REAL-TIME ADAPTIVE DATA-DRIVEN PERCEPTION FOR ANOMALY PRIORITY SCORING AT SCALE

by

SEYED ALI MIRAFTABZADEH, M.Sc.

DISSERTATION
Presented to the Graduate Faculty of
The University of Texas at San Antonio
In Partial Fulfillment
Of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL AND COMPUTER ENGINEERING

COMMITTEE MEMBERS:
Paul Rad, Ph.D., Co-Chair
Mo Jamshidi, Ph.D., Co-Chair
Chunjiang Qian, Ph.D.
Hejamadi Rao, Ph.D.
Jeff Prevost, Ph.D.

THE UNIVERSITY OF TEXAS AT SAN ANTONIO
College of Engineering
Department of Electrical and Computer Engineering
December 2017

DEDICATION

This dissertation is dedicated to my loving parents.

ACKNOWLEDGEMENTS

This dissertation could not have been completed without the many individuals who have helped me along the way. First and foremost.

I would like to express my deepest gratitude to my supervisors, Dr. Paul Rad and Professor Mo Jamshidi, who awarded me the opportunity to pursue my research under their supervision. Their profound knowledge and engineering curiosity have set high standards and are a perpetual source of inspiration and brilliance. They have been instrumental throughout all my PhD endeavor.

I would like to acknowledge the following:(i) Support by NSF grant CNS-1419165 to the University of Texas at San Antonio, (ii) time grants to access the Chameleon Cloud (NSF Cloud), and (iii) U.S. Air Force Research Laboratory and OSD, USA under Grant AQ1 FA8750-15-2-0116.

This Masters Thesis/Recital Document or Doctoral Dissertation was produced in accordance with guidelines which permit the inclusion as part of the Masters Thesis/Recital Document or Doctoral Dissertation the text of an original paper, or papers, submitted for publication. The Masters Thesis/Recital Document or Doctoral Dissertation must still conform to all other requirements explained in the Guide for the Preparation of a Masters Thesis/Recital Document or Doctoral Dissertation at The University of Texas at San Antonio. It must include a comprehensive abstract, a full introduction and literature review, and a final overall conclusion. Additional material (procedural and design data as well as descriptions of equipment) must be provided in sufficient detail to allow a clear and precise judgment to be made of the importance and originality of the research reported.

It is acceptable for this Masters Thesis/Recital Document or Doctoral Dissertation to include as chapters authentic copies of papers already published, provided these meet type size, margin, and legibility requirements. In such cases, connecting texts, which provide logical bridges between different manuscripts, are mandatory. Where the student is not the sole author of a manuscript, the student is required to make an explicit statement in the introductory material to that manuscript describing the students contribution to the work and acknowledging the contribution of the other author(s). The signatures of the Supervising Committee which precede all other material in the Masters Thesis/Recital Document or Doctoral Dissertation attest to the accuracy of this statement.

December 2017

REAL-TIME ADAPTIVE DATA-DRIVEN PERCEPTION FOR ANOMALY PRIORITY SCORING AT SCALE

SEYED ALI MIRAFTABZADEH, Ph.D.
The University of Texas at San Antonio, 2017

Supervising Professors: Paul Rad, Ph.D. and Mo Jamshidi, Ph.D.

With the aim of ultimately contributing to humanitarian response to disasters and violent events, detecting anomalies in daily human life is a crucial requirement for developing secured smart-home and smart-communities in the context of smart cities. In early security systems, it was necessary for a team of security experts to analyze a vast amount of surveillance data from a network of cameras, for instance to pick out patterns of human behavior identified as potential harmful threats. Now, however, in the big data era, online excavation and interpretation of streamed zettabyte data requires two automated technologies: (1) intelligent models - to extract suspicious patterns and discover latent anomalies; and, (2) agile systems - to take real-time action based on decision-making processes. As such, the two primary contributions of this dissertation are: (1) developing accurate intelligent models that perform much like human precision, and (2) proposing sub-systems of smart city infrastructure that intimately incorporate these intelligent models.

For this dissertation, pattern recognition models with applications in real-time video analysis were developed based on four computer vision tasks: (1) identity recognition, (2) object detection, (3) gesture recognition, and (4) action recognition. Applications of these models include, but are not limited to, recognition of: suspicious identities, active threats (life-threatening events i.e. bomb threats, civil unrest, criminal activity, earthquakes, evacuations, fires, hazardous materials), and suspicious packages. To perform these tasks, the intent is to have the models emulate the processes that take place within a human brain, i.e. with a close resemblance to human neuro processing, albeit in high-powered computational machines. Deep learning, the state-of-the-art concept in artificial intelligence, was the developmental basis for the proposed cognitive models.

In order to run efficiently, these computationally intensive models rely on the use of and coordination between high-throughput, high-performance, and many-task (parallel-task) computing-enabled machines that have a high level of computing performance compared to general-purpose computers. This variety of computational resources are served at scale in a cloud system, which serves as a central location with the core building blocks needed for compute, storage and networking. Nevertheless, anomaly detection and then taking real-time actions demands a faster processing speed than what is possible when communicating with cloud networks. It requires the use of physical infrastructure that is closer to the edge, near the source of the data (end-device); otherwise, when the data is centrally processed and stored, there is too much bandwidth required. This edge-computing approach helps reduce latency for critical applications, lower dependence on the cloud, and better manage the massive deluge of data being generated. In addition, security and privacy can also be improved with edge computing by keeping sensitive data within the end-device. In this dissertation, these distributed and decentralized deep learning systems aimed at enabling smart city applications –spread throughout the end-device, edge, and cloud– are designed following the requirements of smart city infrastructure.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: INTRODUCTION

The ability to efficiently and effectively perform anomaly detection (e.g. from video feeds from a network of cameras, or social media platforms such as Facebook and Instagram) is increasingly important, as made evident through recent real-world events (e.g. terrorist attacks on places of mass gatherings in different countries). However, real-time anomaly detection utilizing data from a network of cameras, such as those deployed in a smart-home or smart-communities – in the context of smart cities – as well as from other sources, such as social media platforms, remains a challenging task. In early security systems, it was necessary for a team of security experts to analyze a vast amount of surveillance data – from a network of cameras. Now, however, in the big data era, online excavation and interpretation of streamed zettabyte data requires two automated technologies: (1) intelligent models - to extract suspicious patterns and discover latent anomalies; and, (2) agile systems - to take real-time action based on decision-making processes. To perform these tasks, the ideal paradigm-shifting development would be to emulate human vision and human perception mechanisms through artificial intelligent (AI) machines.

Human vision perception is one of the most remarkable processes that has ever existed. From sparse, noisy, hopelessly ambiguous local scene measurements our brain manages to create a coherent global visual experience. But, how can this task, **while seemingly effortless for humans, remain so excruciatingly difficult for a computer?** AI is the science of training computers to perform tasks that typically require human intelligence to complete. At its core is the ability for the machine to learn how to apply logic and reason to gain an understanding from very complex data. Simply put, the machine learns from data it receives by identifying patterns and relationships within the data itself.

> *Machine learning focuses on the construction and study of systems that can learn from data to improve a performance function, such as optimizing the expected reward or minimizing loss functions. The goal is to develop deep insights from data assets faster, extract knowledge from data with greater precision, improve the bottom line and reduce risk.*
>
> — *Thompson*

One of the main foundational building blocks of AI is machine learning (ML) which gives computers the ability to learn and act without being explicitly programmed. ML explores the construction of algorithms that make data-driven predictions and decisions through building a model from sample inputs. These models evolve from pattern recognition and computational learning theory. ML has been widely employed in a range of computing tasks where modeling and programming explicit algorithms with reliable, repeatable performance is difficult or infeasible; examples of this include email filtering, optical character recognition (OCR), robot control, and computer vision.

The research performed for this dissertation focused on visual perception, which is the dominant sense in humans and has been used from the first days of building AI machines. Human perceptual modality of objects and the totality of a visual scene is accomplished by the coupled mechanisms in the eyes and brain. Computer vision (CV) is an interdisciplinary science which aims to give a similar, if not better, capability to programmable machines. It tackles the hard problems of analysis and understanding of useful information from a single image or a sequence of images. It involves the development of a theoretical and algorithmic basis for achieving automatic visual understanding. For this dissertation, ML algorithms were developed to enable machine perception with automated anomaly detection in the context of CV. The applications of such models are numerous and include: augmented reality, autonomous vehicles, biometrics, character recognition, forensics, facial recognition, gesture analysis, robotics, security and surveillance, and transport. For example, research topics in CV which can enhance and support these visual perception and decision-making processes are:

- Classification and Regression for Detection and Tracking of Human Faces in Different Poses
- Object Recognition Through Generative Model-based Approaches
- Cognitive Loop Models for Robust Environment Perception
- Automated Tasks Pipelines for Combining Generative and Discriminative Recognition

Part of the answer to the questions raised in regard to visual perception, such as "while it is seemingly effortless for humans, why does it remain so excruciatingly difficult for a computer," is that humans rely on years of prior visual experience to make sense of the world, while computers have to start tabula rasa. Clearly, research is needed to make progress on this severely unconstrained problem. Viewing the CV problem from this angle makes ML a potential method with a high probability for success, since the performance of ML methods is heavily dependent on the choice of data representation (or features) on which they are applied. The rapidly developing field of representation learning is concerned with questions surrounding how human can best learn meaningful and useful representations of data. For the purposes of this research, a broad view of the field is considered and includes topics such as DL and feature learning, metric learning, compositional modeling, structured prediction, reinforcement learning, and issues regarding large-scale learning and non-convex optimization. The range of domains to which these techniques apply is also very broad, from vision to speech recognition, text understanding, music, and more. The primary concern is the ability to gather enough data.

Today's ubiquitous technologies collect data in numerous applications; for instance, social media, the internet, surveillance systems, and other online communications all gather data from across the world. Efficient and accurate management and interpretation of this massive amount of information is a grand challenging in this new era of big data. Three major specifications for defining big data, including volume, velocity and variety, force researchers to seek out innovative approaches to collect, transfer, store and process big data. Based on trends, it is anticipated that the volume of big data will reach the Exabyte level by 2020. The variety of data collected and data sources are getting more complex from across the science, business and government sectors.

Finally, the velocity of collection has rapidly increased and the previous need for batch processing is on its way toward switching to real-time processing in the competitive marketing environment. There are three general steps for handling big data: implementation, management and processing. Interpretation of each of these steps can be different for different applications to reach a desired level of system efficiency. Researchers are looking for an approach that removes the gap between using big data for different applications and builds a bridge between implementation, management and processing.

This dissertation is also focused on managing, processing, and interpreting video data at scale – the next frontier for big data – with the aim of developing a data-driven perception for anomaly detection systems. "Perception" refers to the process of becoming aware of the elements of the environment through physical sensation, which can include sensory input from a network of cameras. In order to, from an engineering perspective, contribute to humanitarian response to disasters and violent events as well as detecting anomalies in daily human life, this dissertation seeks to develop systems of machines that can perform automatic tasks based on capabilities found in the human visual system. A machine can ingest massive amounts of information, extract key features, determine a method of analysis, write the code to execute the analysis and produce intelligent output, all through an automated process. Once operational, this newly developed automated process would require minimal intervention (though substantial in influence) from its human counterparts.

In this regard, the proposed pattern recognition models were developed for real-time video analysis applications based on four computer vision tasks: (1) identity recognition, (2) object detection, (3) gesture recognition, and (4) action recognition. Applications of the proposed models include, but are not limited to recognition of: suspicious identity, active threats (life-threatening events i.e. bomb threats, civil unrest, criminal activity, earthquakes, evacuations, fires, hazardous materials), and suspicious packages. To perform these tasks, the intent is to have the models emulate the processes that take place within a human brain, i.e. with a close resemblance to human neuro processing, albeit in high-powered computational machines. Deep learning, the state-of-the-art concept in artificial intelligence, is the developmental basis for the proposed cognitive

models. Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) were leveraged to implement the novel deep learning models presented in the papers published towards completing this dissertation. Irrespective of the type of neural network used, all the deep learning algorithms perform sophisticated mathematical and statistical computations.

In order to run efficiently, these computationally intensive algorithms rely on the use of and co-ordination between high-throughput, high-performance, and many-task (parallel-task) computing-enabled machines that have a high level of computing performance compared to general-purpose computers. This variety of computational resources are served at scale in a cloud system, which serves as a central location with the core building blocks needed for compute, storage and networking. Nevertheless, detecting anomalies and then taking real-time action based on this detection demands faster processing than what is possible when communicating within cloud networks. It requires the use of physical infrastructure that is closer to the edge, near the source of the data, versus the bandwidth required when the data is centrally processed and stored. This edge-computing approach helps reduce latency for critical applications, lower dependence on the cloud, and better manage the massive deluge of data being generated. In addition, security and privacy can also be improved with edge computing by keeping sensitive data within the end-device.

Hence, distributed and decentralized deep learning systems - throughout the end-device, edge, and cloud ? have been designed following the requirements of smart city infrastructure. These systems incorporate intimately with the proposed deep learning models for real-time data-driven perception for anomalies on low-powered devices running at edge. Localized core building blocks - compute, storage and networking - are devised closer to the end-device at the edge while global building blocks are placed in the cloud. All the inference operations of the computer vision tasks are implemented as a visual intelligent technology, referred to as ?AI embedding,? on the end-device. By performing computer vision tasks within the end-device, AI embedding reduces the amount of cloud resources (bandwidth, processing unit, and storage) used when compared with the alternative of sending raw streams of video over the network. The generated private embeddings on

individual end-devices serve as a private machine-to-machine communication (M2M) language.

Dissertation Statement: develop new intelligent, accurate deep learning models, close to human precision, for scene perception and content interpretation for anomaly detection in crowds, and devising sub-systems of smart city infrastructure that intimately incorporates with the proposed models to support real-time data-driven perception for anomalies at scale and prioritize them, as well as, keeping the data privacy concerns.

1.1 Dissertation Contributions

The primary contributions of this dissertation are summarized in the following points:

- Proposing subjective loss function and new architecture for prediction layer in deep learning (Chapter 2)
- Presenting a configurable, interactive, embedded system to extract object features (i.e. faces) in crowds. The proposed system generates unique object embedding using the researcher?s privacy-aware ResFace neural network and vectorized-l2-loss function in the training phase (Chapter 3).
- Implementing the DNNs pipeline to extract crucial information from the scene for further verification. Potential applications of this system include places of mass gatherings and key (critical infrastructure) installation sites, which can be used to identify and re-identify objects of interest (Chapter 3).
- Presenting a decentralized biometric access control system to ensure identity consistency in cyber physical space (Chapter 4).
- Presenting a novel neural network architecture, named LSTM-ResNet, that has been designed to learn the temporal dynamics of human biometrics, such as faces, for authentication (Chapter 4).

- Presenting a model that was designed to implement machine learning algorithms as an embedded device to keep data securely decentralized on Internet of Things (IoT) devices for real-time online and offline applications (Chapter 4).

- Introducing a Ranking Method to serve as the elastic and inelastic task scheduler to support the heterogeneous requests and resources of machine learning applications in a multi-cloud environment (Chapter 5).

- Proposing an intelligent decision support system for heterogeneous autonomous agents (Chapter 6).

Chapter 2: ON THE IMPORTANCE OF SUBJECTIVE LOSS FUNCTION AND PREDICTION LAYER IN DEEP LEARNING

Deep neural networks (DNNs) are currently among the most commonly used machine learning methods in computer vision. One of the best characteristics of these methods is their modular design [65] – the ability to change the connectivity patterns of layers, try different activation functions, inject different statistical approaches (for instance, normalization and dropout) in the network and many other actions – in every aspect of deep learning networks. While the majority of deep learning applications simply use cross-entropy and L_1, and L_2 losses, subjective loss function can actually result in impressive performance improvement. In addition, architecting the last layer of DNNs, referred to as the prediction layer, according to the needs of the application increases the discriminative power of the DNNs. This chapter aims to investigate how particular choices of loss functions and prediction layer architecture affect deep neural networks and their learning dynamics, as well as the robustness of various effects.

2.1 Introduction

The aspirations in computer vision (CV) tasks – for instance: object recognition, scene categorization, integrative scene understanding, human motion recognition, material recognition, etc. – is developing methods for acquiring, processing, analyzing and understanding digital images; and, the extraction of high-dimensional data in order to produce numerical or symbolic information so that machines can ingest massive amounts of information, extract key features, and determine a method of analysis – all through an automated process. In CV, content perception is a challenging high-dimensional interpolation problem. It requires the extraction of a pattern to distinguish invariants, symmetries, and diffeomorphisms. Four different types of machine learning algorithms that address the high-dimensional interpolation problem are: supervised learning, unsupervised learning, semisupervised learning, and reinforcement learning.

The intent of supervised learning – which the majority (about 70 percent) of machine learning

methods refer to – is training a system with already known labeled data to predict the values of the labels through methods like classification, regression, prediction and gradient boosting. About 15 percent of machine learning is unsupervised learning; in this case, the system is not told the "right answer", and the expectation from unsupervised learning algorithm is to find a hidden structure or manifold in unlabeled data. Semisupervised learning is used for the same applications as supervised learning, but this technique incorporates both labeled and unlabeled data for training – typically, a small amount of labeled data with a large amount of unlabeled data. With reinforcement learning, the algorithm discovers for itself which actions yield the greatest rewards through trial and error. The objective is for the agent to choose actions that maximize the expected reward over a given period of time.

Content perception is a high-dimensional interpolation problem. It requires a transformation to low-dimensional linear projection. This type of projection requires the cultivation of data in order to extract unseen features, and learns short and long term data dependencies. Feature representations have to decide to suppress a variation or, in contrast, maintain it. Fixed feature representations are not agile enough to interpret natural data; but, deep contractive projection generalizes the feature transformations for natural data. For all these reasons, supervised learning was the primary type of machine learning used in for the research performed for this dissertation.

In supervised learning, a function $F(x)$ is approximated from q training samples $\{x^i, f(x^i)\}_{i<q}$, where x is a data vector of very high dimension $d(\geq 10^6 in real world problem)$. To reduce the dimension, advanced neural network architectures try to separate different values of f by applying a low-dimensional linear projection $\varphi(x)$ such that if $f(x) \neq f(x')$, then $\varphi(x) \neq \varphi(x')$. To distinguish invariants, symmetries, and diffeomorphisms the strategy was to linearize the variations of f with a first change of variable $\varphi(x) = \{\varphi_k(x)\}_{k\leq d'}$, which is also considered Deep Learning. In this regard, neural networks show the state-of-the-art performance because of their inter network contractions. Convolutional Neural Networks are capable of extracting unseen features and Recurrent Neural Networks enable the decision-making process based on an understanding of previous events, and a special kind of recurrent neural network (RNN), called long short-term

memory (LSTM), is capable of learning long-term dependencies.

One of the best characteristics of DNNs is their modular design – allowing for actions such as changing connectivity patterns of layers, trying different activation functions, injecting different statistical approaches (for instance, normalization and dropout) in the network and many others – in every aspect of them as deep learning networks. While the majority of deep learning applications simply use cross-entropy or L_1 and L_2 losses, subjective loss function can result in impressive performance improvement. In addition, the ability to architect the last layer of DNNs, referred to as the prediction layer, according to the needs of the application, increases the discriminative power of the DNNs. In this chapter, through a case study, the effect of particular choices of loss functions and prediction layer architecture on deep neural networks is reviewed, along with their learning dynamics as well as their robustness to various effects.

2.2 Case Study on Classification Task

Deep structured learning is an emerging approach that shows non-trivial breakthroughs in many machine learning applications. Instead of hand-crafted feature engineering, deep structured learning, also well-known as deep learning, uses a cascade of layers of nonlinear processing units (neurons) to learn latent data representations which correspond to different levels of abstraction. For example, in natural language processing (NLP), unfolded representations are used to analyze, understand, and derive meaning from human language in a smart and useful way. Tasks such as automated summarization, entity recognition, hidden information extraction, sentiment analysis, topic segmentation, translation, and speech recognition are objects of NLP. Potentially, customer loyalty analysis as a multidimensional problem could n be addressed by utilizing deep learning algorithms in NLP.

Single-question customer metrics have become popular as tools for measuring customer loyalty and predicting a company's future in a single score (`https://hbr.org/2012/07/there-is-no-one-best-measure-o`). Asking a question in the form of NPS is very specific and objective. Applying simple arithmetic on this score from the company's customer base

gives the customer loyalty, and allows the company to sort their customers into sets of Promoters, Passives and Detractors. Here, we address this problem as a classification task. The difficulty is that the problem can be classified into three major categories: Firstly, customer feedback is messy. From the point of view of somebody who writes computer algorithms for understanding language, text written by people in an uncontrolled environment is extremely messy. There are no regular sentences, no attention to white space, misspellings, abbreviations, autocomplete errors, etc. Given an open-ended question in a survey, many responders will simply write meaningless fillers like idk (I don't know), nothing specific, all good, yes, p, etc. Secondly, customer feedback is diverse. There are many people who just comment on one thing, e.g. too expensive; but, just as many like to list various aspects, e.g. a single sentence like - He made a painfully long process bearable with regular points relating to both dealing with the company process overall (negative), the respective customer support person (positive), and, more specifically, their communication. Customer complaints are particularly lengthy with stories explaining how the company did them wrong. Finally, customer feedback is hard to interpret due to its often subjective nature. For example, if a customer says, "She made me feel as if she really had time for me," how would one categorize this? Even a person reading this would spend a lot of time linking this aspect of customer service with similar ones that may appear in the customer responses.

The work described here contributes to the understanding and management of customer loyalty measurement by addressing all of the previously mentioned problems in customer feedback (messiness, diversity, and interpretation) utilizing a deep learning model that is based on a convolutional neural network. An algorithm was used to classify the reviewers? feedback into three categories: Promoters, Passives, and Detractors. Intrinsically, this allows one to detect the inconsistencies between the feedback and the score. In addition, as sub-products, the resulting outcome can be used for auto classification of unstructured feedback and routing, as well as entity detection of people or companies. Experiments were performed on text datasets to calculate brand NPS, as well as disclose some additional, theoretical insights into the problem. To measure customer loyalty, a deep learning model was developed based on a convolutional neural network. To address

Algorithm 2.1 DeepNPS

Input: $\boldsymbol{x}_r^i$ feedback $\boldsymbol{x}$ from rth reviewer for ith item.
Output: $P(y = j|\boldsymbol{x}_r^i)$, $j \in \{Promoters, Passives, Detractors\}$.

1. **Loop** (i, r) **:**
2. split $\boldsymbol{x}_r^i$ into sentences $\boldsymbol{s}_k, k = 1, 2, 3, ..., m$
3. **Loop** k **:**
4. split $\boldsymbol{s}_k$ into word vectors $\boldsymbol{V}_w, w = 1, 2, 3, ..., n$
5. **if** the word is not recognized:
6. $V_m =$ word2vector(**W**)
7. make matrix of $V_{word} \mapsto \boldsymbol{S}(\boldsymbol{V_w})$
8. make tensor of $\boldsymbol{S}(\boldsymbol{V_w}) \mapsto \boldsymbol{T}(\boldsymbol{S}(\boldsymbol{V_w}))$
9. **Loop** $j \in \{Promoters, Passives, Detractors\}$ **:**
10. $P(y = j|\boldsymbol{x}_r^i) =$ **DeepNPS**$(\boldsymbol{T}(\boldsymbol{S}(\boldsymbol{V_w})), \boldsymbol{W^j})$
11. **NPS** $= 10 \times \{P(y = Promoter|\boldsymbol{x}_r^i) - P(y = Deteractors|\boldsymbol{x}_r^i)\}$

this problem, a novel discriminative loss function (which is not typically used for deep nets objectives) was developed that classifies a customer into one of the three categories noted previously Promoters, Passives, and Detractors. In addition, the model can address problems in the inconsistency between the feedback and the rate, and can act as an accurate classifier to extract essential information for further analysis on customer loyalty.

