2012.

[64] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, 2014.

[65] M. Shardlow, “An Analysis of Feature Selection Techniques,” *Univ. Manchester*, no. 1, pp. 1–7, 2016.

[66] M. Sapkota, X. Shi, F. Xing, and L. Yang, “Deep Convolutional Hashing for Low Dimensional Binary Embedding of Histopathological Images,” *IEEE J. Biomed. Heal. Informatics*, vol. 2194, no. c, pp. 1–12, 2018.

[67] Jun-yi Li and J. Li, “Fast image search with deep convolutional neural networks and efficient hashing codes,” *2015 12th Int. Conf. Fuzzy Syst. Knowl. Discov.*, pp. 1285–1290, 2015.

[68] S. Zhu, B. N. Kang, and D. Kim, “A deep neural network based hashing for efficient image retrieval,” *2016 IEEE Int. Conf. Syst. Man, Cybern. SMC 2016 - Conf. Proc.*, pp. 2483–2488, 2017.

[69] X. Lu, X. Zheng, and X. Li, “Latent Semantic Minimal Hashing for Image Retrieval,” *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 355–368, 2017.

[70] H. Wang, Y. Cai, Y. Zhang, H. Pan, W. Lv, and H. Han, “Deep Learning for Image Retrieval: What Works and What Doesn’t,” *Proc. - 15th IEEE Int. Conf. Data Min. Work. ICDMW 2015*, pp. 1576–1583, 2016.

[71] Y. Li, Y. Xu, Z. Miao, H. Li, J. Wang, and Y. Zhang, “Deep feature hash codes framework for content-based image retrieval,” *2016 8th Int. Conf. Wirel. Commun. Signal Process. WCSP 2016*, 2016.

[72] P. Li and P. Ren, “Partial Randomness Hashing for Large-Scale Remote Sensing Image Retrieval,” *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 3, pp. 464–468, 2017.

[73] Y. Kutlu and A. Yayik, “Neural Network Approach for Secure Hash-Based Color Image Authentication and Analysis,” *Int. J. Sci. Technol. Res.*, vol. 1, no. 7, pp. 27–37, 2015.

[74] Y. Z. Yang Li, Zhuang Miao, Yulong Xu, Hang Li, “Combining Nonlinear Dimension Reduction and Hashing Method for Efficient Image Retrieval,” *Asian J. Pharm. Clin. Res.*, vol. 11, no. 10, pp. 238–341, 2018.

[75] A. Jose, S. Yan, and I. Heisterklaus, “Binary hashing using siamese neural networks,” *Proc. - Int. Conf. Image Process. ICIP*, vol. 2017-Septe, pp. 2916–2920, 2018.

[76] H. Lai, P. Yan, X. Shu, Y. Wei, and S. Yan, “Instance-aware hashing for multi-label image retrieval,” *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2469–2479, 2016.

[77] N. Passalis and A. Tefas, “Learning Neural Bag-of-Features for Large Scale Image Retrieval,” vol. 14, no. 8, pp. 1–12, 2015.

[78] H. F. Yang, K. Lin, and C. S. Chen, “Supervised Learning of Semantics-Preserving Hash via Deep Convolutional Neural Networks,” *IEEE Trans.*

Pattern Anal. Mach. Intell., vol. 40, no. 2, pp. 437–451, 2018.

[79] J. Lu, V. E. Liong, and J. Zhou, "Deep Hashing for Scalable Image Search," *IEEE Trans. Image Process.*, vol. 26, no. 5, pp. 2352–2367, 2017.

[80] J. Zhang and Y. Peng, "Query-adaptive image retrieval by deep-weighted hashing," *IEEE Trans. Multimed.*, vol. 20, no. 9, pp. 2400–2414, 2018.

[81] Z. Lai, Y. Chen, J. Wu, W. K. Wong, and F. Shen, "Jointly Sparse Hashing for Image Retrieval," *IEEE Trans. Image Process.*, vol. 27, no. 12, pp. 6147–6158, 2018.

[82] S.-F. C. Jun Wang, Sanjiv Kumar, "Semi-Supervised Hashing for Scalable Image Retrieval," *Int. J. Soc. Welf.*, vol. 14, no. 2, pp. 107–115, 2005.

[83] Meina Kan, Dong Xu, Shiguang Shan, and Xilin Chen, "Semisupervised Hashing via Kernel Hyperplane Learning for Scalable Image Search," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 704–713, 2014.