2.3 Approach

A non-convex optimization problem was developed to address the multidimensional problem, analyzing customer loyalty, including a combination of behavioral and attitudinal data sources (which are referred to here as customer feedback). To assess and enhance the reliability and accuracy of the solution, a deep structured learning algorithm was applied on preprocessed input data. A convolutional neural network (CNN) with adaptive filter size was trained to make the deep learning algorithm, named DeepNPS. Grouping from different sources such as item surveys, basic sen-

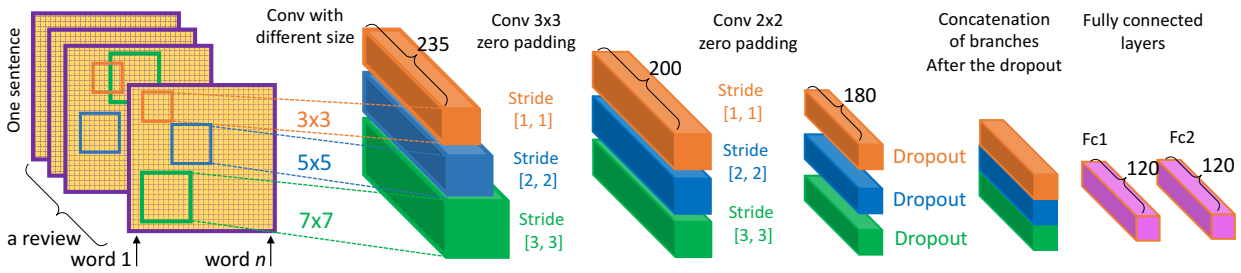


Figure 2.1: Proposed DeepNPS architecture

tences, and words were used to gather data. In addition, Web data (such as Amazon reviews [94]) and another dataset formed from multiple websites were used to train the model and compare the results.

The approach is presented in the form of a DeepNPS algorithm. Feedback x from rth reviewer for ith item is split into sentences, and each sentence is split into words. Then, each word is mapped using a pre-trained word2vec model [97]. As a result, for each feedback x, a matrix of the word vectors for each sentence was produced, and a tensor containing all these matrixes related to this feedback is fed to the proposed DeepNPS network.

In DeepNPS, the dataset passed through many layers including the embedding, convolutional, max pooling and Softmax layers. The embedding layer plays a major role in gaining the accuracy of the model. There are different profound learning models accessible for slant investigation such as RNN, Perceptron Model, CNN, Naive Bayesian classifier and so forth. In this case, CNN was selected as the model to be used, keeping in mind that the end goal was to understand all layers of the CNN and that it is one of the most intense models for sentiment handling. The dataset was bolstered into the CNN demonstrate, which had various layers to process the data and, as expected, yield an output of three classes Promoters, Passives and Detractors.

2.4 Dataset

Web data: Amazon reviews, this dataset consisted of 18 years of Amazon reviews for 35 million customers. Product, user information, rating, and each plaintext review were included in the dataset. For sentiment analysis, the original dataset was obtained from Cornell University [111],

which had a total of 10,662 lines of movie reviews. Furthermore, a second dataset was created by merging data from IMDB, Rotten Tomatoes and various other websites [20] into one big set containing 75,509 lines. The efficiency and accuracy of the model increased as a result of using this combined dataset. Increasing the size of a dataset plays a vital role in determining, based on positive or negative reviews, what are the parameters have distinguished influence on the accuracy percentage of the method.

2.5 Deep Structured Network Architecture

To consider the hierarchical structure of language: several words make a phrase, several phrases make a sentence and, ultimately, sentences convey information. For this task, the feedback is broken up into sentences, then words of a sentence are extracted and presented as a vector. The sentence is then reconstructed as a combination of these vectors and convolutional neural networks are applied with the different filter sizes on each sentence.

2.5.1 Embedding Sentence

The first and vital layer of the proposed CNN model is the embedding layer, see figure 2.1. Parameters taken into consideration are vocabulary size and sequence length. Word Embedding transforms the word to word vectors in the low dimensional vector space from the given input data. The popular equations is then followed, which explains the vector transformation:

$$\mathbf{V}_{queen} = \mathbf{V}_{king} - \mathbf{V}_{man} + \mathbf{V}_{woman} \tag{2.1}$$

and usually, in word prediction applications, is illustrated as "*a king who is not a man but a woman is queen*". The vector equivalent for the known words is considered in mapping the new words. Sentence, $\mathbf{S_i}$, with sequence length as n in denoted as:

$$\mathbf{S_i} = \mathbf{C} \circ \mathbf{V} = \mathbf{V_1} \oplus \mathbf{V_2} \oplus \cdots \oplus \mathbf{V_n} \tag{2.2}$$

where $\mathbf{C} \circ \mathbf{V}$ denotes the concatenation operation on the word vectors in the sentence. All the sentences are padded to have same length. Hence, in the further processing the vectors, equivalents for each word are considered and trained into the model for predictions and to increase its accuracy.

2.5.2 CNN Networks

A convolutional neural network is a model that has various layers including a convolutional layer, embedding layer, and fully connected layer. The model follows a non-static single channel wherein the training and prediction are performed. The first layer is the embedding layer, which embeds the input words to vectors contained in low-dimensional vector space allowing for the review of sentences and formation of vectors for further processing. A convolution process is performed over the word embedded vectors and a number of different filter sizes like 3×3, 4×4, and 5×5 to extract features. Each filter size has a number of features and the default used here was 235, which was chosen after many trial and error experiments ranging from 3 to 252 and which can be varied accordingly. The window of the filter was applied through the ith sentence of the review, $\mathbf{S_i}$, to formulate the kth feature, f_k. The varied filter sizes yield different features that are used to create a feature map essential for predicting the sentiment connected to the words. The feature f_k is calculated for the word vectors in the filter window and concatenation of the process with different f_k results in the feature map $\mathbf{F}$. For the purposes of this research, three filter sizes in $\{3\times3, 5\times5, 7\times7\}$ and k were 0, 1, or 2, respectively. By this definition, the k is called as a branch number for the rest of the paper. The network is followed by two more general convolutional layers for each of the branches and then an average-pooling operation is applied at the end of each layer. Since each filter yields results with different tensor shapes and all the vital features extracted have to be combined as a single full hot feature vector at the last layer, a dropout layer is added in each branch to add more intelligence through the learning process for each branch (feature). This means that each branch is strictly forced to learn independently. The final feature map is calculated as:

$$\mathbf{F} = \mathbf{C} \circ (\text{conv}(f_k, \mathbf{C} \circ \mathbf{V})) = \text{conv}(f_1, \mathbf{C} \circ \mathbf{V}) \oplus \text{conv}(f_2, \mathbf{C} \circ \mathbf{V}) \oplus \text{conv}(f_3, \mathbf{C} \circ \mathbf{V}) \quad (2.3)$$

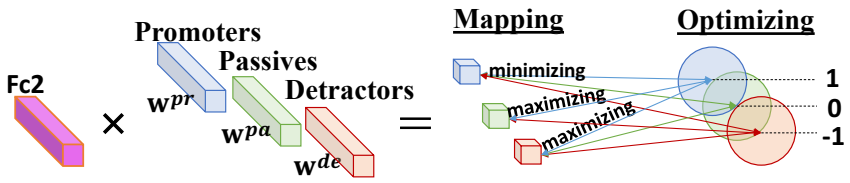


Figure 2.2: Illustration of the loss function.

The network is followed by two fully connected layers with the size of 120. At the end of the process, the Softmax layer performs an element-wise operation to form the $\mathbf{Z}$, which implies one of the categories in {Promoters, Passives and Detractors}. Thus the raw scores are generated from the vectors that yield predictions. The loss of the network is calculated using the cross-entropy loss function that results from the errors made in the network and the accuracy using the reduce-mean function helps one to understand the behavior of the network during training and evaluation.

2.5.3 Promoter, Passive, or Detractor

To address the deep NPS, the conventional architecture of the last layer of the CNN is changed. Instead of having the last layer as a classification layer, three separate discriminative variable vectors are proposed for each category in {Promoters, Passives, and Detractors} as $\mathbf{W^{pr}}$, $\mathbf{W^{pa}}$, and $\mathbf{W^{de}}$, respectively. This leads to suppressing the linear dependency of the usual only variable vector at the last layer. At the end of the process, multinomial logistic regression was applied for each category independently using the softmax function. The input to the function is the result of K (here 240), distinct linear functions from the last fully connected layer. The probability for a customer loyalty, which is the category $j \in$ {Promoters, Passives and Detractors} is given by:

$$P(y = j|\mathbf{x}) = \frac{\exp(\mathbf{x}^T \mathbf{w}_j)}{\sum_k^K \exp(\mathbf{x}^T \mathbf{w}_k)} \tag{2.4}$$

The proposed loss function is leveraged to address the non-convex optimization problem in terms of minimizing the eucludium distance of each category with the corresponding mapping (see figure

2.2 for the illustration of the loss function).

$$\min_{\mathbf{W^{pr}},\mathbf{W^{de}},\mathbf{W^{pa}}} \left\{ \begin{aligned} &||y^{\text{th}} - \frac{\exp(\mathbf{x}^{\mathrm{T}}\mathbf{w}^{\text{pr}})}{\sum_k^K \exp(\mathbf{x}^{\mathrm{T}}\mathbf{w}^{\text{pr}})}||+ \\ &||y^{\text{th}} + \frac{\exp(\mathbf{x}^{\mathrm{T}}\mathbf{w}^{\text{de}})}{\sum_k^K \exp(\mathbf{x}^{\mathrm{T}}\mathbf{w}^{\text{de}})}||- \\ &||y^{\text{th}} \times \frac{\exp(\mathbf{x}^{\mathrm{T}}\mathbf{w}^{\text{pa}})}{\sum_k^K \exp(\mathbf{x}^{\mathrm{T}}\mathbf{w}^{\text{pa}})}|| \end{aligned} \right\} \tag{2.5}$$

in which yth represents the ground truth mapping {Promoters, Passives and Detractors} $\rightarrow \{1, 0, -1\}$, respectively. It is worthwhile to emphasize two clues. First, as a result of introducing three different variable vectors, $\{\mathbf{W^{pr}}, \mathbf{W^{pa}}, \mathbf{W^{de}}\}$ to three different categories, the proposed loss function aims to train each variable vector on the target category while keeping the distance from the other two non-target categories. In addition, since the NPS formula focuses on the Promoters and Detractors, $\mathbf{W^{pa}}$ is used on the optimization problem to tune and refine the CNN network intrinsically.

2.6 Experiment and Results

This model was implemented in large-scale machine learning platform, TensorFlow [1]. The CNN model was first run with the dataset that consisted of only 10,662 lines of data and evaluated for the same; the loss was very high and almost increased to the peak for a 10 scale. Different values for dropout rendered different results as they impacted the training, test split, and the corresponding loss and accuracy percentage along with the graphs were recorded. Hence, as a next trial the second dataset was used, which had much more data to use to test and train the model (equaling almost 75,509 lines of data). Further, the same parameters were altered with different values within the range and the corresponding values and graphs indicated a remarkable increase in accuracy and reduction in loss.

Multiple training iterations were executed with different values, changing one parameter at a time, altering both like drop out, number of filters and the results observed to note down the perfect values of each parameter to find the most efficient one. The fine tuning of the parameters was key

Table 2.1: Accuracy percentage of the prediction

Test Class	Sample Examples	Accuracy Percentage		
		Promoters	**Passives**	**Detractors**
True Positive	1	98.97	92.03	99.48
	2	96.83	100	97.98
	3	100	90.01	98.02
False Positive	1	1.03	7.93	0.52
	2	3.17	0	2.02
	3	0	9.99	1.98
True Negative	1	0.01	0.08	0.01
	2	1.08	0	0.13
	3	0	0.09	0.04
False Negative	1	99.99	99.92	99.99
	2	98.02	100	99.87
	3	100	99.91	99.96

to increasing the prediction accuracy of the model. The range of the values used for each parameter was noted down and analyzed for best results.

All these executions files were utilized to visualize the flow graph of the model using the tensor board application. The flow graph clearly depicts the model architecture, which aids in understanding all of the details. Nodes in the layer show the internal construction of each function used in the network.

The accuracy for the data was calculated in two different ways; one by using the whole test data, which was split into test and train sets, and the other by entering any random review sentence not originally present in the dataset. The prediction was noted for every other sentence and accuracy was calculated. This method improved the understanding and enhanced the analysis of the model and its prediction accuracy. The test cases were formulated by checking for true positive, true negative, false positive and false negative cases for each category in {Promoters, Passives, and Detractors}. The negative sentence entered was given a label as [1, 0] and the positive sentence as [0, 1] as the input data was being labeled. A table, which depicts the accuracy percentage of the test cases, is shown below. Please note that the reported accuracy in the table is calculated based on the number of reviewers in each category not as a whole reviewer for a sample example.

Table 2.2: Examples of the studied feedbacks from Amazon for four selected items

Item #1: Samsung Countertop Microwave	
Promoters	I like the handle rather than push button open; the push button broke on my last microwave. Working great. No problem popping microwave popcorn. No smell. The instructions tell you how to turn off the beeps. The interior is easy to clean.
	Received this MW on time. Very nice looking and would recommend.
	This is perfect for a basic microwave. The size is nice and although it has a little less power than the one it replaces, it meets my needs perfectly.
Passive	I like it and the knob is pretty cool once you get used to it. I do think its seems to be getting a little louder than when I first received it but I still am happy with it.
	This microwave looks nice, but it's the most user friendly. I wish the knob was able to set the time freely without pushing the "cook" button and then the power level. Also I wish it had a kitchen timer feature. Other than that no complaints cooks stuff really well!!
Detractors	I'm not satisfied with this microwave, it's so noisy, takes double time that my old microwave to warm my food... light is awful inside... lots of little details, Samsung must keep working on making microwaves better because this model is horrible. I want to return it!!
	Glass is too dark/light inside too dim - cannot see what is happening to food.
	Used the microwave twice before it died. Returned for refund. Went to Sears bought a Kenmore. Reviews said it was a good product. It was loud like some reviews stated.
Item #2: Coffeemaker Keurig15	
Promoters	This is the 2nd one I'm getting. I love it!!! I am the only one who drinks coffee in the house, so no waste. I get to have a fresh hot cup when needed. Easy to clean and use. I would highly recommend this machine.
	I've had Keurigs before and have loved them all. This one is simple and easy to use. And inexpensive.
	Had another one that lasted for years. Finally given up the ghost we purchased this one. It works better than the old one by brewing coffee much faster. Happy with it.
Passive	I have to take a 1 star away for the fact that, without removing the drip tray, it doesn't really fit anything other than standard 6-8 oz cups (like you might find a Denny's or IHOP). I would take a 2nd star away because, although it can brew a 10 oz cup, it doesn't fit ANY of our insulated travel mugs. Obviously that's not Keurig's fault but it still annoys me that I have to brew into one cup and pour it into my travel mug..
Detractors	Unfortunately, it doesn't make great coffee. If you try to make a standard mug of coffee (10 oz) it'll taste watery and weak, 8oz seems to be the limit. When I use my own grind, it has to be fine, and I use the maximum amount of coffee the basket will hold. With commercial K-cups I find most roasts are burnt (even those claiming 'medium roast'), presumably to try and mask the watery taste. The coffee also comes out warm, not hot at all, which could be the reason it's watery. In any case I have to microwave it for at least 20 seconds to get it reasonably hot. So for quality of coffee, just 1 star.
Item #3: Hero5gopro	
Promoters	I previously had a gopro hero 3. I loved everything about it. After several years of camping, white water rafting, scuba diving . .. the camera still runs without a flaw. For the price of the new hero 5, I decided to try an upgrade. I picked one up at best buy this morning. I didn't think gopro could improve on what I already thought was an awesome product, but they did. Voice control works great. Waterproof without the case. Love it. My son is happy also since he now gets to keep my old camera. He already placed a mount on his rc truck and is out recording the truck from a first person view. Thank you gopro.
Passive	It came with nice packaging but screen doesn't work properly screen has black spots and dirt inside the screen it was the biggest disappointment for me. That was the first product that i bought from gopro company and definitely it should be the last.
	Camera is overall great everyone knows this, but one thing the camera frame is not that great. it's very hard to unlatch, not have access for charge port as well replacement of battery without the removal of camera from this frame.
Detractors	Haven't used it outdoors yet because it became very hot after only a couple of minutes being turned on.
	I finally bought the hero 5 and I am thoroughly disappointed with one thing. The battery system is medieval. You have to use tools to remove the battery. The tab came off. In the process sd card got screwed in its socket. It's that hard. Why can't the company make simple clip system? Now I will return the stuff and ask for a replacement.
	Battery life is an issue and since the power-up sequence is so complex, you need to leave on standby.
Item #3: Polaroid Snap Touch	
Promoters	This camera was great for what my wife and I wanted to use it for. When we got married we took the camera with us to capture our special day. WE then used it to make a photo book of the day that we could share the moments with family. We eloped in Ireland so no one was there at all and we wanted to show our family what our wedding was like.
	If you are just looking for a camera with decent quality instant print photos then this is perfect otherwise dont expect premium prints. Certain colors such as white scenes come out looking a little streaky but most other colors come out just fine. The bluetooth feature is nice because u can print photos from a device so it almost acts as a mobile printer as well.
Passive	Camera works good. Photo quality for the megapixel is so so. Lens is small so it really holds back the quality. Photo prints are good for the size. Not sure reprints of large size would be very good. If you want a good little camera to do quick point and print it's great. If you want a digital to take family pics to keep buy a Sony with a Carl Zeiss lens. Overall it's good just not what it could be but then it would cost more.
Detractors	Bought this as a Christmas gift for my daughter and was never able to use because a message keeps popping up saying the door latch is not completely closed, when in fact it is closed. Too much time has passed and I'm unable to get any tech support from seller but am currently going back and forth with Polaroid directly to see if we can get this fixed. Otherwise it's a nice camera, the color isn't really a deep red it's more like a pastel coral color.
	The one I got must have been a factory defect. It only worked less than a few hours. I had to return it for a full refund. I'm afraid to try another one.

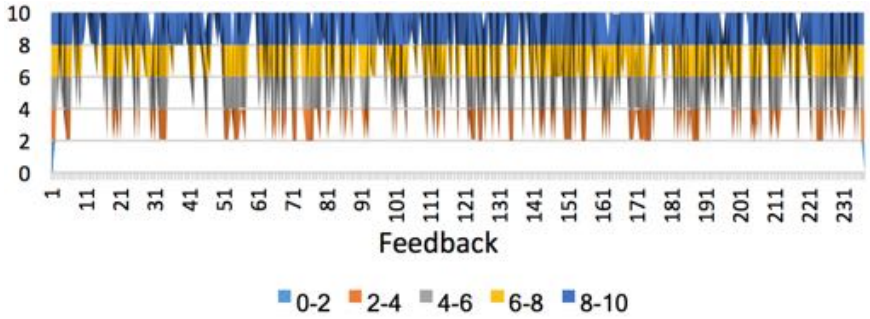


Figure 2.3: Illustration of the DeepNPS result to present customers' loyalty for a sampled item

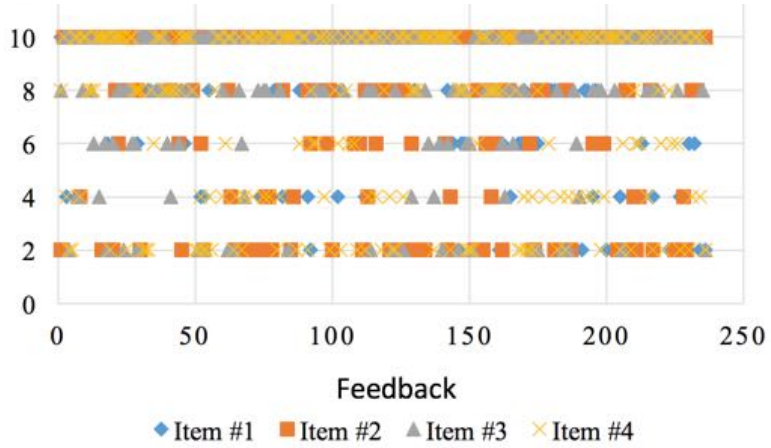


Figure 2.4: Visualizing the DeepNPS results for four item feedbacks

Table I shows the accuracy percentage of the prediction with a random review sentence inserted for evaluation. For each case, three (3) sample examples were given. In the true positive case the input sentences were entered with correct labels. A positive sentence was entered with [0, 1] as the label, a negative sentence was entered with [1, 0] as the label, and the predicted result is also a positive class. A false positive case is determined when prediction accuracy is much less when the entered test case was positive but the predicted result was a negative class. A true negative case is when the entered negative sentence is predicted correctly. A false negative test case is when a negative test case was predicted as a positive test case. The accuracy percentage depicted that the prediction is almost exact for the test cases predicted. This test case analysis helped for interpreting more details about the behavior of the model with different data inputs.

The results of one of the case studies are presented in figures 2.3, 2.4, and 2.5 – which in each a subsample of the 232 feedbacks for four items selected from Amazon were used. Some of

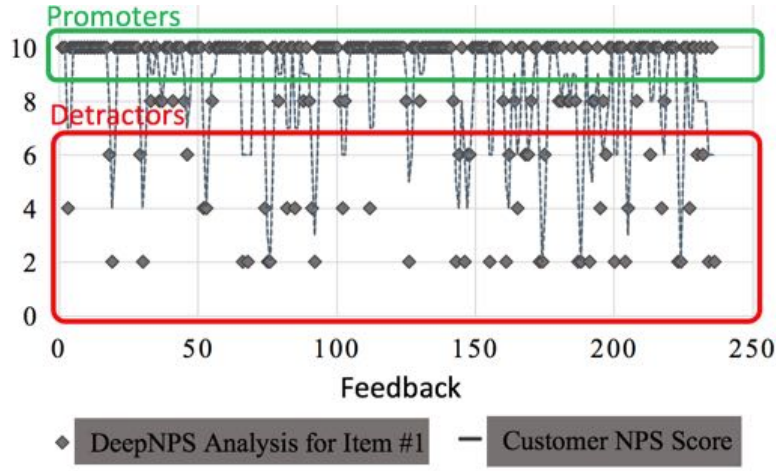


Figure 2.5: Visualizing the DeepNPS results for four item feedbacks

these feedbacks are listed in Table II. Following DeepNPS algorithm steps, results of the proposed DeepNPS model based on CNN for a sampled item are depicted in figures 2.3, 2.4, and 2.5.

The error with the reported NPS is 0.16%. Figure 2.5 illustrates this comparison while down-sampling was applied to the data for more clear visualization. In figure 2.5, the dashed line graph depict the trends of the reported score and the diamonds depict the output of the DeepNPS model of the feedbacks. The Green zone characterizes the promoters and the red zone reflects the detectors scores. As figure 2.5 verifies, DeepNPS follows customer loyalty strictly. The green zone reflects the promoters' scores and red line illustrates the detractors' scores.

2.7 Conclusion

The DeepNPS model is presented for analyzing customer loyalty based on customer feedback. The deep learning network based on convolutional neural network architecture was trained using a novel loss function to address the problem. Feedback from the promoters and detractors were matched to 1, and -1 respectively, while passive customers were matched to 0 and used to add more discriminative power to the network. The distinguished accuracy achievement makes DeepNPS not only a trustworthy gage for recognizing customer loyalty, but it can also be used as an accurate customer feedback classifier for extracting more information from the customers' point of view. For instance, to determine what characteristics of the product make the NPS high or low, or how a

delivery system effects customer feedback on the item.

Chapter 3: A PRIVACY-AWARE ARCHITECTURE AT THE EDGE FOR AUTONOMOUS REAL-TIME IDENTITY RE-IDENTIFICATION IN CROWDS

This chapter has been published: Miraftabzadeh, Seyed Ali, Paul Rad, Kim-Kwang Raymond Choo, and Mo Jamshidi. "A Privacy-Aware Architecture at the Edge for Autonomous Real-Time Identity Re-Identification in Crowds." IEEE Internet of Things Journal (2017).

The capability to perform identity re-identification in a crowd (e.g. from video feeds from a network of cameras, and social media platforms such as Facebook and Instagram) efficiently and effectively is increasingly important, as evident in recent real-world events (e.g. terrorist attacks on places of mass gatherings in different countries). However, real-time re-identification in a network of cameras, such as those deployed in a smart city, and from other sources, such as social media platforms, remains a challenging task. In this paper, a new embedding algorithm pipeline is presented to extract and administrate the crowd-sourced facial image features (e.g. social media platforms and multi-cameras in a dense crowd, such as a stadium or airport). The proposed facial embedding is a privacy-aware parameterized function, which maps facial images to high-dimensional vectors in order to facilitate the identification and tracking of individuals. In other words, we are able to uniquely identify person(s) of interest, without the need to determine their true identity. To extract the facial embedding information in crowds, concurrent Residual Neural Network (ResNet) embedding pipeline for each camera is proposed. Specifically, facial embedding feature vectors are generated in real-time by each camera using the proposed enhanced ResNet architecture, which is trained with vectorized-l2-loss function for face recognition. The multivariate kernel density estimation (MKDE) matching algorithm is then applied to facial embedding pipelines generated by cameras at the fog cloud for identity re-identification and security verification. This allows us to ensure the privacy of individuals captured by the camera without

compromising on the capability for identity re-identification. Evaluations using mixed datasets in real-time demonstrate that our proposed approach achieves a 2.6% accuracy over other state-of-the-art approaches.

3.1 Introduction

Accurate, efficient and automated real-time identity detection platform plays an important role in ensuring the security and resilience of a (smart) city and her residents. The application of identity detection to recognize and track suspicious individuals (or persons of interest) is becoming more feasible due to the increasing deployment of surveillance cameras and increasing sources of open/closed source intelligence (e.g. data from social networks such as Instagram, Twitter and Facebook, as well as government systems). Computer systems interact with their environment, such as real-time surveillance monitoring services, may operate without human supervision or intervention. Automating identity detection necessitates top-down analysis and fusion of events from network and system monitors to accurately detect an identity (e.g. a person of interest) without human interaction. Interactive computing systems usually operate by modeling how their environment is changing and may shift goals as conditions change. We posit that edge computing can facilitate and enhance real-time decision making, since computing and storage nodes are placed in close proximity to Internet of Things (IoT) devices or sensors. In other words, by placing the computation power at the edge of the network helps with the physical implementation of a real-time decision making model for detecting an identity at the right time in the right place.

Contemporary supervised machine learning methods can achieve high accuracy in identity recognition based on facial images [135, 155], partly due to the availability of large data and computation resources (e.g. distributed cloud systems) [85, 126]. Generally, training the supervised machine learning models is a time consuming process, which can range from days to even weeks [162, 177]. Cutting edge hardware technologies such as distributed processing platform on multiple graphics processing units (GPUs) can potentially reduce the training process to a couple of minutes. In addition, combination of a high performance distributed machine learning platform

with techniques, such as transfer learning, can reduce the offline training time of a pre-trained model while enhancing the accuracy of an online trained model overtime.

However, supervised machine learning is not always a panacea to addressing the needs of real world problems. The basic function of supervised machine learning methods is to extract and learn complex and hidden features of the input data during the training phase. This learning helps predicting or classifying an output for a previously unseen or unknown input. The hypothesis behind supervised learning approaches is that the presented samples in training data are sufficiently comprehensive to learn and address all the features required to make an appropriate inference for any unseen sample. This is, however, not applicable in recognition tasks such as identity re-identification in the wild, unless we have access to all possible facial features in the world. Thus, one viable solution is a reconfigurable embedding algorithm designed to dynamically update the parameters of a proposed machine learning architecture based on the observed facial images to address identity re-identification.

Identity re-identification allows an individual person to be matched based on images captured from one or more video cameras at different time instants and locations. Identity re-identification can be categorized as a subclass of person re-identification [46]. Unlike person re-identification (which seeks to recognize a person of interest by matching similar attributes of a person in a gallery image or video set), identity re-identification focuses on matching facial images captured from different cameras in different poses, illumination, and environmental variations (e.g. sunny, rainy, foggy, and snowy weather during a day or at night). While a number of face recognition approaches based on deep learning have been proposed in the literature [106, 135, 155, 169], they are generally far from perfection for real-word applications, especially in security verification scenarios, as demonstrated in a recent study using the public benchmark comprising one million faces [73]. Face recognition models [135, 166] are generally designed to increase identification [155] and verification [165] accuracy in large datasets. Identity re-identification narrows down the problem to a much smaller dataset, typically a security coverage area, but with a near perfect accuracy rate. This has applications in real-time emergencies such as trying to identify attackers in a crowd.

Thus, embedding feature vector approaches designed for identity re-identification may not be applicable in face recognition tasks. In addition, identity re-identification does not deal with the challenges associated with aging facial features [90]. On the other hand, facial natural variations such as different expressions [34], scales [61], poses [27], occlusions [170], makeups [178], and illuminations [168], that are not usually addressed by face recognition tasks compound the challenges in identity re-identification.

In this paper, we propose a re-identification algorithm that takes into account human biometric parameters such as facial features. The proposed algorithm is designed to facilitate surveillance of wide areas (see Figure 3.1), such as airports [19] and sport events [175], where there are disjoint networks of cameras with different views. Disjoint cameras make identity re-identification a challenging problem [110], due to changes in orientation, distance and illumination that modify the perceived appearance of an identity across cameras. The identity re-identification algorithm presented in this paper uses the enhanced residual network [53], ResFace, to generate facial embedding information. Such embedding can be used to find an identity in the hundreds of thousands of faces captured on surveillance cameras. We also design a vectorized-l2-loss for the embedding system, in addition to center loss [165] to increase the discriminative ability of the residual network. For optimized performance, the face detection and embedding procedure is implemented in multiple steps in a pipeline consisting of multiple neural networks. ResFace is the last neural network in the DNNs pipeline.

In addition, the process of the face-identification is being distributed to the camera (with an attached NVIDIA Jetson TX2 embedded system), fog computing, and cloud environment. An important feature of fog computing [39] is to provide high-quality implementation of the re-identification and matching tasks, as well as dissemination of embedding vectors into cloud. The proposed algorithm can be extended as an application to be scheduled in the cloud and implemented in heterogeneous cloud environment [100].

We will present related literature in the next section. In Section 3.3, we describe our proposed approach. We then describe our training data in Section 3.4, prior to presenting the evaluation of

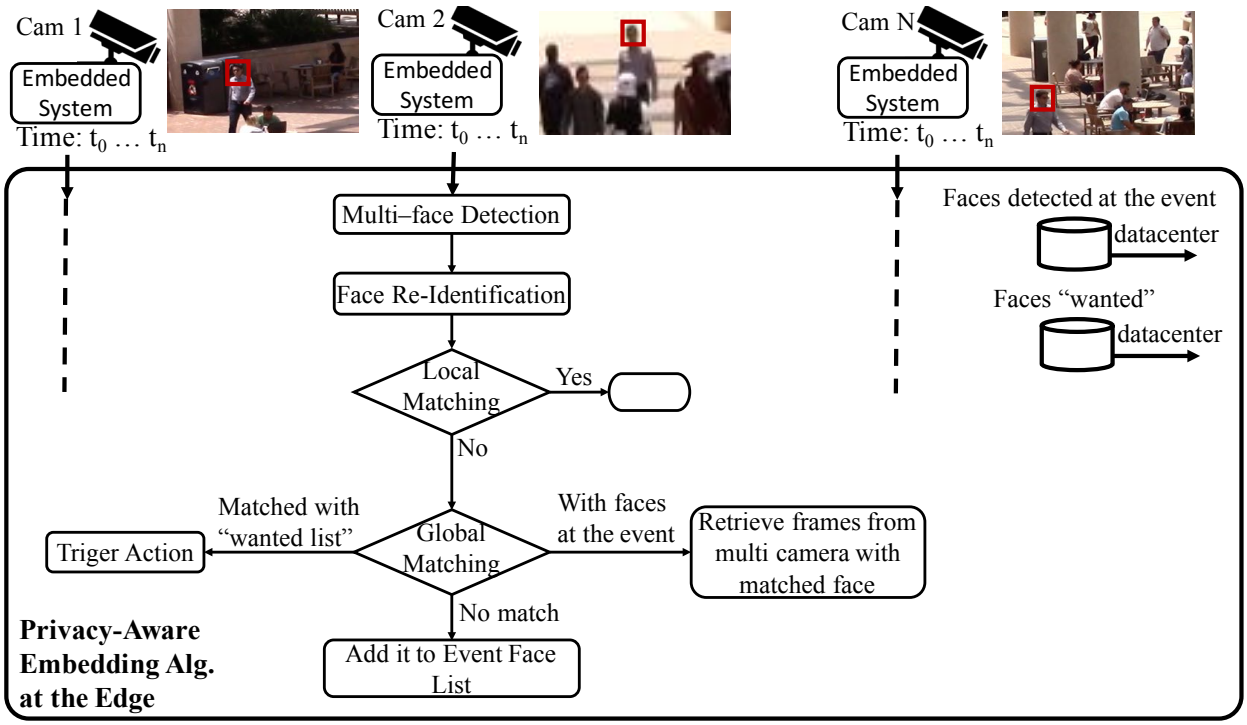


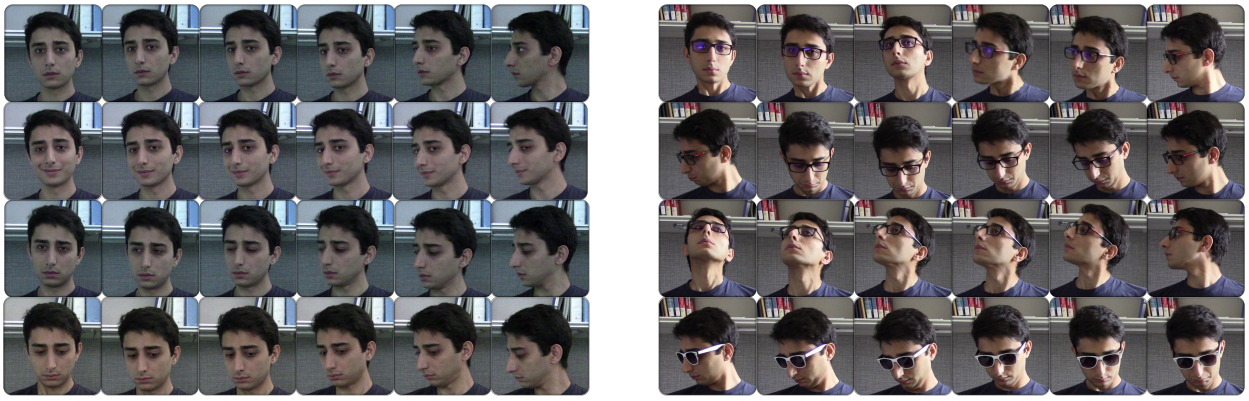
Figure 3.1: Schematic illustration of the proposed face embedding algorithm for identity re-identification using facial features.

the approach in Section 3.5. The last section concludes this paper.

3.2 Related Literature

Over the last decade, hundreds of thousands of images have been uploaded to the internet every minute through various online social networks and photo sharing platforms. For instance, Google reports 24 billion selfies were uploaded to Google's Photos application in 2015, and the number of selfies posted on Instagram increased by 900 times between 2012 and 2014 [142]. As of October 4th, 2017, the number of hours of video viewed on YouTube daily is reportedly one billion hours, with over one billion users (`https://www.youtube.com/yt/about/press/`).

In supervised learning, especially for computer vision tasks, the significant volume of data is precisely what is required in the model's training phase. However, when dealing with such a large training set volume, conventional machine-learning algorithms (e.g. Principal Component Analysis - PCA, and Super Vector Machine - SVM) tend to saturate. A number of approaches have



(a) The set of face images in different expressions and pose, increasing yaw angle from left to right

(b) The set of face images wearing glasses in different roll positions the illumination direction is coming from right

Figure 3.2: A set of ten images for one subject, with considerable facial feature variation.

been proposed in the literature to improve the performance of these models, such as smart biometric identification. However, such models have shown to be sensitive to unconstrained settings (e.g. illumination, expression, and aging) that can considerably degrade the performance of the system.

Krizhevsky et al. [76] presented a very large and deep convolutional network, AlexNet. This provides supervised algorithms a platform to deal with large training datasets, as evidenced in the outcome of the ImageNet competition. AlexNet has also been successfully implemented for complex computer vision tasks, such as human pose estimation [158]. In general, deep and wide neural networks perform significantly well under two conditions: when the volume of the training data is immensely large, and scalable computation resources are available.

A key challenge with identity detection is the variation in pose (e.g. yaw and roll), and illumination expression – see Figure 4.2. In the past, neural networks have been applied to a large variety of face recognition tasks, for example, face verification [22], face detection [109], and face alignment [147]. Most of these techniques expect either a single model to learn through massive amounts of training data, or a function that maps the input patterns into a target space and the norm of the output would be considered as the semantic distance in the input space [22]. In the last few years, there have been renewed interest in face recognition [92, 137, 146, 158, 160, 164, 166, 172, 174]. In [155], for example, a conventional neural network (CNN) based architecture approach by applying several locally connected layers without weight sharing was proposed. In [164], the authors

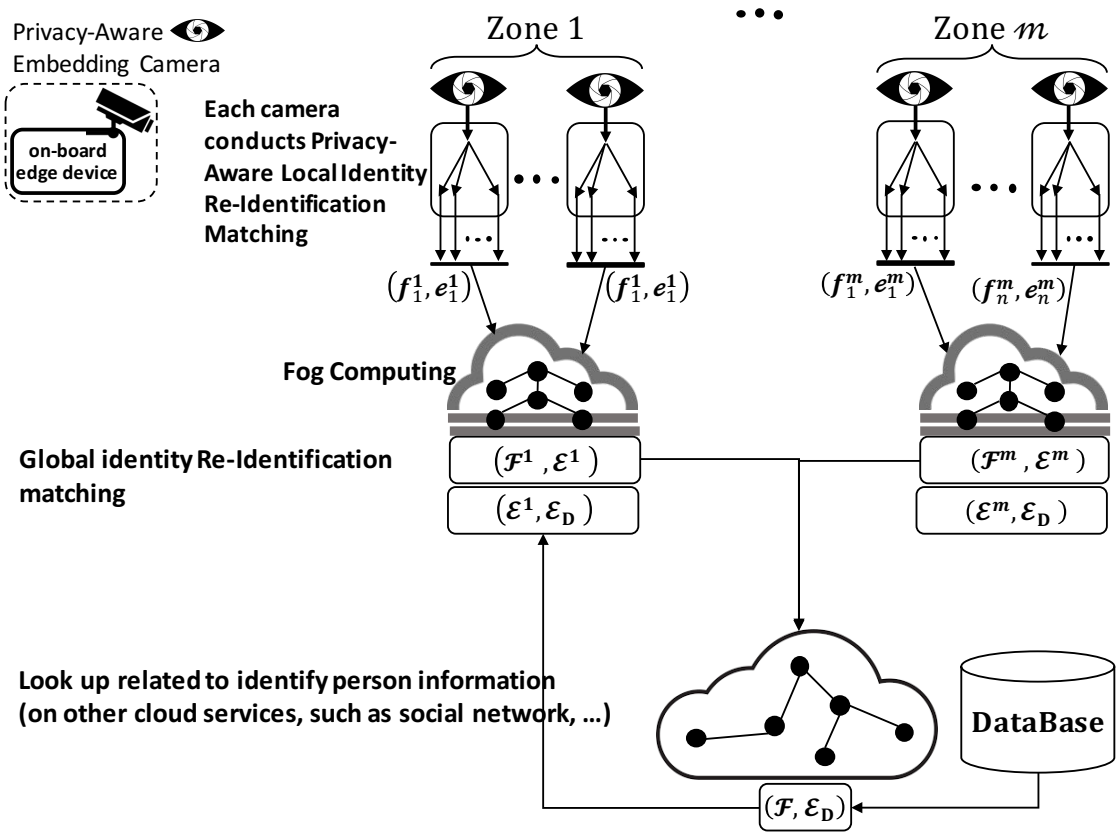


Figure 3.3: Layered structure of an AI-Embedding camera network, where the layers are determined based on the capabilities posed by the devices, fog computing, and cloud computing.

proposed a joint identification-verification supervision function that results in discriminative features. In [149], supervision enhancement is achieved by manipulating the standard CNN structure, where fully connected layers and loss functions are added to each convolution layer. Investigations were carried on the triplet loss in [135], which uses deep embedding to minimize the distance between face images of the same person while maximizing the distance of non relevant face images.

3.3 Proposed Architecture

In this section, we will describe our approach in the crowd identity re-identification tasks, namely: face detection, face embedding, local matching, and global matching. Algorithm 1 describes the privacy-aware embedding algorithm, and Figure 3.3 illustrates the implementation of the algorithm.

The goal is defined as generating the embedding tensor $\mathcal{E}$ consisting of all embedding vectors

of the captured faces. Embedding vector of a face is used for two purposes. First, it acts as the unique identifier of a face and tracks the person if there is a match to the person(s) of interest in $\mathcal{E}_{\mathcal{D}}$. Second, aggregation of the embedding vectors forms the embedding tensor, which is used for fine tuning the identification model.

The process is carried on three layers, namely: AI-embedding camera (or intelligent IoT device), edge (or fog computing), and cloud computing (as a GPU-enabled platform) – see Algorithm 1. Tasks are distributed in decentralized and centralized coupled jobs, and each layer is responsible for specific jobs. However, hyper-parameters in each layer are influenced by other layers to increase the performance.

Video streams from a camera is processed at the camera location with two jobs, namely: face detection (f_i^m, camera m, face i), and forming the embedding vectors of the extracted faces (e_i^m). At the embedded camera, the generated embedding vectors are stored in the local cache. In the zone within the coverage of the camera, the local matching tracks changes in the stored embedding vectors, and unrecognized ones (i.e. an indication of new identity entries) are reported to the next level at the edge. At the edge, information from multiple cameras are aggregated and corresponding tensors are made (i.e. $\mathcal{F}$ faces tensor, and $\mathcal{E}$ embeddings tensor) with two jobs, namely: preparing data for fine tuning the face identification algorithm, and locating the person of interest in the zone area within the coverage at the edge level. The latter is facilitated using global matching. To minimize the required computations and avoid deteriorating the already trained neural network model, transfer learning is applied as the fine tuning strategy at the GPU-cloud enabled platform. GPU-cloud is enabled as an on-demand provisioning computing resource, and is responsible for the fine tuning of the identification model and updating of the running model in all connected devices. Using the cloud for fine tuning allows us to meet the computational requirements to optimize the model. In addition, transferring the model to the datacenter reduces network latency and thus, increases performance [122, 140].

Face detection is implemented on the AI-embedding camera by leveraging the modified version of the multi task convolutional neural network in [174]. Specifically, for face detection, the authors

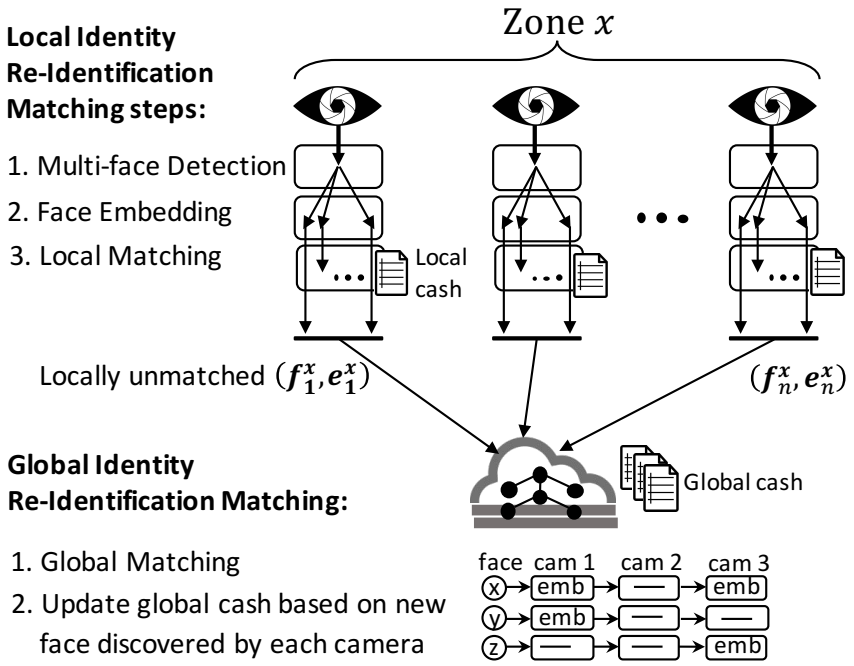


Figure 3.4: Demonstration of the pipeline at the edge and explanation of the modules functionalities.

used three CNNs with different architectures and model hyper-parameters (number of hidden units, filter size, etc) where each CNN carries out a sub-task. However, for resource constrained devices in a typical IoT deployment, we leverage three replicas of single CNN with a shared architecture and model hyper-parameters. The same CNN is trained for different sub-tasks and the obtained weights for each sub-task are stored at the AI-embedding camera separately. The cascade of the three replicas of the CNN architecture are leveraged to perform face detection. The first CNN is MTCNN-1, and the combination of the second and the third CNN forms a network (MTCNN-2). Thus, face detection tasks are spread through MTCNN-1 and MTCNN-2. We also include two additional tasks to MTCNN-1, namely: event classification and image quality recognition. The primary usage of event classification is for training.

Then, face recognition tasks are carried out using the proposed ResFace with the aim of producing unique embedding vector per identity in all images from multiple cameras. The unique embedding vector per identity represents the facial image(s) of the identity captured from one or more cameras. To satisfy the "privacy-aware" functionality, the embedding vector is produced at

the camera level and is only known by the system. From this point onwards, the system stores the embedding vectors as the unique identifier of the faces in the crowd instead of storing entire face images; thus, preserving the privacy of individuals. Furthermore, we present a distributed caching system for sharing facial embedding information across multiple cameras to facilitate identity re-identification.

3.3.1 DNN Pipeline Conducted at Cameras

The overall pipeline at the intelligent camera devices is shown in Figure 3.4. Given an image (i.e. input of the following stages), we will extract the most critical information into the output dictionary. Once this dictionary is produced, crowd recognition tasks can be easily implemented using standard content based video retrieval (CBVR) techniques, such as those using genetic algorithm and fuzzy logic [12].

MTCNN-1: We exploit a fully convolutional network to explore the event class (e.g. does the scene include faces), and the report for the image ambiguity. Event class directly maps to the final dictionary except for face validation, and image ambiguity is used for face identification and enhances the efficiency of the training. In a real world application, the report of image ambiguity is useful, as it can be used to inform the decision as to whether request for further image processing. If the content of the frame includes faces, then the frame is forwarded to MTCNN-2 for multiple faces detection task in crowd.

MTCNN-2: To obtain the facial candidate windows and the corresponding bounding boxes, two cascaded CNNs are presented. The former provides candidates and the latter refines them and presents the selected candidate to the face's attribute detectors. At the end of each CNN, non-maximum suppression is applied to merge highly overlapped candidates. The second CNN has significant supervision with the help of facial landmarks' positions. This approach is suggested in [174] as well.