[84] C. Zhang and W. S. Zheng, "Semi-Supervised Multi-View Discrete Hashing for Fast Image Search," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2604–2617, 2017.

[85] J. Zhang and Y. Peng, "SSDH: Semi-Supervised Deep Hashing for Large Scale Image Retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 1, pp. 212–225, 2019.

[86] H. Hu, K. Wang, C. Lv, J. Wu, and Z. Yang, "Semi-Supervised Metric Learning-Based Anchor Graph Hashing for Large-Scale Image Retrieval," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 739–754, 2019.

[87] Y. M. Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, "PCANet: A Simple Deep Learning Baseline for Image Classification," vol. 24, no. 12, pp. 5017–5032, 2015.

[88] V. Risojevic and Z. Babic, "Unsupervised Quaternion Feature Learning for Remote Sensing Image Classification," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 9, no. 4, pp. 1521–1531, 2016.

[89] M. W. Fahkr, M. M. Emara, and M. B. Abdelhalim, "Siamese-Twin random projection neural network with Bagging Trees tuning for unsupervised binary image hashing," *5th Int. Symp. Comput. Bus. Intell. ISCBI 2017*, pp. 14–19, 2017.

[90] H. Zhang, L. Liu, Y. Long, L. Shao, and S. Member, "Unsupervised Deep Hashing With Pseudo Labels for Scalable Image Retrieval," vol. 27, no. 4, pp. 1626–1638, 2018.

[91] "Convolutional Neural Network - MATLAB & Simulink." [Online]. Available: https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. [Accessed: 06-Aug-2019].

الخلاصة

تحليل محتويات الصورة أصبح أحد الموضوعات المهمة في الحياة الحديثة. لغرض التعرف على الصور بطريقة فعالة ، ظهرت عدة تقنيات وتم تحسينها بشكل دوري من قبل المبرمجين. عملية استرجاع الصوراصبحت واحدة من المشاكل الرئيسية التي تواجه مجتمع الكمبيوتر داخل ثورة التكنولوجيا.
لزيادة فعالية نظام استرجاع الصور يعتبر التعلم العميق في السنوات القليلة الماضية بمثابة العمود الفقري لتحليل الصور باستخدام الشبكة العصبية.

يركز هذا العمل على تصميم نظام استرجاع للصور باستخدام عدة تجارب بدءًا من أربع طبقات وانتهاءً بطبقتين، وقد تم الحصول على أفضل أداء باستخدام النظام المتكون من طبقتين حيث تتكون الطبقة الأولى من 128 عقدة والطبقة الثانية تتكون من 32 عقدة ، حيث وصلت الدقة إلى 0.9012. حيث يتيح ذلك تصميم مصنّفات صور عالية الأداء يمكن تطبيقها على العديد من التطبيقات مثل أنظمة قيادة السيارة المستقلة.

يعتمد نظام تصنيف الصور على جزأين رئيسين وهما: الشبكة العصبية و مجموعة من الصورمخزنة بطريقة منتظمة (CIFAR-10 dataset).

تركيز هذه الأطروحة على تقليل القيم المستخلصة من مجموعة الصور من أجل الحصول على أنظمة عالية الأداء لتصنيف الصور. حيث تم استخدام احدى خوارزميات الذكاء الاصطناعي من اجل القيام بعملية التقليل.

أظهرت التجارب قوة أداء الأنظمة المقترحة ، حيث تم استخدام هذه الأنظمة لتصنيف صور من (CIFAR-10) وكذلك استخدام صور محملة من الانترنت.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة الأنبار
كلية علوم الحاسوب وتكنولوجيا المعلومات
قسم علوم الحاسبات

تطبيق الشبكة العصبية الملتوية المعدلة على أساس طريقة تحسين سرب الجسيمات لاسترجاع الصور

رسالة مقدمة الى

قسم علوم الحاسبات – كلية علوم الحاسوب وتكنلوجيا المعلومات جامعة الانبار،

وهي جزء من متطلبات نيل درجة الماجستير في علوم الحاسبات

قدمت من قبل

سمير اسماعيل علي

بإشراف

أ.د سفيان تايه فرج الجنابي

و

أ.د بلال اسماعيل الخطيب

1441 هـ 2020م