ResFace: We propose a deep leaning model based on residual convolution blocks for face recognition tasks, namely: verification and identification. The characteristics of residual blocks,

Algorithm 1 Identity Re-Identification algorithm

Input: $\boldsymbol{C^m} = \{c_1, c_2, c_3, \dots, c_n\}$
c_i video streaming on $i$camera
n number of cameras at zone m

Output: $\boldsymbol{\mathcal{E}} = \{\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_3}, \dots, \boldsymbol{e_n}\}$
$\boldsymbol{\mathcal{E}}$ embedding tensor of the networked cameras
$\boldsymbol{e_i}$ is the embedding matrix of the ith camera, $i \leq n$

$\boldsymbol{\mathcal{E}_D}$ **(**embedding tensor of the suspicious targets**)**
at the camera level (on-board edge device):

1: **at each zone,** m:
2: **at each camera in zone** m (c_i in $\boldsymbol{C^m}$):
3: $\boldsymbol{f_i^m} = \textbf{FaceDetection}(c_i)$ **(** $\boldsymbol{f_i^m} = \{\boldsymbol{\not{f}_1}, \boldsymbol{\not{f}_2}, \boldsymbol{\not{f}_3}, \dots\}$, detected faces**)**
4: $\boldsymbol{e_i^m} = \textbf{ResFace}(\boldsymbol{f_i^m})$ **(**$\boldsymbol{e_i^m} = \{e_1, e_2, e_3, \dots\}$, e_j embedding vector of $\not{f}_j$ **)**
5: **Local Matching**$(\boldsymbol{e_i^m})$
6: if match with local cashing
7: done
8: else
9: send $(\not{f}_i^m, e_i^m) \rightarrow$ next edge level
at the next edge level (fog computing):
10: $\boldsymbol{\mathcal{F}^m} = \{\boldsymbol{f_1}, \boldsymbol{f_2}, \boldsymbol{f_3}, \dots, \boldsymbol{f_n}\}$ **(**tensor $\boldsymbol{\mathcal{F}^m}$ included of all faces**)**
11: $\boldsymbol{\mathcal{E}^m} = \{\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_3}, \dots, \boldsymbol{e_n}\}$ **(**embedding tensor $\boldsymbol{\mathcal{E}^m}$**)**
12: **Loop** e_i, e_d **(** $e_i \in \boldsymbol{\mathcal{E}^m}$, and $e_d \in \boldsymbol{\mathcal{E}_D}$ **)**
13: **Global Matching**$(\boldsymbol{\mathcal{E}^m})$
14: if $\{e_i, e_d\}$ match **(** look for a suspicious target)
15: trigger an action(s)
16: elseif e_i match global matching
17: retrieve all related frames
18: register e_i for existing $\not{f}_i^m$
19: else (no match)
20: add e_i to the global cashing

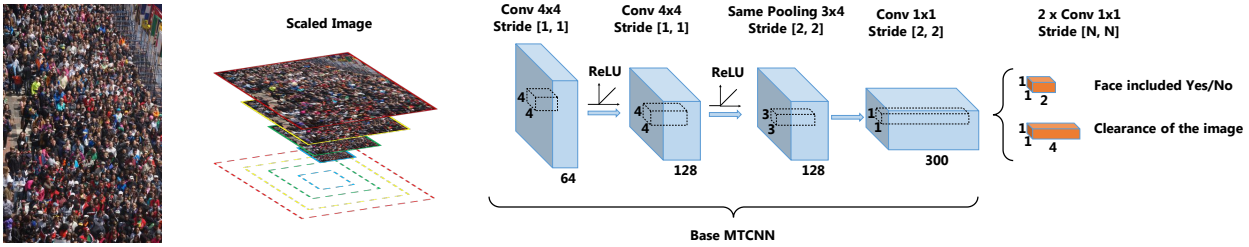


Figure 3.5: Architecture of MTCNN-1, where Conv denotes convolution.

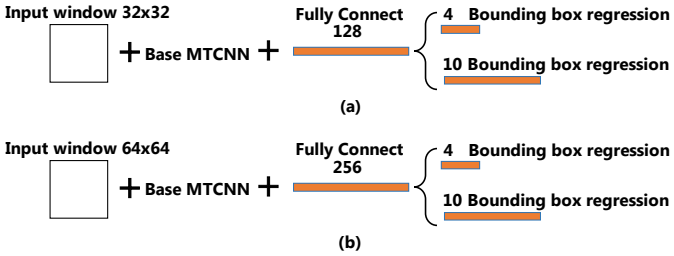


Figure 3.6: Architecture of MTCNN-2, which includes two cascaded CNNs: (a) First CNN, (b) Second CNN.

degradation and overfitting compensation, improve the learning progress. To efficiently enhance the deeply learned features, a very deep residual architecture is leveraged. The combination of center loss and vectorized-l2-loss (defined in Equation 3.2) also improves the discriminative power of the model. For data augmentation, generally for face recognition tasks, the images are randomly horizontally flipped.

3.3.2 DNN Pipeline Architecture and Training

The architecture of MTCNN-1 is depicted in Figure 3.5 and is referred to as the base MTCNN. The network is tuned through experiments. MTCNN-2 has the same base MTCNN architecture but in two separated CNNs, see Figure 3.6. To add more insights in MTCNN-2, the fully connected layer is added before exploring a target's logits. Figure 3.7 presents the results of using the face detection model, cascade of MTCNN-1 and MTCNN-2, on a sample of pictures of a cloud captured by CCTVs prior to the Boston Marathon tragedy in April 2013. In such incidents, face detection and identification are indispensable parts of a person of interest identification system.

The experimental results show that the face detection model, as the pre-requirement, has high accuracy even when target facial images are in different poses (e.g. yaws and rolls). The illustrated

limitations in applying the model on a single frame could be improved when the model has been applied on a set of coming frames. This improvement has significant effects on the next step in face recognition and then re-identification of an individual based on a person's facial images captured from a camera(s) in different conditions.

For face recognition tasks, we use rigid residual comparison [53] – see Figure 3.8. Each residual units start with the filter-expansion layer (1×1 convolution), which is used for scaling up the dimensionality to match the depth of the outputs of the residual branch and the identity mapping branch. This is required to compensate for the dimensionality mismatch induced by the residual unit. So, not only the feature map size of the identity mapping branch is affected, but also the input of the residual units is manipulated to be trained in the plain network. This minor modification allows us to add the feature-extraction functionality to the filter-expansion layer without incurring additional computational cost.

Several versions of residual units are studied, and only one of them is detailed here. Each residual branch consists of three convolution layers, namely: two filter-expansion layers (1×1 convolution) with valid zero-padding, and one middleware 3×3 convolution layer also with zero-padding. All intra-convolutional layers, convolutions inside the residual unit, are followed by ReLU as the activation layer. The convolutional layers mostly have 1×1 filters as the sharp discriminators. The network ends with a global average pooling layer and $N - ways$ (N denotes the number of layers) fully-connected layers. Figure 3.9 presents the ResFace architecture and a sample set of the embedding vectors.

3.3.3 DNN Pipeline Learning Objects and Optimization

To train our DNNs, the combination of multi tasks using l2-loss function is used. For the tasks in {'face classification', 'bounding box regression', 'facial landmark localization'}, we formulate the learning object as the l2-loss function (Euclidean loss) [65] of the targets. L2 loss function

minimizes the squared differences between the estimated and existing target values:

$$L_{batch}^{target} = -\frac{1}{N}\sum_{i}^{N} ||\hat{y}_i^{det} - y_i^{target}||_2^2 \tag{3.1}$$

For each sample n, $\hat{y}_i^{det}$ is the ground-truth label, and $p(y_i)$ is the probability of the target task produced by the network. Also, $\hat{y}_i$ is the ground-truth label, and N is the number of examples in the batch.

Furthermore, for face identification (i.e. not verification), we propose the rigid vectorized-l2-loss. This is the cross entropy on the embedding vector of the target class and the corresponding ground truth labels in the same class and other classes. For each sample n in N, we have:

$$\begin{aligned} H_{\hat{y}}^{n}(y) :=& < \hat{y}^n, y^n > \\ &= -\frac{1}{N}\Big\{ \sum_{i}^{E} \hat{y}_i^n \log(p(y_i^n)) \\ &\quad + \sum_{m(m\neq n)}^{N} \sum_{i}^{E} (\hat{y}_i^m - \hat{y}_i^n)(\hat{y}_i^m - \log(p(y_i^n)))\Big\} \end{aligned} \tag{3.2}$$

In the above equation, E denotes the embedding size. The first term is the conventional cross entropy, designed to minimize the distance between the predicted embedding vector of sample n and the ground truth of sample n. It can be interpreted as the inter class learning. The function of the second term is to maximize the distance between the elements of the embedding vector of sample n belonging to a specific class and other samples belonging to other classes in the same batch. It can be interpreted as the intra-class learning.

The combination of the center loss [165], shown by Equation (3.3), with the proposed vectorized-l2-loss, increases the identity mapping into the embedding. While center loss attempts to find the central point for each class, vectorized-l2-loss adds the additional force on the hyper plane to locate the global minimum.

$$L_C = \frac{1}{2}\sum_{i}^{N} ||y_i - c_i|| \tag{3.3}$$

In the above equation, c_i denotes the y_ith class center of the deep features.

3.3.4 Private Aware Local and Global Distributed Matching

Distributed local matching is carried out in parallel at each camera. The same matching algorithm is conducted at the fog for all collected data from all cameras at the common zone.

The embedding values of the ResNet output are represented in an n-dimensional vector, where n denotes the number of features extracted from the facial image. Due to the environmental noise and different face poses, feature values for an identity are not always deterministic and may vary among captured images. For this reason, multivariate kernel density estimation (MKDE) [13] is used as the stochastic model to incorporate possibilities of embedding vector variations. Given a set of feature vectors e_i, we have:

$$e_i = (e_{i1}, ..., e_{in}) \qquad i = 1, ..., N_e \tag{3.4}$$

$$\hat{p}(\boldsymbol{z}) = \frac{1}{N_e \sigma_j ... \sigma_n} \sum_{i=1}^{N_e} \prod_{i=1}^{n} \kappa \frac{z_j - e_{ij}}{\sigma_j} \tag{3.5}$$

In the above equation, the probability of obtaining an embedding vector $\boldsymbol{z}$ with the same components e_i is defined by $\hat{p}(\boldsymbol{z})$, where N_e is the size of the embedding vector, $\kappa(.)$ denotes the applied Gaussian kernel on all embedding vectors, and σ_j is the standard deviation of the kernels obtained from the empirical results. Kullback-Leibler [79] is defined as the robust and discriminative distance function:

$$D_{KL} = (\hat{p}(\boldsymbol{z}^{t+1}) | \hat{p}(\boldsymbol{z}^t) = \sum \hat{p}^{\boldsymbol{z}^t} \cdot \log \frac{\hat{p}(\boldsymbol{z}^t)}{\hat{p}(\boldsymbol{e}^{t+1})}, \tag{3.6}$$

where $\hat{p}(\boldsymbol{z}^t)$ denotes the observed embedding vector z corresponding to an identity at time t. Maximum D_{KL} is considered to be the tracking of the same identity.

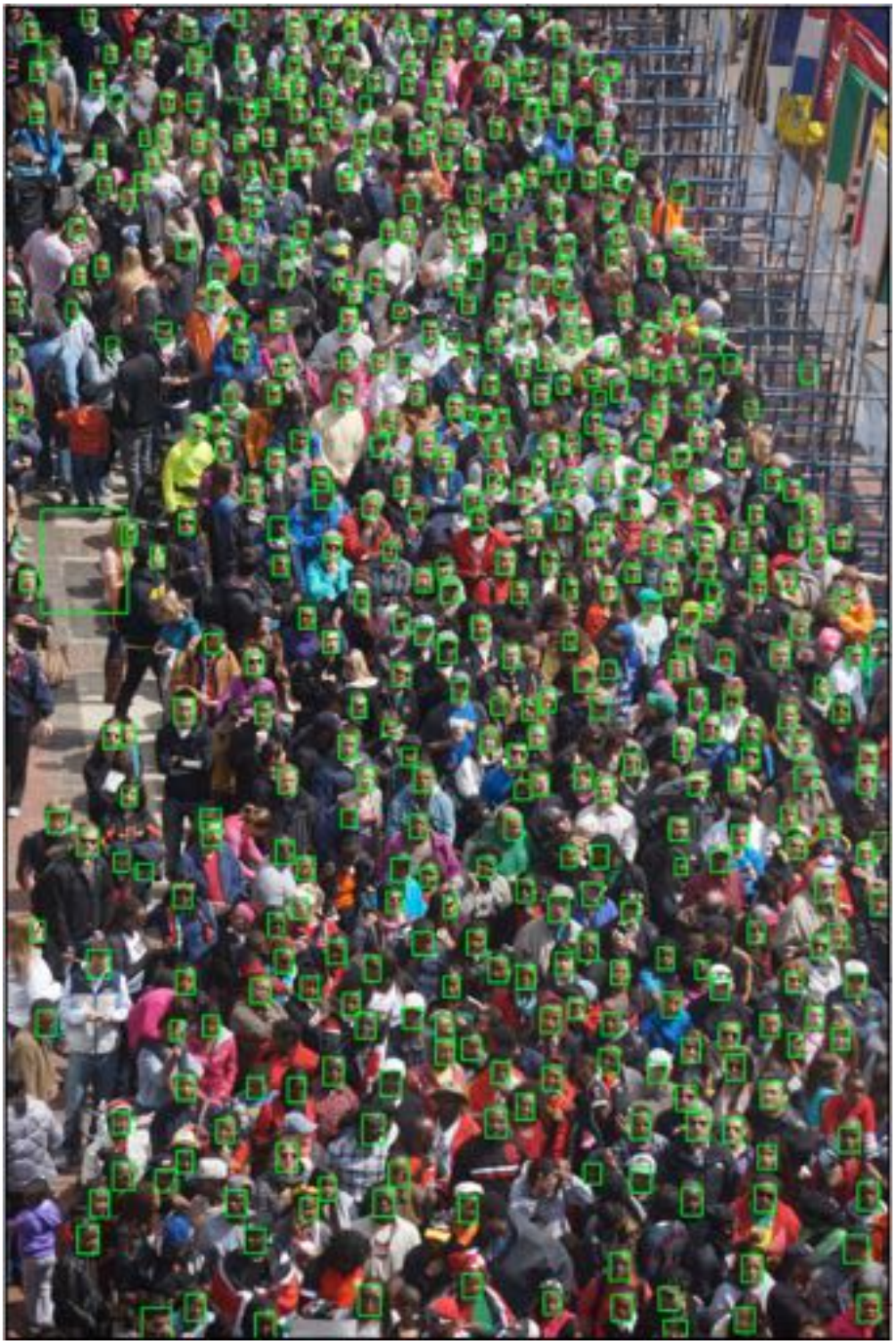
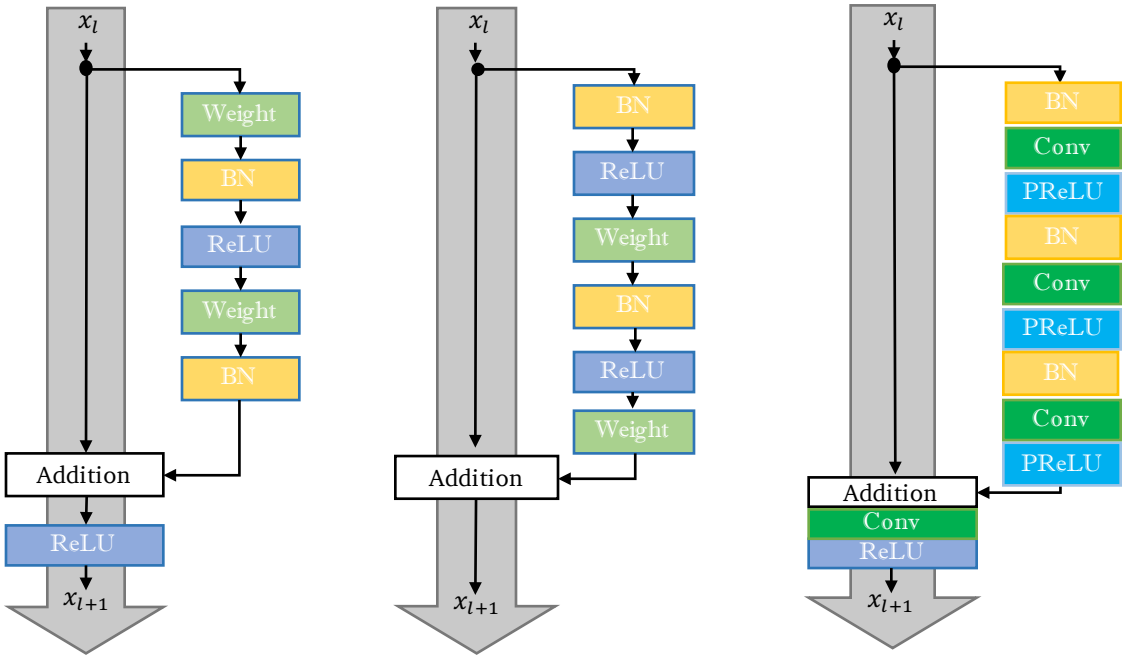


Figure 3.7: Detected 474 out of 523 counted faces in a crowd scene, where the faces are in different poses captured in different resolutions

.



(a) Original ResNet sack (b) Enhanced ResNet stack (c) Proposed ResNet stack

Figure 3.8: Residual block structure: (a) Original residual block. (b) Enhanced residual block for image recognition tasks. (c) Proposed residual block for face recognition tasks.

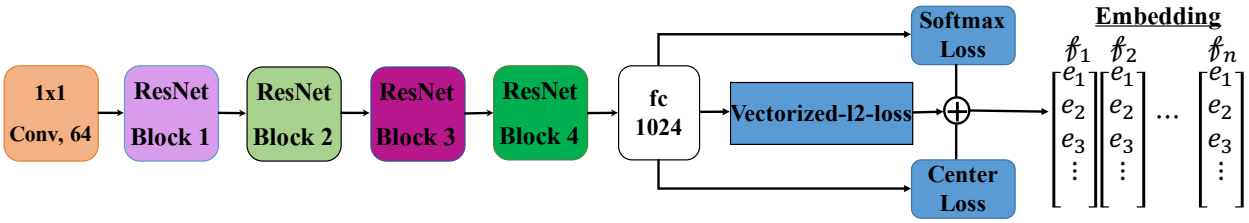


Figure 3.9: ResFace architecture of face recognition.

3.4 Training Data

We collected crowd data from various sources, including images from the web, the WIDER FACE [171], the FacrScrub [108], and the Labeled Faces in the Wild (LFW) [84]. A total of 152,366 images with more than unconstrained 514566 faces were studied. For crowd event categorization, WIDER FACE with 61 classes was used, while for other crowd recognition, the combination of all datasets were considered. Figure 3.12 presents the samples of the studied datasets, and paired similar faces can be either from the same dataset or multiple datasets.

3.5 Evaluation

In this section, we describe our evaluation of the proposed re-identification algorithm on face recognition and video benchmarks in an unconstrained environment.

3.5.1 ResFace Experiments

The experiments involve classifying of facial camera images of various people with varying expression and poses. There is also variation in the person, such as clothing worn by the person, and position of the person's face within the image or video frame. Using our proposed approach, a variety of targets can be learned. For example, given a face image as input, ResFace can be trained to output the identity of the person, the direction of the face, the gender of the person, and whether or not the face has occlusion.

Experiments on the mixed of datasets: We selected the LFW dataset as the first benchmark, which is widely used by other state-of-the-arts models in the literature. A comparative summary of the evaluations is presented in Table 3.1.

For the second benchmark, we used a mixed pool of the aforementioned datasets. We presented the results of three existing well trained state-of-the-arts models and three versions of ResFace (i.e. A, B, and C) implementing the residual blocks in Figure 3.8. Findings from our evaluations demonstrated that although significant achievements are obtained on specific dataset such as LFW, the performance of existing models degrades on the mixed dataset. A combination of the vectorized-l2-loss and center loss resolves this problem by increasing the discriminative power of the ResFace with a 2.6% increase in the recognition task accuracy on the mixed dataset.

Class Samples Dependency: Subsets of LFW and FaseScrub were selected, as shown in Table 3.2. Seventy percent of each subset was used for training and the remaining was allocated for evaluation. We observed that the model does not strictly rely on the number of images per person.

Probing and Memorizing Features: To explore how the model probes the features for a given example and to emphasize the discrimination potential of the model, a poor grayscale face image was selected. As demonstrated by the findings of the evaluations, the same logic can be extended

Table 3.1: Verification performance of different methods on LFW and mixed dataset. ResFace-C-1 was trained using center loss, and ResFace-C-2 was trained using vectorized-l2-loss and center loss.

METHOD	ACC. ON LFW	ACC. ON MIXED DATASET
FACENET	99.5± 0.2	73.7± 0.2
DEEPFACE	97.4± 0.2	72.0± 0.6
RESNET-101	89.2± 0.4	70.8± 0.3
VGG-FACE	97.3± 0.1	71.3± 0.6
RESFACE-A	84.2± 0.3	70.1± 0.6
RESFACE-B	89.7± 0.2	71.0± 0.6
RESFACE-C	91.3± 0.2	71.2± 0.6
RESFACE-C-1	98.3± 0.9	74.3± 0.3
RESFACE-C-2	99.3± 0.9	76.3± 0.3

to other image types, unconstrainted color images, and aligned deep funnel images. Figure 3.10 illustrates the tracking of the activation layers for a sample. From the observation, it is clear that ResFace memorizes the face area and extracts a collection of the face features.

Pose-Aware Recognition: ResFace was used to learn discriminative representations of the different face poses. The model was trained considering random data augmentation, namely: horizontally flipped and 30 degree rotation without normalizing images to a single frontal pose. According to findings presented in Figure 3.12, we had some false detection and correct ones in different poses, yaws and rolls of the faces.

Resolution Limitations: To investigate the minimum facial image resolution in a terms of pixels, face images with resolutions of $\{250 \times 250, 120 \times 128, 60 \times 64, 30 \times 32\}$ were used from the Web-Collected training data. Test epochs comprise subsets of the MegaFace [73], and collected private dataset. The results shown in Figure 3.11 indicate that 30×32 facial image size conveys enough information for targeted facial recognition tasks.

3.5.2 Testing on the DNN Pipeline Constructed at Cameras

The proposed algorithm was implemented following the layered structure, presented in Figure 3.3. The experiment was conducted using a 2k high resolution camera with 20 fps and NVIDIA

Table 3.2: Dependency of the proposed residual model on sample numbers.

min $\frac{\#images}{person}$	max $\frac{\#images}{person}$	ACCURACY ON LFW
14	25	79.91± 0.2
20	34	85.43± 0.2
32	51	90.20± 0.4
45	75	93.27± 0.1
14	75	87.22± 0.3

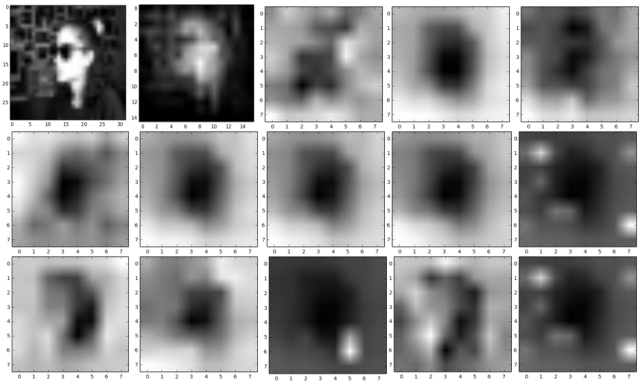


Figure 3.10: Feature extraction and show case memory of the activation layer in ResFace.

Table 3.3: Verification performance of different methods on LFW and mixed dataset.

FACE RECOGNITIONS TASKS	ACCURACY
VERIFICATION	99.5± 0.2
IDENTIFICATION	97.4± 0.2
POSE+DIRECTION+IDENTIFICATION	89.2± 0.4

Jetson TX2 as the embedded system at the camera. After applying MTCNN-1 and predefined target, we proceeded to face extraction. From the detected frames, faces were extracted and fed to MTCNN-2 for face detection and alignment. In the next step, ResFace maps the faces to the embedding for further face recognition tasks (i.e. identification and re-identification) in multi cameras system. After face detection, the frames were sent to the pipeline for more processing (i.e. pre-preparation of the data, face alignment, flipping the images, and illumination enhancement). Pre-trained residual blocks were fed by the sifted data. On average, three people were present in each frame or image. The video recordings were about one minute long, with 20 fps. The recording environment also varied.

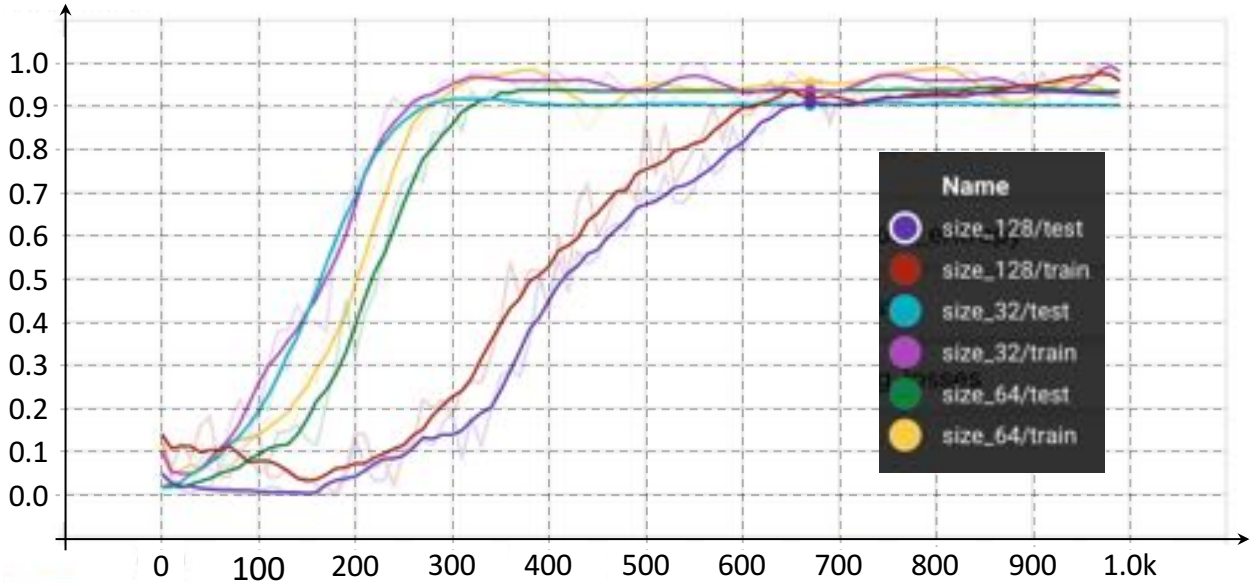


Figure 3.11: Training and testing accuracy on different sizes of input images.

.

Table 3.4 presents the process time of the local identity re-identification matching algorithm, namely: face detection and re-identification (re-identification points to two tasks face embedding and local matching). In our evaluations, a number of faces in a frame is increased to study the processing time of the algorithm. As reported in Table 3.4, the processing time for face detection is stable while the number of detected faces increases to 200 faces per frame. In addition, the processing time of the re-identification stage is a fraction of the processing time for the local identity re-identification matching algorithm. In other words, decreasing the required processing time of the detection stage to less than a millisecond can result in less than a millisecond identity re-identification algorithm at the camera level; thus, viable for real world implementation.

In Figure 3.13, we have a visual observation of the facial embedding vectors for 75 identities in different poses, illuminations, expressions, and with or without occlusions. A 2D t-distributed stochastic neighbor embedding method was then used to illustrate facial embedding vectors of a set of sample identities, 128 element for each face, prior to applying the local matching. As shown in Figure 3.13, different face images of a person were clustered accurately. In other words, embedding vectors are minimizing intra-class variations while discriminating inter-class similarities.

Figure 3.12: Example detections by ResFace in different poses, yaws, and rolls. (a) Correct recognitions. (b) False recognitions

3.6 Conclusion

Increasingly, computing systems that interact with their environment, such as real-time surveillance monitoring service and driverless vehicles, may need to operate autonomously (i.e. without or with minimal human interaction). Such systems can be found in both civilian (e.g. smart cities and Internet of Vehicles) and military settings (e.g. Internet of Battlefield Things or Internet of Military Things).

In this paper, we presented a configurable interactive embedding system to extract the facial

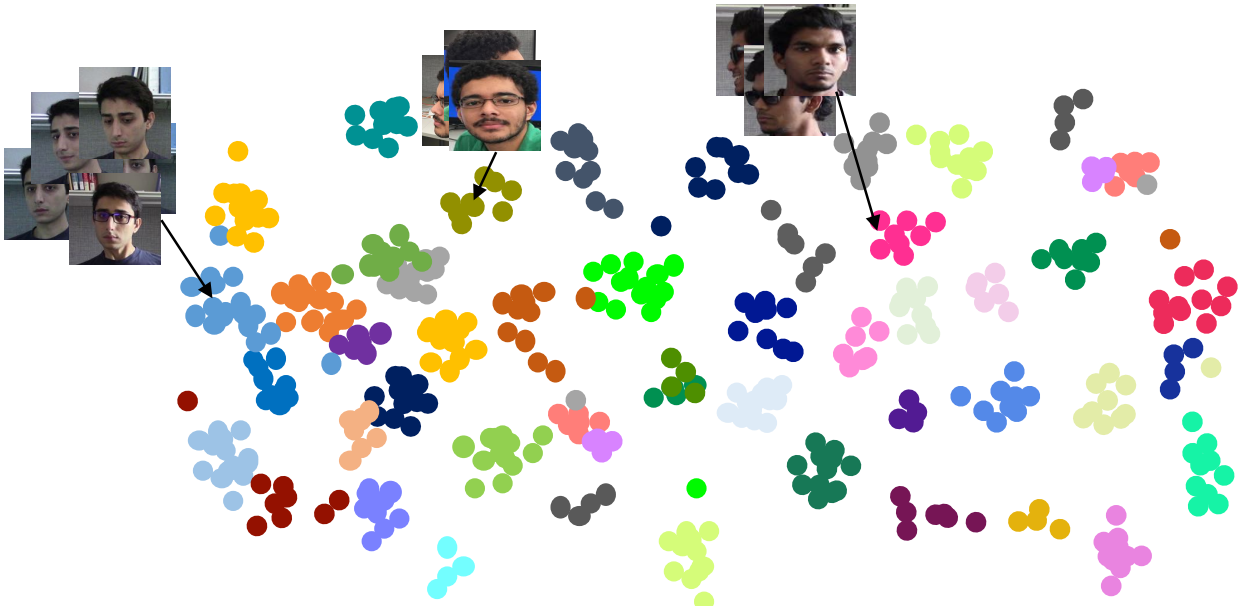


Figure 3.13: A 2-dimensional illustration of the embedding vectors using stochastic neighbor embedding for 75 identities in different poses, illuminations, and expressions.

image features in crowds. The proposed system generates unique facial embedding per person using our privacy-aware ResFace neural network and vectorized-l2-loss function in the training phase. Edge computing is then utilized to facilitate or expedite real-time decision making. Furthermore, the multivariate kernel density estimation (MKDE) matching algorithm is applied to facial embedding pipelines generated by cameras at the edge for identity re-identification as well as security verification tasks. We address potential performance degradation issues associated with face recognition networks by parallelizing and moving the computation to each camera. In addition, we implemented the DNNs pipeline to extract crucial information of the scene for further verification. Potential applications of this system includes places of mass gatherings and key (critical infrastructure) installation sites, which can be used to identify and re-identify persons of interest in a crowd. For example, a person of interest (e.g. suspected suicide bomber) may be carrying a backpack (with a homemade improvised explosive device) in images captured by one camera. The person of interest then placed the backpack in a location where the explosion of the improvised explosive devices is likely to cause the maximum impact, and threw the jacket in a trashbin. In other words, this person may no longer be carrying a backpack or wearing the jacket, and may even

Table 3.4: Processing time of the detection and re-identification tasks at the AI-Embedding camera (in milliseconds).

# OF FACES IN A FRAME	DETECTION IN MILLISECONDS	RE-IDENTIFICATION IN MILLISECONDS
1-10	7.99± 0.23	0.27± 0.05
12-20	7.91± 0.28	0.27± 0.07
21-30	8.06± 0.31	0.26± 0.08
31-40	8.08± 0.24	0.28± 0.06
41-50	8.12± 0.26	0.29± 0.09
51-75	8.11± 0.17	0.27± 0.07
76-100	8.12± 0.27	0.29± 0.08
101-200	9.23± 0.34	0.31± 0.11

sport a different hairstyle (e.g. wearing hair wig). The proposed system will be able to identify this person as the same individual.

Future work includes integrating big forensic data reduction and management techniques such as those proposed in [116, 117] in future design of the system, to facilitate timely pinpointing of relevant evidence and intelligence from heterogeneous distributed systems.

Chapter 4: TEMPORAL FACE EMBEDDING AS BIOMETRIC TOKENIZATION FOR DECENTRALIZED IOT

This chapter is submitted to be published: Miraftabzadeh, Seyed Ali, Paul Rad, and Mo Jamshidi. "Temporal Face Embedding as Biometric Tokenization for Decentralized IoT." In Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference (2017).

With IoT proliferation, a decentralized biometric access control system is crucial to ensure the identity consistency in cyber physical space. In this paper, a biometric secure modality is proposed based on temporal domain that is often available in IoT systems. A novel neural network architecture, named LSTM-ResNet, is designed to learn the temporal dynamics of human biometrics such as face for authentication. The model is designed to implement as a device-embedded temporal face embedding as biometric tokenization to keep data securely decentralized on IoT devices for real-time on-line and off-line identification. The combined novel temporal face embedding, LSTM-ResNet and decentralized embedded IoT approaches prevent tampering and spoofing unauthorized access. The preliminary results are promising for decentralized biometrics IoT and motivate future work in this area.

4.1 Introduction

From a system-level perspective, the IoT authentication can be looked at as a highly dynamic and heterogeneously distributed networked system, composed of a very large number of smart objects producing and consuming information. For instance, mobile communications security, smart vehicles, door locks, connected homes, medical devices and critical infrastructure all part of a growing world of IoT devices. For these devices, an automated recognition of authorized individuals poses a serious technical challenge which needs to be addressed. Such a system should be convenient and guarantees the specific level of accuracy as well as keeping the individuals' data

safe from hackers. These requirements necessitate changes in conventional security modalities methods, and in implementing authentication systems and algorithms.

Security modalities are classified into three categories: (1) possession, what we have as an authentication tools: e.g. key, RFID or ID card. (2) cryptographic, knowledge base authentication: e.g. PIN, password, challenge-response answers like as parents maiden name. (3) biometrics: e.g. fingerprint, face, iris, etc. Obviously, possession base methods are in contrast with the easygoing usage of IoT devices and are not always applicable. IoT authentication requires automated recognition of individuals in many smart connected devices continuously which causes cryptographic methods not enforceable as primary secure modality. However, knowledge base methods can be incorporated in two-factor authentication for some cases through the secure sockets layer protocol. Due to the IoT requirements, biometric is a preliminary candidate to address the issues. Biometrics focus on evaluation, comparison, and statistical analysis of people characteristics. It falls into three main classes: physiological which are related to the shape and size of the body, examples include face. Behavioral which are related to the change in human behavior over time, for instance gait (the way one walks), rhythm of typing keys. Combination of both which includes both traits, where the traits are depending upon physical as well as behavioral changes, for example, voice recognition. It depends on health, size, and shape of vocal cord, nasal cavities, mouth cavity, shape of lips, etc. and the emotional status, age, illness (behavior) of a person. A secure and dynamic biometric modality is proposed based on physical and behavioral changes of facial images over time. The proposed method is different from common face recognition models designed for identification and verification tasks in main viewpoints. The face recognition task can be done in a number of various ways, such as by capturing a facial image using an optical camera (in the visible spectrum) or by processing the facial heat emission captured by an infrared camera. In the visible spectrum, face recognition tasks focus on modeling features from the central portion of the facial images that do not change over time while avoiding superficial features such as facial expressions or hair. Face recognition models are generally designed to increase identification and verification accuracy in large datasets. However, proposed model relies on dynamic physical and behavioral characteristics

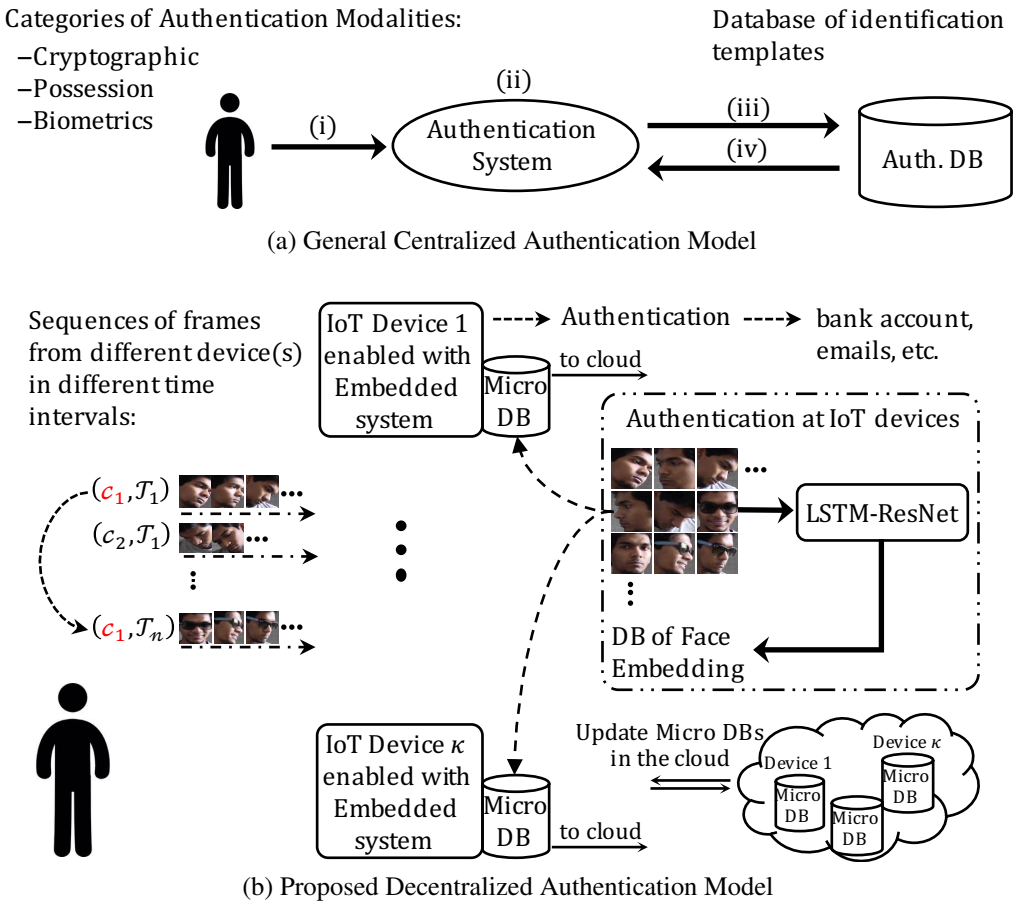
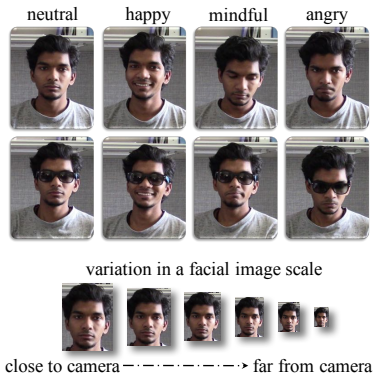


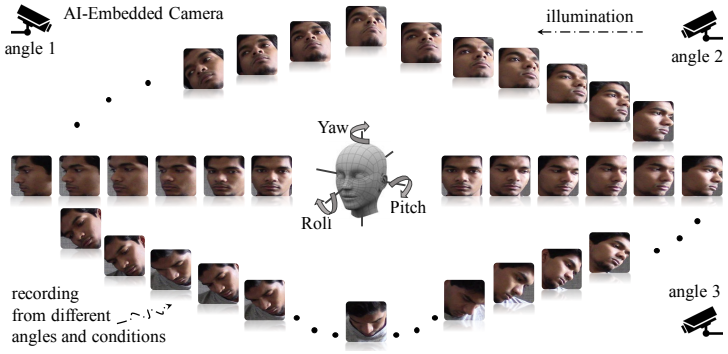
Figure 4.1: Centralized authentication and overview of proposed decentralized authentication.

of a person facial images over time. It process the central portion of the person facial images in a sequence of frames coming from one or multiple cameras. It considers the face variations in different expressions in an uncontrolled environment, and focuses on an application for a typical security coverage area with a near perfect accuracy rate. Notwithstanding, the proposed model can be extended for smart city as a distributed model.

A centralized biometric system is essentially a pattern recognition process in which (i) a raw data from an individual is acquired, (ii) a notable feature set from the raw data are extracted, (iii) and then new extracted feature set is evaluated against the feature sets stored in the database, (iv) finally a command to execute an action according to the result of the comparison is transmitted by a center located not at the IoT device. The similar process is applied for other secure modalities



(a) Sample of facial images of an individual in different expressions and scales



(b) Captures images of the same person in different views. The illumination direction is from right to left

Figure 4.2: Possibility variation of an individual's facial images for the authentication task

as well, see figure 4.1a. To maintain privacy and eliminate the inconvenience of storing sensitive biometric data, the proposed model is implemented as a device-embedded biometric authentication system, see figure 4.1b.

Device-embedded biometric authentication keeps data securely decentralized on local IoT devices and safe from hackers. In addition, it improves the security of endpoint IoT devices; all the process, feature extraction and comparison software built into the IoT device itself. Since the device essentially belongs to a person or specific group of people, the typical usage is a one-to-one comparison of a probe biometric sample against the group biometric reference stored on the device, see figure refdecentralized.

In the visible spectrum, several approaches to modeling facial images are Local Feature Analysis, Elastic Graph Theory, Principal Component Analysis, Multi-Resolution Analysis, and Neural Networks. The authentication problem can be translated to a binary classification problem, giving input faces the answer is yes or no for each face. A new neural network architecture is proposed to learn temporal features captured from a sequence of frames to classify faces' dynamic structure. The network learns a global interpretation of a face' temporal evolution over time from a camera or a groups of cameras. The network, consist two sub-networks: RNN-LSTM and ResNet.

To learn hidden patterns in time sequence, the proposed architecture drives a benefit from

a recurrent neural network (RNN) with Long Short Term Memory (LSTM) units. RNN has a drawback of vanishing and exploding gradients called Vanishing Gradient Problem, which makes it inefficient for discovering long-term temporal relationships from the input sequences. Thus, the RNN is implemented by leveraging enabled memory LSTM units to store the information for long-term temporal learning. To extract dynamic facial features, residual neural network, ResNet, is exploited as a discriminative learning neural network which also addresses the vanishing gradient problem in the optimization phase. For each coming frame, one LSTM unit stack above ResNet as a top layer before the final softmax layer. Therefore, the architecture leverages the local and dense property from residual operation and learn long-term temporal structure by supplying information in each LSTM units. Our contributions can be summarized as follows:

1. A biometric secure modality is proposed based on person dynamic facial features changes over time instead of face recognition task for authentication.

2. Novel LSTM-ResNet architecture is leveraged to learn the long-term temporal structure from a sequence of frames captured by a camera(s), and then classifying faces' dynamic structure for authentication task.

3. The model is designed to implement as a device-embedded biometric authentication to keep data securely decentralized on local devices and safe from hackers. In addition, preserve users privacy and eliminate the inconvenience of storing sensitive biometric data.

4.2 Related Works

Face identification is turning into a vital research subject, because of its extensive variety of conceivable applications, similar to security get to control, demonstrate based video coding, content-based video ordering, or propelled human and PC collaboration. It is additionally a required preparatory stride to face acknowledgment and appearance examination.

Various methodologies for face identification have been proposed in the most recent decade, a significant number of them depicted and analyzed in two intriguing late reviews by Learned-

Miller [83], and Zhang et al. [173]. Most face identification techniques depend on neighbourhood facial element recognition and order utilizing factual and geometric models of the human face. Low level investigation initially manages the division of visual components utilizing picture properties, for example, edges, force, shading, movement, or summed up measures. Different methodologies depend on format coordinating where a few relationship layouts are utilized to identify nearby sub features, considered as inflexible in appearance (eigen features) or deformable. At that point, visual elements are composed into a more worldwide idea of face through facial component and heavenly body investigation utilizing face geometry imperatives.

The principle downside of highlight based methodologies is that either minimal worldwide requirements are connected on the face layout or removed components are essentially impacted by impediments, and changes in face look and perspective. With a specific end goal to deal with troublesome situations where various appearances of changed sizes and postures must be recognized in vigorously jumbled foundations, some propelled picture based example acknowledgment methods have been produced. They keep away from the particular and conceivably off base face displaying by picking up fundamental tenets contained in very factor confront designs from vast preparing sets of face illustrations. They have ended up being exceptionally tolerant to commotion and twists influencing the face designs.

In [33], Erkin et al. used the standard Eigen faces recognition algorithm in order to create the first privacy-preserving biometric face recognition protocol. The protocol is a secure two-party computation protocol which allows only the inquiring side (client) to see the matching value. Hence, only a client is interested in determining whether it has any image in common with server's database, without worrying about the leakage of unnecessary information. The Euclidean distance between the face image feature vectors from the client and server's face image database is computed. The facial image data record that has the smallest distance to the client's input biometric data is returned. An additive homomorphic encryption (AH-Enc) scheme is used on the data exchanged. Only the data exchanged during execution of the protocol are encrypted. However, the data on the server is not encrypted, and so, the server is aware of its biometric database.

A privacy-preserving biometric identification protocol was developed by Blanton and Gasti in 2011 [175]. This protocol used iris codes based on Hamming distance. AH-Enc and garbled circuits were also used with this protocol. The iris reading of the client is compared with an iris image in the database managed by the server. The client is made aware of the comparison results and every record in the database of server, but the contents of the server are not disclosed to the client.

Later, in 2012, a secure outsourcing approach developed by Blanton and Aliasgari [14] outsourced the computations of iris biometric comparison with data records. Two protocols were put forward; single-server setting and multiple-server setting. The first protocol utilizes a predicated-encryption scheme, where the server performs non-interactive computations. The homomorphic encryption is more secure than the predicate encryption. This is why the protocol presents a more secure biometric system, with greater privacy protection. In their work, the protocol is dependent on at least three independent servers. Another drawback is that, at the end of the protocol execution, the server detects which encrypted biometric data record matches the client's input. This access pattern leakage can breach the security guarantee of the underlying encryption scheme. our scheme does not disclose any information; therefore, it offers the same security protection as the encryption scheme used to encrypt the outsourced biometric image.

4.3 Method

To include all the facial image variations, see figure 4.2, a sequence of video frames $(c_1, c_2, ..., c_n)$ is considered as an input and the output of the network is a binary number y authorizing the user in multiple devices connected to the IoT, see figure 4.3. We propose a residual network with a LSTM layer on top of that to extract intra-class similarity and inter-class discriminatory of captured facial images from different video frames; in other words, the conditional probability of the output, $p(y|(c_1, c_2, ..., c_n))$.

4.3.1 Extract Temporal Features as an Embedding Vector

Temporal feature of a facial image in a frame is presented as an embedding vector. The embedding vector per identity is constructed through the residual network architecture consisting residual blocks. The general form of each block can be formulated as:

$$\begin{aligned} y_l &= h(x_l) + F(x_l, (W_r, b_r)_l) \\ x_{(l+1)} &= f(y_l) \end{aligned} \tag{4.1}$$

where x_l and x_{l+1} are input and output of the lth unit, h is a forward function of the plain unit, F is a residual function and r stands for the number of repeated convolution layer in the residual function, and f is a differentiable threshold function. Figure 4.5 (b) demonstrates an example of the general form of the residual unit. The initial idea of ResNet is to achieve additive residual function F with respect to $h(x_l)$ and facilitates minimizing the loss function. In this regards, [53, 54] emphasize on the importance of the identity mapping, $h(x_l) = x_l$, so in the general formula we denote on r to represent the repetition times of the convolutional layers in residual branch and we follow the identity mapping for the plain branch. In residual block, the other noteworthy nob is differentiable threshold function. If f is also considered identify mapping, for any deeper unit L and shallower unit l:

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, (W_r, b_r)_i) \tag{4.2}$$

This assumption turns the matrix-vector products, say:

$$x_L = \prod_{i=0}^{L-1} W_i x_0, b_i = 0 \tag{4.3}$$

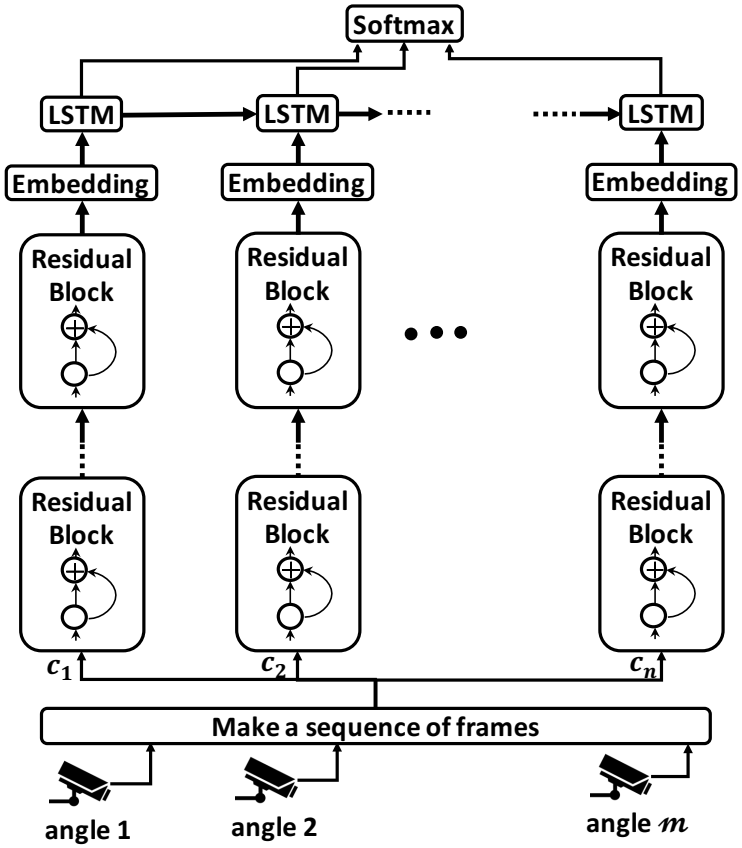


Figure 4.3: LSTM-ResNet architecture unrolled in time for m cameras and n frames, $n > m$.

to the summation of the outputs of all preceding residual functions (plus x_0) [54], and consequently clean backpropagation formula:

$$\begin{aligned}\frac{\partial E}{\partial x_l} &= \frac{\partial E}{\partial x_L}\frac{\partial x_L}{\partial x_l} \\ &= \frac{\partial E}{\partial x_L}\left[1 + \frac{\partial \sum_{i=l}^{L-1} F(x_i, (W_r, b_r)_i)}{\partial x_L}\right]\end{aligned} \tag{4.4}$$

one of the most interesting properties of this architecture is reducing the probability for the gradient to be canceled out. Refer back to the general form of the residual units, there are other residual units with the properties of increasing dimensions and reducing feature map sizes [53,152] by using the conventional activation function, Rectified Linear Unit (ReLU), as the differentiable threshold function:

$$\frac{\partial E}{\partial x_l} = \frac{\partial E}{\partial x_L}\left[\frac{\partial x_L}{\partial h}\frac{\partial h}{\partial x_l} + \frac{\partial \sum_{i=l}^{L-1} F(x_i, (W_r, b_r)_i)}{\partial x_L}\right] \tag{4.5}$$

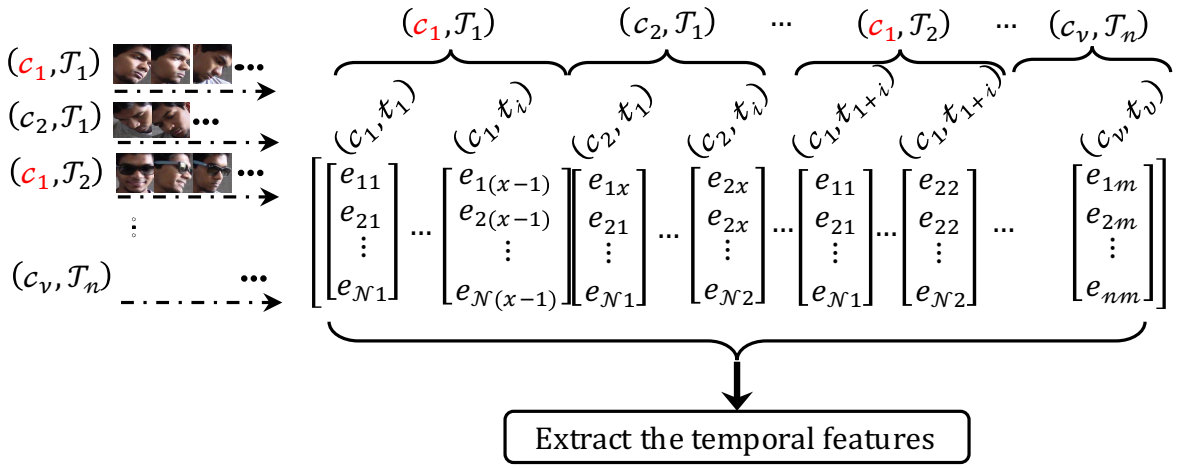


Figure 4.4: Illustration of mapping facial images captured from different cameras to the embedding vectors

The last residual block maps a facial image into the embedding vector. Figure 4.4 illustrates such a mapping for different facial images captured from a camera(s) in different angles and time windows.

4.3.2 Exploring Temporal Relation in a Sequence of Embeddings

The output of the embedding vector is feed to the LSTM unit which is the modified version described in [41]. LSTM units [56] have the ability to learn long range dependency from the input sequences. At the time step t, the behavior between input(x_t), output(h_t), and internal state is controlled through three gates. For each unit c_t stores the internal state and three gates are input gate(i_t), output gate(o_t), and forget gate(f_t). where W and b are model parameters, σ is sigmoid function and g_t is the non-linear transformation of inputs, see Figure 4.6. To capture the temporal relation from the video frames sequence which is importance for identity authentication, outputs and cell memories from last time step are connected to the three gates through defined dot products in:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{4.6}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{4.7}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \tag{4.8}$$

$$g_t = \text{PReLU}(W_{xg}x_t + W_{hg}h_{t-1} + b_g) \tag{4.9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{4.10}$$

$$h_t = o_t \odot \text{PReLU}(c_t) \tag{4.11}$$

inputs of the three gates consist of the current time step of the input and last time step of the output and internal memory. The cell memory is updates as a result of the combination of input gate(i_t) and forget gate(f_t), Equation 4.10. The influence of the input in the internal state is controlled by input gate and forget gate takes the control over the contribution of the last internal state to the current internal state.

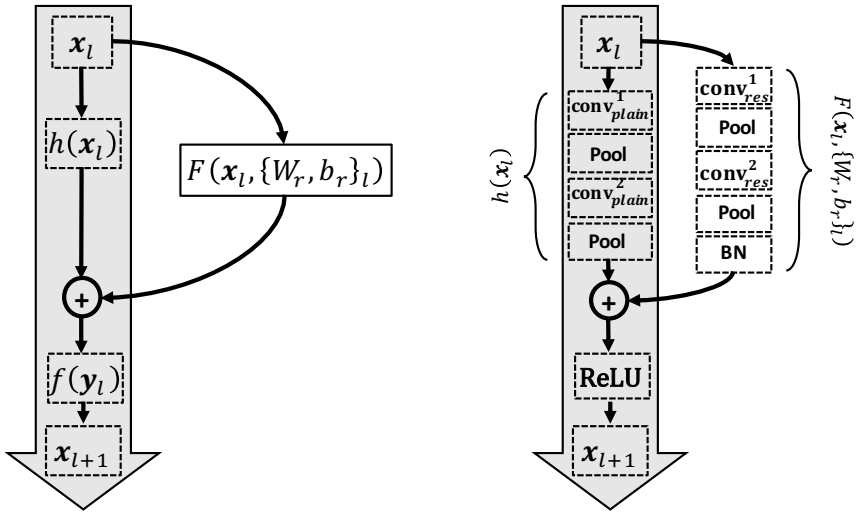
4.3.3 Architecture

In general, the experiment results show combination of residual architecture and embedding address the over-fitting and degradation problems. The augmented LSTM units with PReLU can learn the fine distinction between sequences of nonlinear periodic embeddings patterns without external resets or teacher forcing.

The implementation for residual network follows the practice in [53, 54]. To address the different scales of the facial images, a pyramid scale of each sample is crated the face images with a resolution in $\{250 \times 250, 120 \times 128, 60 \times 64, 30 \times 32\}$ are used and he maximum mini-batch sizes are {32, 64, 128, 128, 324} respectively.

4.4 Training Methodology

The LSTM_ResNet model is implemented and optimized by using TensorFlow, the open-source software library for machine intelligence [1]. The residual network and LSTM units are trained separately on advanced NVIDIA Tesla P100 GPUs with 5.3 TeraFLOPS built for data center. The test is carried on embedded system, NVIDIA Jetson TX2.



(a) (b)

Figure 4.5: Left: (a) General form description of the residual unit. Right: (b) illustration of the residual unit including: residual branch consists of two convolution layers each following by pooling and Batch Normalization at the end, and plain branch with two convolution layers follow by the pooling.

Different residual network architectures are trained from scratch while sharing almost the same optimization strategy and parameters. In the training phase, precise investigation of the effects of different stochastic gradient descent (SGD) algorithms are conduced: Adaptive Subgradient [29], RMSProp [156], Adam [75], and Momentum [151]. It is observed that Adam optimization algorithm outperforms the other methods because of the bias-correction characteristic of the algorithm. The best performance were achieved with $\beta_1 = 0.96$, $\beta_2 = 0.\overline{9}$, and $\epsilon = 91$. For several epoch iterations the learning rate is set to 0.001 and it is reduced by a factor of 10 to stabilize the optimization. $L2-$norm with a weight decay of 0.1 is applied as an artificial constraint to implicitly reduce the number of free parameters and not to make the network difficult to optimize. Since the network is trained for the specific group of authorized users, it is avoid to exploit dropout as the regularization method.

LSTM units are trained following the practice in [41] using stochastic gradient decent (SGD) and a momentum of 0.9 with the truncated BPTT. To capable LSTM to learn bridging minimal time lags, the gradient is truncated where the performance degradation is observed.

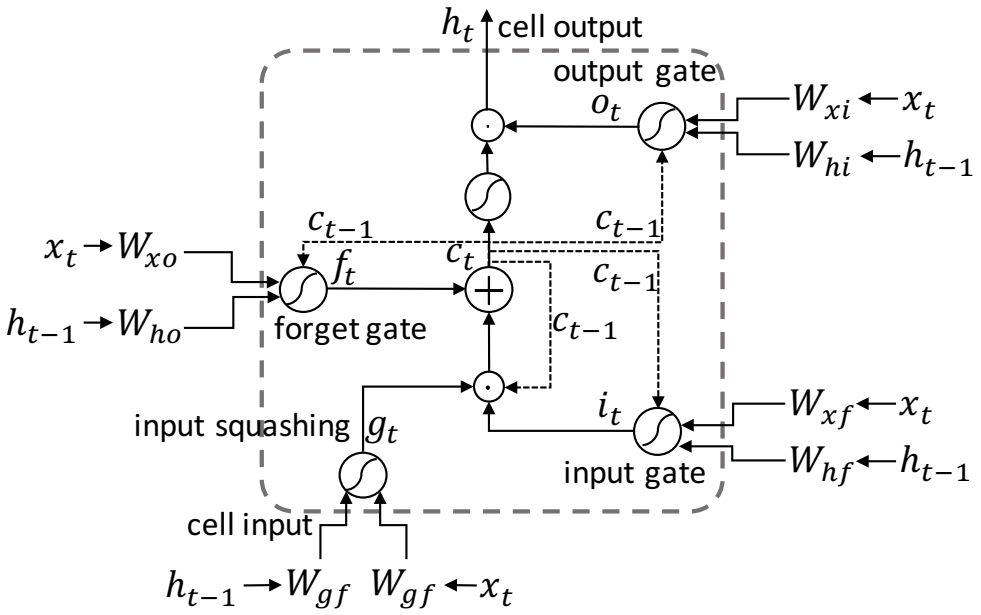


Figure 4.6: LSTM unit illustration. Each circle with a curve means a non-linear transformation. Circle with a dot is element-wise product operation. Square with a plus sign is element-wise plus operation. The dashed arrow means connection from the last time step.

4.5 Experiment

The residual network is trained using FaceScrub dataset [108] which is one of the most accurate face datasets in terms of duplicated, mislabeled, and morphed faces. It comprises a total of 106,863 facial images of 530 celebrities (male and female), with about 200 images per person. As such, it is one of the largest public face databases, with an average of 2.15 images per person. Images are collected from Internet and are taken in uncontrolled conditions (real-world situations). To maximize the usage of the dataset, it is splitted into seven folds; five folds for training, one fold for validation, and two folds for testing. Note that there is no common subjects in the folds. To prevent the unbalance problem, it is tried to make an even ratio of image per person in each fold.

The LSTM units of the LSTM-ResNet model are trained using YTF dataset [166], with no people overlapping with FaceScrub. YTF dataset consists a large variations in pose, expression and illuminations for 1,595 recorded different identities in 3,425 videos (in average 2.15 videos per person). The dataset is made up of divers video durations, 48 frames to 6,070 (in average 181.3 frames per video).

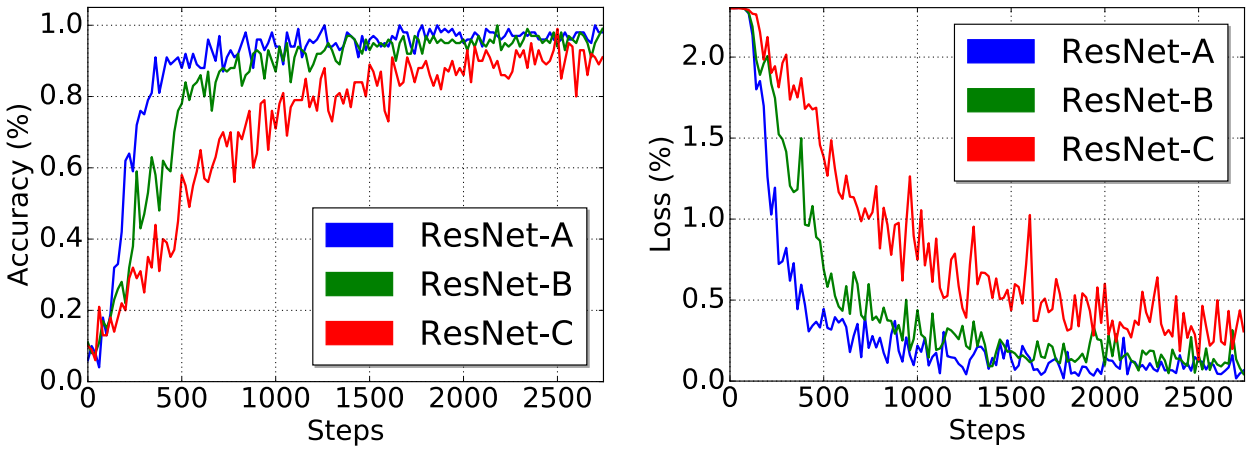
Table 4.1: Different architecture for applied residual network for LSTM-ResNet. Building blocks are shown in brackets, with the numbers of blocks stacked.

Layer name	ResNet-A	ResNet-B	ResNet-C
conv_0	$7 \times 7, 64$ stride 2		
conv_1	$7 \times 7, 128$ stride 1		
block_0	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$
block_1	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$
block_2	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$ $\begin{bmatrix} 1 \times 1, 32 \end{bmatrix} \times 1$	$-$
block_3	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 32 \end{bmatrix} \times 3$	$-$	$-$
Embedding	Average pool, $128 - d$ fc, softmax		

The facial data in the images and the videos varies in size and length, and contains a lot of background noises. To make them ready for LSTM-ResNet model to train, face detection and alignment are the primary steps. Multitask convolutional neural network proposed in [174] is applied on both datasets for the experiment. Cases in which the algorithm fail to detect face locations are eliminated from the experiment process. For data augmentation in the training phase, after cropping face locations images are randomly horizontally flipped, rotated $\{3°, 5°, 7°, \text{or } 10°\}$, and brightness and contrast of the images are manipulated.

4.5.1 Model Details and Results

Three types of residual network architecture are explored, see Table 1. Their practical differences lie in the variation in the number of residual blocks and consequently FLOPS. Depending on



(a) Training accuracy

(b) Learning rate

Figure 4.7: Classification results for training different ResNet architectures on FaceScrub. The deeper network has better training accuracy, and thus training error.

the application, the best model may be different. A model running at the embedded system as the edge device can have many parameters and handle more computation per second, where as the model running on a mobile phone has restrictions in a term of memory capacity and device battery life. Suggested approach in [91] is followed and feature extraction is performed by adding $1 \times 1 \times d$ convolutional layers rather than playing with different filter sizes. Dimension matching in connections is performed by the sole convolutional layer at the end of each block except the last one connected to the embedding.

On the FaceScrub dataset, the residual networks are trained to learn a feature mapping from facial images to a feature vectors in a compact euclidean space in which $L2-$norm distances present the similarity of faces [135]. To address hard samples problem [135], the residual networks supervised by center loss are trained and optimized by standard stochastic gradient descent instead of the original triplet loss function for such an embedding system. All residual architectures are trained with previously described folds, the average loss of a validation fold in multiple experiments is used to tune the models parameters. The best parameter is exploited to retrain models on the whole training folds and then the final result is reported on the fold test. Figure 4.7a shows the classification accuracy obtained from the training on different permutations of the five training folds. The deeper residual architecture, ResNet-A, was training much faster, see figure 4.7b.

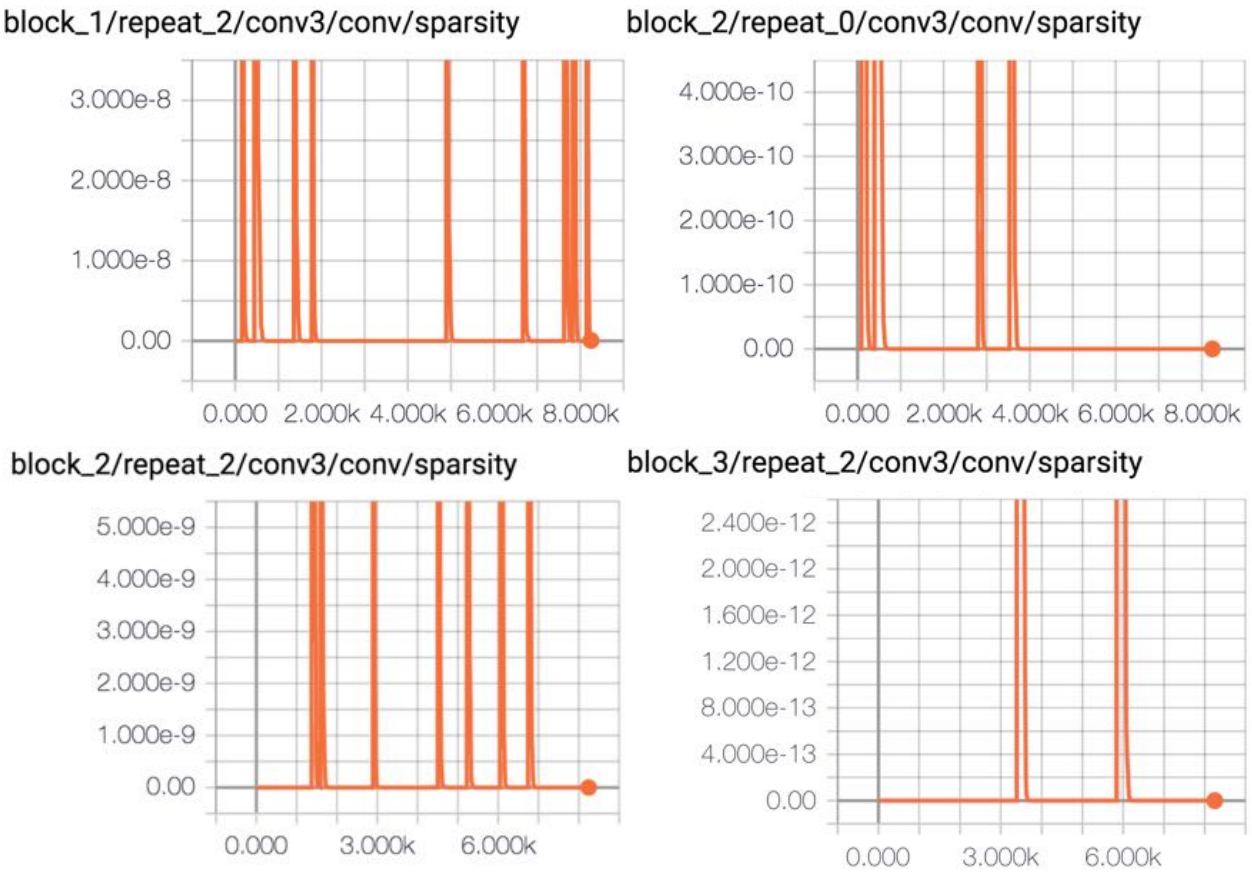


Figure 4.8: Sparsity illustrations of the activation function at the end of each residual block for ResNet-A architecture.

The experiment illustrates the deeper network has better training accuracy, and thus training error. The proposed network is explored and examined with multiple probes to monitor the bias-variance trade-off of the weight vectors, in the training phase. To increase the algorithm performance, the hyper-parameters are fined and tuned to achieve a balance between bias and variance. In turn, the algorithm utilize all variables to learn fair assumptions about the form of the target function while suggesting small changes to the estimate of the target function, see figure 4.8 showing sparsity of the activation layer at the end of each residual block. Table 4.2 presents some comparisons between various versions of residual networks for top-1 error and top-5 error on validation fold and test fold. These major observations affirm ResNet-A reduces the top-1 error by 2.68%, which is a result from the successfully reduced learning error in the training phase, see figure 4.7b. This comparison verifies the effectiveness of residual learning on extremely deep systems. Figure 4.9

Table 4.2: Error rates (%) of single-model results on the FaceScrub validation fold and test fold.

Model	Validation Fold		Test Fold	
	top-1 error	top-5 error	top-1 error	top-5 error
ResNet-C	17.81	5.22	18.03	3.01
ResNet-B	15.37	3.12	16.36	1.88
ResNet-A	15.13	1.97	16.12	1.23

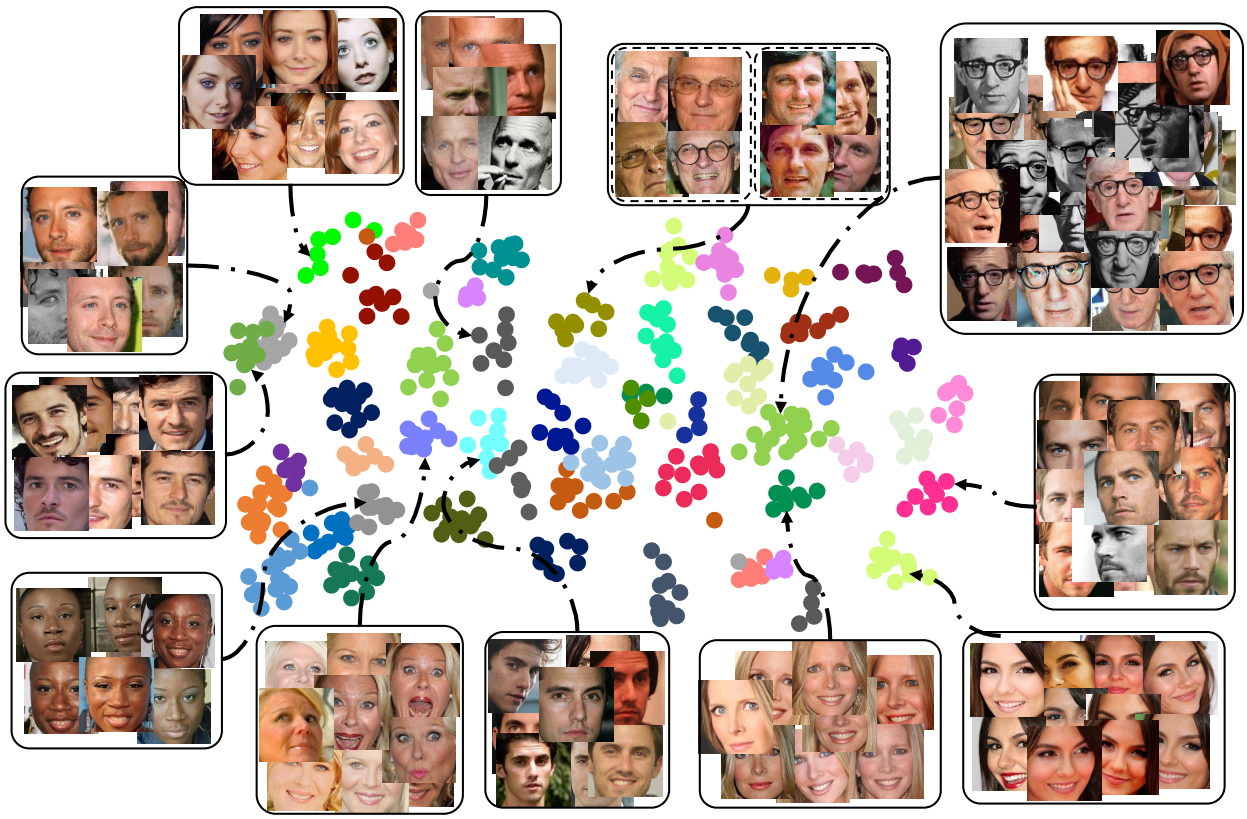


Figure 4.9: Sparsity illustrations of the activation function at the end of each residual block for ResNet-A architecture.

illustrates a classification result of the $128-d$ embedding for ResNet-A architecture after applying $2D$ principal component analysis.

The basic result of the LSTM-ResNet architecture on YTF dataset is summarized in table 4.3. The result indicates that LSTM-ResNet architecture can learn a discriminant representation of the input facial images over a sequence of frames. Two protocols are introduced in table 4.3, large protocol which indicates that the related method is trained using large datasets in sophisticated efforts,

Table 4.3: Performance of different methods on YTF datasets.

Method	Protocol	Acc. on YTF
DeepFace [155]	Large	91.4%
FaceNet [135]	Large	95.1%
Deep FR [112]	Large	97.3%
ResNet-A	Small	84.4%
LSTM-ResNet-A	Small	94.7%

and small protocol indicating that small training dataset is used for the related method. Note that the proposed approach focuses on small protocol rather than the large one. The methods which are trained in large datasets show decent results in face recognition task on YTF. Significantly, the proposed LSTM-ResNet method exhibits analogous results. Note that face recognition methods try to recognize an individual on an image correctly, but LSTM-ResNet tries to recognize an individual on a sequence of frames recorded the person. The best obtained accuracy of ResNet-A architecture by itself is 84.4%, and 94.7% accuracy for LSTM-ResNet-A architecture, which is a notable improvement compared to the ResNet-A. By learning temporal features from input sequences, the LSTM-ResNet-A model performs superior than general residual neural network and analogously to sophisticated architectures trained on large datasets.

4.6 Conclusion

A biometric secure modality is proposed to recognize an individual. The model performs base on the temporal extracted person facial feature changes over time. A neural network based architecture is proposed, named LSTM-ResNet, to make a precise recognition task. LSTM-ResNet model leverages cascaded residual neural network blocks to extract the temporal facial feature and map them into the embedding vector; each embedding vector presents a sole person in a frame. Then for each individual related embedding vectors are fed into LSTM units to learn the long-term temporal structure from a sequence of frames captured by a camera(s), and then perform the recognition task. The model shows decent results comparing with sophisticated models, although it is trained

in small dataset. The model is designed to implement as a device-embedded biometric authentication to keep data securely decentralized on local devices and safe from hackers. In addition, the decentralized model preserves users privacy and eliminate the inconvenience of storing sensitive biometric data.

Chapter 5: DISTRIBUTED ALGORITHM WITH INHERENT INTELLIGENCE FOR MULTI-CLOUD RESOURCE PROVISIONING

This chapter has been published: Miraftabzadeh, Seyed Ali, Paul Rad, and Mo Jamshidi. "Distributed Algorithm with Inherent Intelligence for Multi-cloud Resource Provisioning." In Intelligent Decision Support Systems for Sustainable Computing, pp. 77-99. Springer International Publishing, 2017.

5.1 Introduction

According to a forecast from International Data Corporation (IDC) (`http://www.idc.com/`) the worldwide spending on public cloud services is expected to surpass $107 billion in 2017 [157]. Among different forms of delivering cloud services, IDC recognized the Infrastructure as a Service (IaaS) model as one of the fastest growing categories with compound annual growth rate of 27.2%. IaaS is a promising solution for enabling on-demand access to an elastic pool of configurable and virtual computational services (e.g., computing power, storage, and networks) in a pay-as-you-go manner. IaaS providers offer computational services in the form of Virtual Machine (VM) with specific resource characteristics such as computing power, memory, disk space and networking bandwidth along with types of operating systems and installed applications. Despite traditional distributed computing systems such as Grid [37], cloud computing focuses on the monetary focal point while promising flexibility and high performance for users and cost-efficiency for operators. Furthermore, taking advantage of the virtualization technology gains more resource utilization and Return On Investment (ROI) [9].

In addition, cloud computing is increasing its penetration among industry, research institutes, and universities across applications. An increasing amount of computing is now performed on public clouds, such as Amazon EC2, Windows Azure, Rackspace, and Google Compute Engine or

on private clouds hosted by enterprise using open source OpenStack software. In addition, numbers of cloud computing research test beds and production sites such as Chameleon or JetStream have been supported by National Science Foundation (NSF) to support academic research.

As a drastic increment of the customers, a variety of requested types and emerging cloud computing providers profit maximization for customers and providers becomes a more entangled problem in terms of minimizing the cost using resource management [9, 121, 122, 141], performance improvement in networking and applications [104, 120], maximizing request revenue by providing multi price services [45], market segmentation [163] and prognostication the demands [176].

Considering the natural perishable characteristic of the cloud resources produces a primary conclusion to maximize the resource utilization and to avoid wasting the non-storable cloud resources. On the other hand, honoring the associated Service Level Agreement (SLA) which guarantees the certain level of Quality of Service (QoS) imposes another robust constraint and increases the complexity of the problem. A topic of recent interest, the federated cloud and multi-cloud deployment, allows different cloud providers to share resources for increased scalability and reliability. The aim of the Multi-Clouds is to integrate resources from different providers in a way that access to the resources is transparent to users and cost can be reduced [96]. Furthermore, the Multi-Clouds is a possible mechanism for maximizing a provider profits by leasing part of the available resources in the pool computing center for the requested bids by other providers [32, 96, 128]. So the Multi-Clouds implicitly maximizes the resource utilization while keeping the required QoS.

Maximizing the resource utilization requires the resource allocation decision which involves determining what, how many, where, and when to make the resource available to the user [87]. Typically, users choose the type and amount of the cloud resources and providers place the petition for specific ones onto nodes in their datacenter. Running the application efficiently entails matching the type of the resources and the workload characteristics. In addition, the amount should be sufficient to meet the indicated constraints in SLA. In a pay-as-you-go environment like the cloud where users can request or return resources dynamically, time scheduling such as an adjustment is also important to consider. To maximize resource utilization, the job scheduler is responsible

for allocating desired resources to an individual job and in parallel it should be guaranteed that each job is given an adequate amount of resources, or its fair share. Such a scheduling decision becomes more complex in the cloud computing as a heterogeneous environment with two players: cloud providers and cloud users. On one side, there are the users who have fluctuating loads with a variety of applications and requests. On the other side, consolidated cloud environments are likely to be constructed from a variety of machine classes, representing different options in configuring computing power, memory and storage [125]. The Multi-Clouds is an exceedingly multi-parts heterogeneous system as a result of sharing diversity sets of datacenters in terms of technical performance and monetary strategies.

Mathematical interpretation of effectively allocating resources in the Multi-Clouds is a multi-dimensional self-optimization problem in distributed autonomic computing systems. As a bridge between providers and users, utility functions have very smart theoretical properties and their use in practical autonomic computing systems started to explore in [17] and [72]. Utility functions can enable a collection of autonomic elements to continually optimize the use of computational resources in a dynamic, heterogeneous environment [161].

There are a lot of works that have been done for two major cloud platform advantages: producing the flexible available resources and minimizing the cost of the datacenters [7, 8, 49]. Nevertheless, relatively less work has been done on the cloud provider sideway to maximize revenue and diminish cost of services. In this paper, based on the Sigmoidal and Logarithmic utility functions the proposed algorithm guaranteed the dynamic and scalable job scheduling in the heterogeneous Multi-Clouds environment for the most flourishing cloud service, IaaS. In parallel, the algorithm guaranteed the minimal management effort or service providers? interaction. The algorithm can rapidly provide and release Multi-Clouds pool resources at once period of processing decision.

5.2 Background in Resource Management in Cloud Computing and Grid Computing

This section describes the resource management found in Grids and Clouds, covering topics such as the compute model, data model, virtualization, monitoring, and provenance. These topics are extremely important to understand the main challenges that both Grids and Clouds face today, and will have to overcome in the future [38]:

5.2.1 Compute Model

Most Grids use a batch-scheduled compute model, in which a local resource manager (LRM), such as PBS, Condor, SGE manages the compute resources for a Grid site, and users submit batch jobs (via GRAM) to request some resources for some time. Many Grids have policies in place that enforce these batch jobs to identify the user and credentials under which the job will run for accounting and security purposes, the number of processors needed, and the duration of the allocation. For example, a job could say, stage in the input data from a URL to the local storage, run the application for 60 minutes on 100 processors, and stage out the results to some FTP server. The job would wait in the LRM?s wait queue until the 100 processors were available for 60 minutes, at which point the 100 processors would be allocated and dedicated to the application for the duration of the job. Due to the expensive scheduling decisions, data staging in and out, and potentially long queue times, many Grids don?t natively support interactive applications; although there are efforts in the Grid community to enable lower latencies to resources via multi-level scheduling, to allow applications with many short-running tasks to execute efficiently on Grids [123].

Cloud Computing compute model will likely look very different, with resources in the Cloud being shared by all users at the same time (in contrast to dedicated resources governed by a queuing system). This should allow latency sensitive applications to operate natively on Clouds, although ensuring a good enough level of QoS is being delivered to the end users will not be trivial, and will likely be one of the major challenges for Cloud Computing as the Clouds grow in scale, and

number of users.

UberCloud is the community and marketplace platform to discover, try, and buy computing resources. This platform has explored the roadblocks of Cloud Computing with a crowd-sourcing approach. In mid 2014, more than 60 cloud resource providers and 80+ software providers joined the UberCloud, to provide free computing cycles to experimental projects, and commercial computing cycles to business. During the Computing Insight UK 2015, the UberCloud and ClusterVision, jointly presented Docker based HPC cloud stack. They showed that capability of Trinity in managing UberCloud highly optimized application containers [40].

Cloud Services Brokerage offers three capabilities: Cloud Service Intermediation, Aggregation, and Cloud Service Arbitrage. STRATOS which facilitates the management by using elements/services sourced on the fly from multiple providers is proposed in [113]. Houidi et all present the algorithm to split the user requests while certify provisioning from multiple cloud and the same algorithm is settled to resourcefully split the cloud requests among the multiple cloud platforms with the intention of decreasing the cost for customers [58]. The other type of broker, a cloud aggregator, may benefit from charging a premium for providing more than a straightforward resale agreement.

5.3 Resource Management and Upcoming Challenge

Two main stakeholders in cloud computing, providers which are serving cloud services and customers which are client or consumers of the produced services, have their own inevitable momentous operative influence to make the cloud environment as a heterogeneous system. Particularly, heterogeneous cloud computing datacenter is considered in this section to overcome the resource management problems in multi-cloud.

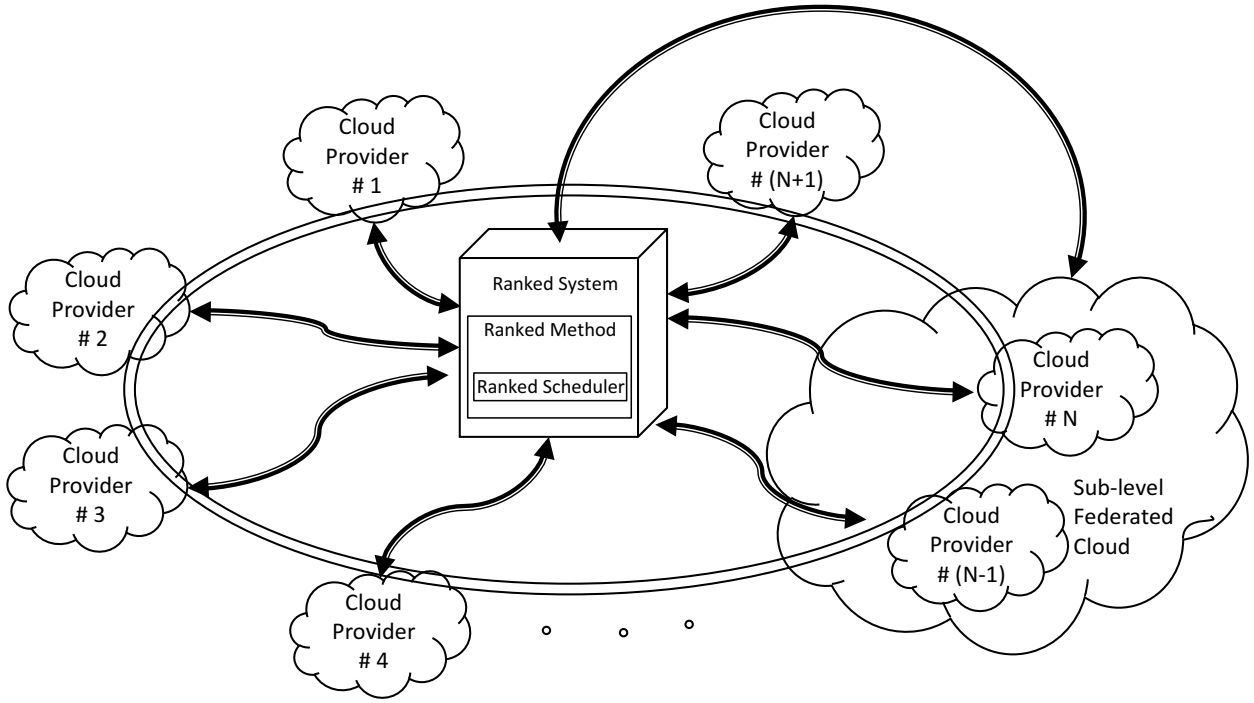


Figure 5.1: Illustration of Ranking Method.

Collaboration across the clouds requires strapping monitoring and management mechanism which are the key components for provisioning, scheduling and failure management [36]. Efficiently provisioning computing resources in a pool of heterogeneous datacenters has been done by the proposing Ranking Method. Producing a dynamic sequence of the available resources based on the hierarchical characteristics of the cloud providers is the first step of the proposed algorithm. The dynamic sequence could be matched with dynamic pricing which it has been acknowledged by the literature [96, 167]. Since then the model could be considered as one of the dominant methods to maximize the request revenue and consequently get the most out of the resource utilization. The dynamic pricing can be explored by using intermediate parameter as Bids between customers and cloud providers. In addition, the Ranking Method can be administrated based on most instantaneous parameters, in [150] more detail for effective parameters are described. Figure 5.1 illustrates the high level topology of the Ranked system. The Ranked System can have been embedded on cloud embodiment or it can function as an independent scheduler system. It consists of Ranked Method and Ranked Method consists of Ranked Scheduler. The Ranked system communicates not

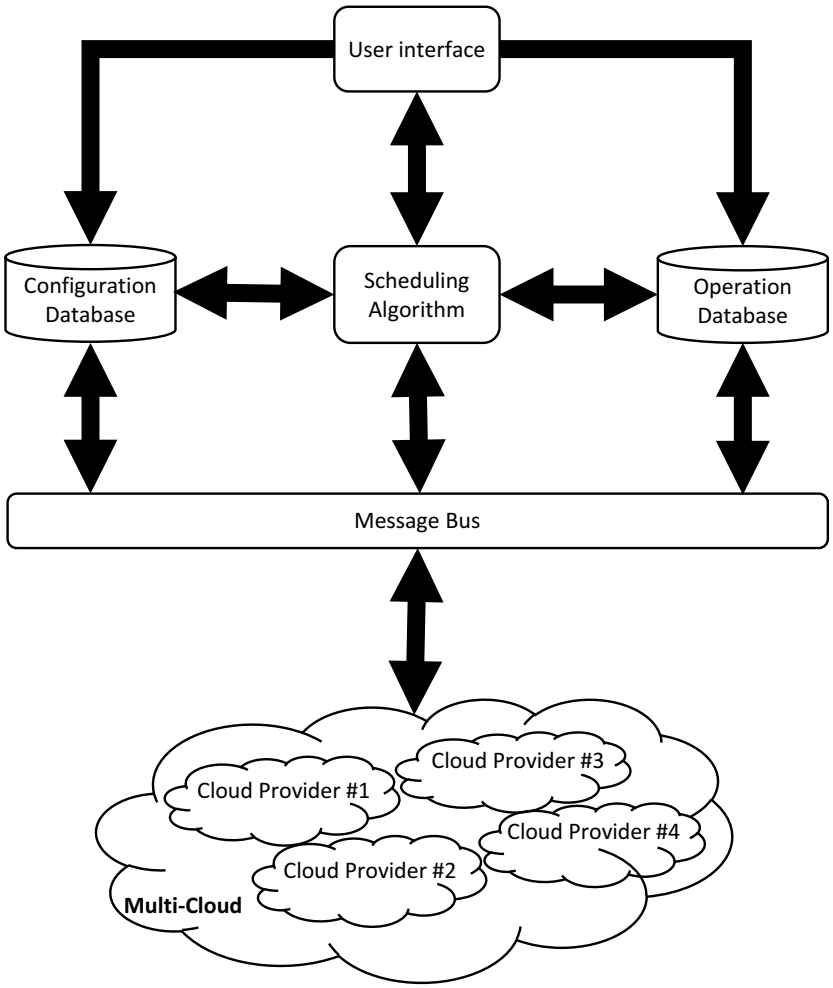


Figure 5.2: Distributed scheduling system components.

only to cloud providers directly but also to other types of cloud topologies such as Multi-Clouds directly or in directly and monitor and control ongoing processes.

Distributed scheduling system consists four major components, see figure 5.2. Following the API, user interface communicates with the Ranked System distributed in Configuration Database, Scheduling Algorithm, and Operation Database. Configuration Database feeds the Scheduling Algorithm with the assigned utility functions. Using intelligent forecasting, utility functions describe the requested services while considering the providers? concerns; for instance, maximizing the resource allocation while minimizing the power consumption. Operation Database applies the final decision of the Scheduling Algorithm and inspects online activities of consumers and providers; the aim of the latter task is for real time notification for the Scheduling Algorithm.

Ranked System follows the following clues:

- similar instances of different providers are situated in the same group (which is named clutch)
- initially top ranked instances have the uppermost slots and bottom ranked instances have the lowermost slots
- for customer satisfaction, evaluation of the requested service is placed in a specific group based on the lowest and highest requested instances; interpolation and other methods can be used
- before fully occupied resources, top ranked instances are allocated to the corresponding request
- after fully occupied resources, top ranked instances may not be allocated for the request but another ranked instance in a same clutch has more opportunity to be allocated to the task. This action strictly depends on the assigned utility function to the request.
- transcendent ranked instances ameliorate customer satisfactions for the assigned service
- transcendent ranked instances are unintended to be occupied if the number of demand increases
- inferior ranked instances keep the request revenue in order to maximize the cloud revenue
- instances in a same clutch can be reshuffled to maximize the resource utilization if there is profit for the corresponding providers
- hedging from clutch to the adjacent clutch is possible in a case of resource shrinking and strictly depends on the utility function
- each clutch could include different providers

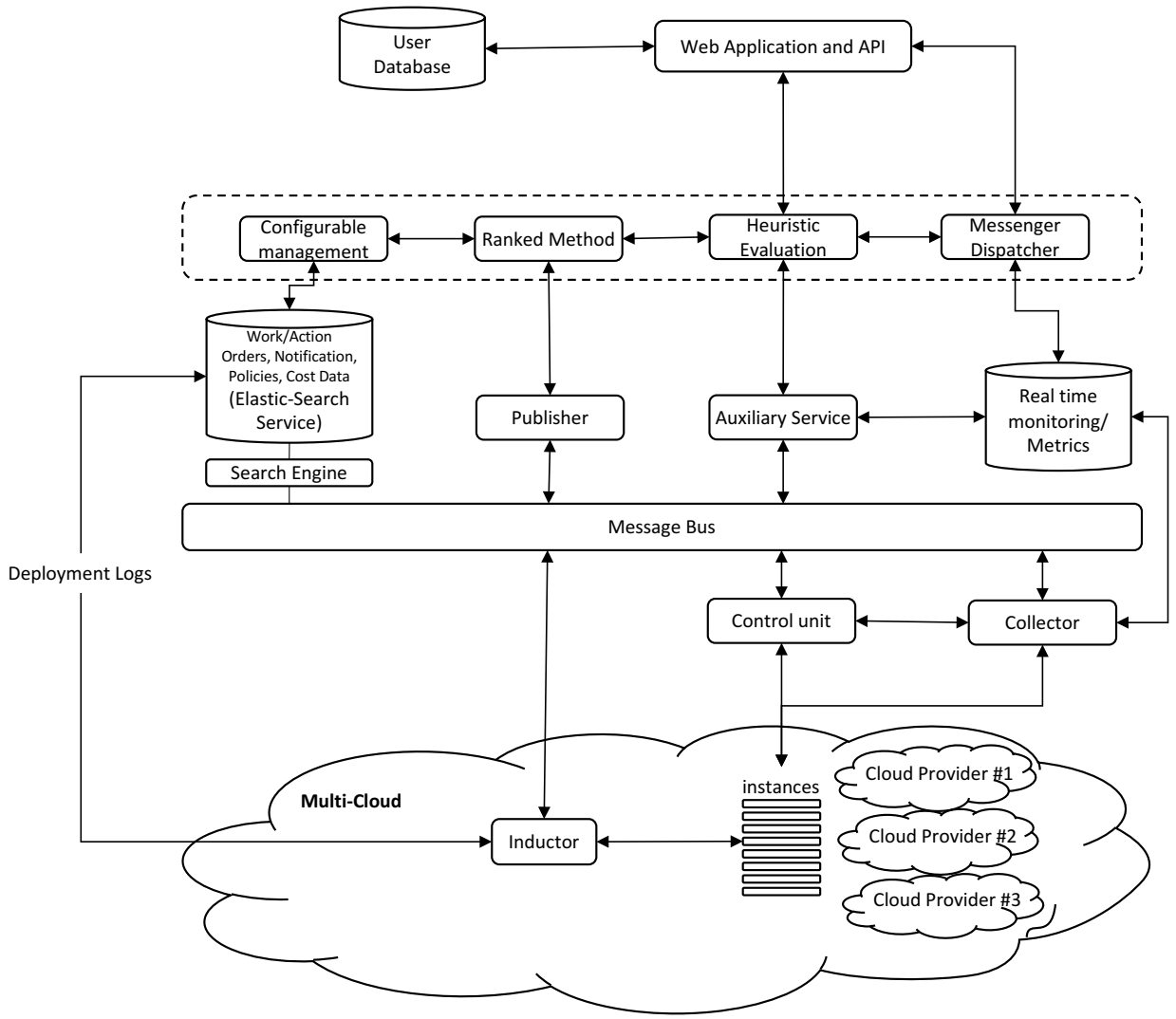


Figure 5.3: Detailed Rank System Architeccture.

Details of Ranked system architecture are depicted in figure 5.3. Web Application and API is the gate for managing applications, services, clouds; it supports Rest based API alongside any other types of input which can be done through User Interface. Configurable management is the manager of models, assemblies, environment; it matches the demands from the consumers to the user profile. Ranked Method is responsible for creating scheduling design and deploy the design; more explanation is provided in the following paragraph. Heuristic Evaluation which is responsible of comparing deployed design to planned design, consumers activities and consumers updates. Messenger Dispatcher provides the details monitoring facilities for User Interface with the metrics data. Elastic-Search Service records all deployments, thread, dataflow, task mapping, assemblies,

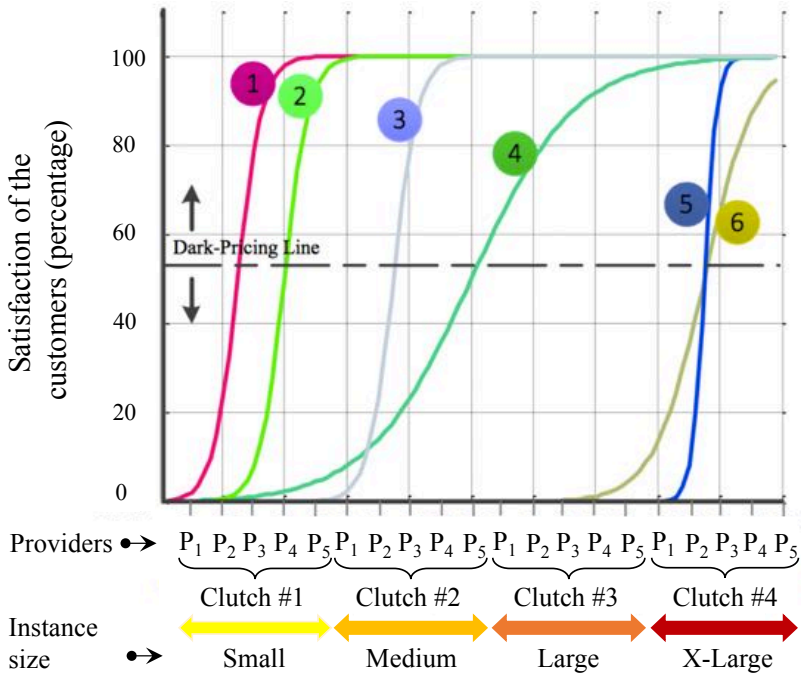


Figure 5.4: Illustration of the utility functions and application in Ranked System.

and environments. The attached engine to it provides log analytics, real-time application monitoring, and click stream analytics. Publisher tracks the Ranked Method and post the design on the massage bus. Auxiliary Service helps implementing policies and costs models. Real time storage of the metrics data which are provided by Collector, is a database having the features such as scalability and high availability without compromising performance. Internal communication between the components is carried on by Message bus. Controller is responsible for distributing action orders. Functionality of the collector includes metrics data from instances for any event. multi-cloud contains an Inductor and the pool of shared. Inductor receives the design and deploy the actions and post the results message to the system. It is responsible for executing steps in implementation, gather the queue of clouds specifications, and make the reaction queue and report it to controller.

5.3.1 Utility Functions in Ranked System

In this section a scenario is studied to clarify how the proposed algorithm gets advantage of the sigmoidal utility function; the concept can be extended to logarithmic function which we use for elastic tasks. Figure 5.4 illustrates four clutches in which five providers make a pool of ranked

instances as small, medium, large and extensive large. Six unalike sigmoidal utility functions corresponding to the six types of customers or services are shown in Figure 5.4. Utility functions 1 and 2 belong to the small clutch in which provider number 5 (P5) has the highest priority to serve any request in the clutch. In this clutch customers who are assigned to utility function 2, rather than 1, has more chance to take the better service based on the chosen ranked parameter.

However, utility function 1 gets services simultaneously from P5 if enough resources are available in the Multi-Clouds. By increasing demands and reducing the number of free instances, the chance of getting services from inferior provider increases. The Ranking Method tries to provision resources for all of types of the customers. However, by increasing the number of requests for a specific provider or clutch, based on the Service Level Agreement (SLA), three scenarios are likely to be happened. First, pass the customer to the lowermost neighbor provider or lower ranked provider. Chosen desired sigmoidal function by the customers carries on this scenario automatically. Second, the customer may lose allocated instances if it does not satisfy the provider proposed conditions such as raising the price. This scenario can be implemented by control the whole number of available resources, $(V - z)$, in each processing period. Finally, the customer accepts the proposed condition and the allocated instances are remained to serve the customer. Utility function 3 describes the only service in the medium clutch. It spread over the whole clutch from P1 to P5; so each of the serving providers can provision the customer.

For large clutch only utility function 4 is presenting services but it spreads not only in large clutch but also from small clutch to extensive large clutch. Consequently, customers in this utility function have a potential to get serve in a variety clutches; the customers have a chance to have an experience in extensive large clutch when enough pooled resources are available. Apparently, it is not a case when Multi-Clouds has an experience in high amount request.

This types of utility function expansively helps a lot to maximize the revenue of the requests by providing motivation such as paying less for this types of functions. Finally, in extensive clutch two types of utility functions are managing the requests revenue and resources of the providers. As it is illustrated utility function 5 has the sharpest slop in this simplified properties illustration

of Ranking Method. This types of utility function can be fixed in a specific provider with two applications: first, serves a customer by a specific ranked or no ranked properties over a time. Next, fixed the instance of the provider for a service after the customer accepts the dynamic conditions of the provider such as dynamic pricing.

Figure 5.5. shows how clutches are formed in the Ranked Method. Monitoring delivers the specifications of the cloud providers to categorize the instances based on VDs properties, for example Computational power, Memory Size, and Hard Drive capacity. To this point it is refer to collecting the information. After the collecting process the selection process is established. Dynamic and static parameters are observed to weight instances and assigned best matched utility function. Dynamic parameters could be any variables which may vary; such as measurement unit, occupation percentage of a cloud provider, networking parameters, elastic resources. Static parameter such as: service type, security, Quality of Service (QoS), pricing unit, instances size, operating system, pricing sensitivity to regions, subcontractor, and many other.

In addition, figure 5.5 illustrates the weighting, indexing, and ranking steps. It is based on statistics and predictions. Predicting the variation is not limited to the cloud providers side, it is connected to the user data also. Indexing steps uses the activity and history of the cloud providers and customers to predict the workload. Machine Learning algorithms and other methods can be used in this steps. The weighting phase monitors the system continuously. However, indexing phase is one-time decision; it is carried on in every specific period of time. Finally, a queue is generated to make the ranked instances using the result of weighting and indexing phases.

So far, it is described that how the Ranking Method provides flexibility to serve the customers dynamically, by applying some principle conditions in the SLA to match the customer requests and utility functions. Maximizing the profits of the cloud providers, as one of the primary goals of the Multi-Clouds, makes the Ranking Method as a method to potentiate the dynamic pricing conditions in the SLA. In the next step mathematical formulation would be studied and the Ranking Method is presented as the distributed algorithm in the Multi-Clouds.

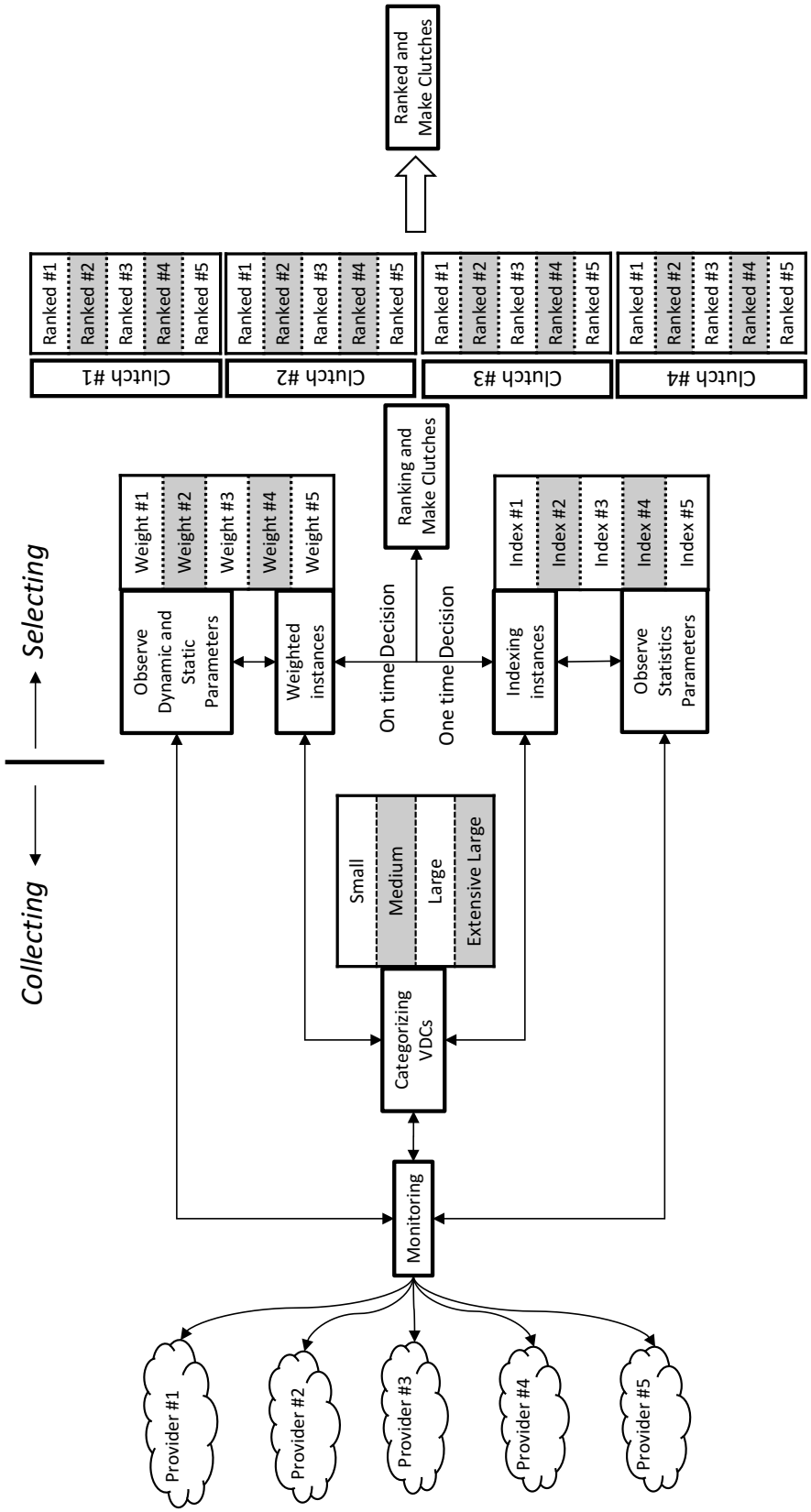


Figure 5.5: General view of ranking steps.

5.4 Mathematical Formulation of the Algorithm

In this section, we will present design and implementation of the distributed algorithm of optimized controller for resource allocation of elastic and inelastic traffic using logarithmic and sigmoidal-like utility functions. This algorithm is based on the utility proportional policy to maximize the requests revenue where the fairness among users is in utility percentage and describes the user gratification with the service of the corresponding bid. Furthermore, this algorithm precisely provides information of the available resources and distributes them among the users. It could be implemented as a part of the cloud service brokerage or as a separate unit in cloud computing architecture of IaaS.

Mathematical formulation of the algorithm includes Sigmoidal [88] and Logarithmic [159] utility functions which have had applications in resource scheduling. These are used as gauges for dynamically resources allocation such that predefined required constrains by SLA are satisfied and cloud providers get the benefit of using all their available resources and minimizes the operating costs. Virtual data center (vDC) is considered as the resource unit; however, the approach can straightforwardly be extended to consider a variety kinds of instances which include different amount of compute, memory, storage and bandwidth resources.

The vDC allocated by controller to the k^{th} service is given by v_k. Presumption of the algorithm is suitable utility function is assigned to the service by using techniques such as interpolation, statistic methods or others. Our objective is to determine vDC that the scheduler should allocate to the specific customer. For this optimization problem we are looking for the strictly concave utility functions that satisfies the following properties:

(a) $U_k(v_k)$ is an increasing function of v_k.

(b) $U(0) = 0$ and $U(\infty) = 1$.

(c) $U_k(v_k)$ is twice continuously differentiable in v_k.

(d) U_k is (a) bounded above.

Note that typically, most utility functions in current networks can be represented by three types of functions:

- Sigmoidal-like
- Strictly concave function
- Strictly convex function

Next, let us consider two major type of tasks: Sigmoidal-like Tasks and Logarithmic Tasks.

Sigmoidal-like Tasks

First type refers to the task that requires the specific amount of resources which are requested by the service. The characteristic of the requested *pay-as-you-go* service match to the suitable sigmoidal function. Usually, inelastic tasks can be defined by sigmoidal function. It can be expressed as:

$$U_k(v_k) = c_k(\frac{1}{1+\exp(-a_k(v_k - b_k))} - d_k) \tag{5.1}$$

to satisfy all the required properties it is enough to have $c_k = \frac{1+\exp(a_k b_k)}{\exp(a_k b_k)}$ and $d_k = \frac{1}{1+\exp(a_k b_k)}$. By choices a_k and b_k describe the utility function of the k^th user. For example, by choosing $a = 5$ and $b = 10$ a good approximation for a step function can be obtained which explains the fixed usage of the allocated resources; for instance, users with the specific requirement of the resources. Normalized sigmoidal-like utility function with $a = 0.5$ and $b = 20$ is another example for representation of the adaptive real-time application.

Logarithmic Task

It is desired to use all resources in a pool of vDCs as much as possible. In a case of temporary real-time services, the same as spot instances, normalized logarithmic utility function describes the good description. It is referred to the mission of which a minimum number of resources are needed and it has a desire to occupy more resources in a case of availability. Mathematical expression of

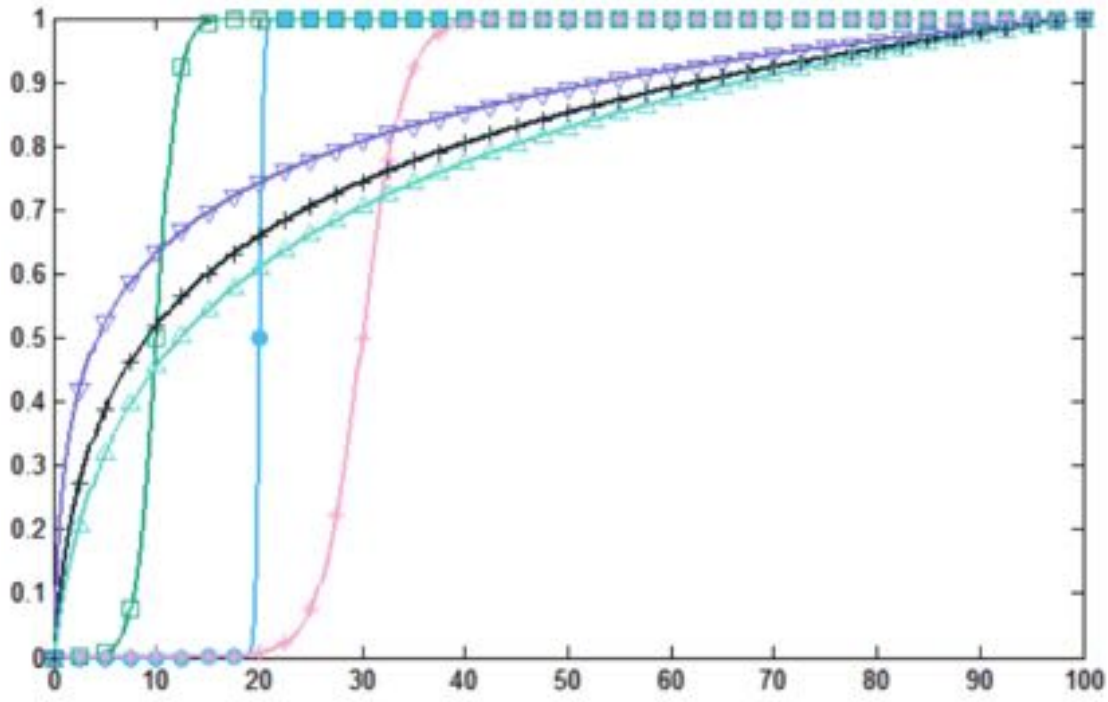


Figure 5.6: Samples of Sigmoidal and Logarithmic functions.

the function is as:

$$U_k(v_k) = \frac{\log(1 + m_k v_k)}{\log(1 + m_k v_m ax)} \tag{5.2}$$

where $v_m ax$ is the point in which the task occupies all available resources as same as what is mentioned in the SLA. For example, if the user is fully satisfied by 30 vDCs, $v_m ax$, should be set to . Any allocated vDCs less than 30 leads to the minus satisfaction but it does not mean user is loss its performance since the performance is based on the required resources for doing the specific job. It means if the user is using all 30 vDCs there is no request issues from the user for share the resources to others. But if the user is not fully using allocated resources and it is confirmed by the monitoring and negotiation process then the user can share the resources by management process. In such a case the customer and the provider get the advantage simultaneously. In 5.2, m_k is the rate of is the rate of increasing utility percentage with the allocated rate v_k.

Both function types are satisfied the constraints as in (a) to (d). In figure 5.6 the normalized sigmoidal-like and Logarithmic examples of utility functions are shown.

To maximize the resource allocation based on the utility function the objective function is

defined as follow:

$$\begin{aligned} &\max_{v} \prod_{k=1}^{M} U_i(v_i) \\ &\text{Subject to} \sum_{k=1}^{M} v_k \leq R \quad \{v_k \geq 0 \quad \text{and} \quad i = 1, 2, ..., M\} \end{aligned} \tag{5.3}$$

where $v = v_1, v_2, v_3, ..., v_k$ and is the number of the requests. This formulation ensures non-zero resource provisioning for customers while guarantees minimum predefined constraints are met. Furthermore, using Sigmoidal function leads to the more resource allocation for corresponding tasks when the specific amount of resources is required.

Now we do have an optimization problem for proportional resource allocation based on the utility functions [32]. Prove of existence of the global optimal solution for (5.3) is straightforward. Since (5.3) is a convex optimization problem a unique trace-back global optimal solution is exists for the problem.

Objective function of $\max_{v} \prod_{k=1}^{M} U_i(v_i)$ is equivalent to $\max_{v} \prod_{k=1}^{M} \log(U_i(v_i))$ and for both applied functions, Sigmoidal and Logarithmic:

$$\begin{aligned} &\frac{d}{dv_i} \log U_i(v_k) > 0 \\ &\frac{d^2}{dv_i^2} \log U_i(v_k) < 0 \end{aligned} \tag{5.4}$$

So both functions are strictly concave natural logarithms. Therefore, the optimization problem is a convex optimization problem and there exist the unique trace-back global optimal solution [33].

5.4.1 Dynamic Distributed Resource Provisioning Approach

To make the optimization problem as the negotiated process for provision resources dynamically, based on demand and supply, this paper gets advantage from the dual problem to distribute the optimization problem among the Multi-Clouds stakeholders. In communication network it has

been done similarly in [34], [35], [32] by defining the Lagrangian function as:

$$\begin{aligned} L(v,p) &= \sum_{k=1}^{M} \log(U_k(v_k)) - p(\sum_{k-1}^{M} v_k + z - R) \\ &= \sum_{k=1}^{M} (\log(U_k(v_k)) - pv_k) + p(R - z) \\ &= \sum_{k=1}^{M} L_k(v_k, p) + p(R - z) \end{aligned} \tag{5.5}$$

where $z \geq 0$ is the slack variable (we discuss in more detail about the slack variable later) and is Lagrange multiplier or the dark price which is the intermediate parameter in negotiation between the Multi-Clouds and customers. To match the ranked instances by Ranking Method which is proposed in the paper, dark price is weighted by the ranked order which is called bid in the paper. Bidding strategy attracts a lot of researcher especially for spot instances [36], [37]. By assigning a set of utility functions not only spot instances but also other tasks can be evaluated simultaneously to make the decision for resources provisioning in the Multi-Clouds.

Satisfying the simultaneous scheduling for multi types of tasks k^{th} the bid for the instance can be given by $w_k = pr_k$. Subsequently, the dual problem objective function can be written as:

$$D(p) = \max_{v} L(v,p) \tag{5.6}$$

and the dual problem is given by

$$\begin{aligned} &\min_{p} D(p) \\ &\text{Subject to} p \geq 0 \end{aligned} \tag{5.7}$$

to find the optimal answer driving a derivation of the new optimization problem leads to the interesting formulation for dark price by using $\sum_{K=1}^{M} W_k = p \sum_{K=1}^{M} r_k$ and solving the equation for p

as :

$$\frac{\partial D(p)}{\partial p} = R - \sum_{k=1}^{M} v_k - z = 0$$
$$p = \frac{\sum_{i=1}^{M} w_i}{R - z} \tag{5.8}$$

latest equation entails critical information to optimize the problem for p:

- Summation of the bids (from all requests)
- Available Instances (all the active resources)

which both of them are available in the centralized node, monitoring and negotiation process. On the other hand since the is separable in we can rewrite the equation as:

$$\max_{v} \sum_{k=1}^{M} (\log(U_k(r_k)) - pv_k) = \sum_{k=1}^{M} \max_{r_k} (\log(U_k(r_k)) - pv_k) \tag{5.9}$$

which implies, the optimization problem can be solved for each utility function separately:

$$\max_{v_k} (\log(U_k(v_k)) - pv_k)$$
$$\text{Subject to } p \geq 0, v_k \geq i = 1, 2, ..., M \tag{5.10}$$

Equation 5.7 and 5.10 distribute the optimization problem 5.3 into two parts. Equation 5.7 is in quest of the dark price which requires supply and demands information and can be considered to be solved in the Multi-Clouds. Equation (5.10) which separately maximizes provisioning of the resources for each utility function and it can be counted up as a customer?s attorney. Between these two optimization problems dark price is an intermediate parameter to moderate the criteria for both sides. Establish an iterative process to drawn out the settle down point in which stakeholders get the mutual advantage is the chosen approach of the algorithm. The mutual proliferation leads to the maximization of the request revenue and utilization of the pooled vDCs in Multi-Clouds environment.

5.4.2 Proposed Algorithm for Resource Scheduling

So far, the reasons and requirements for implementation of the expedient scheduling algorithm have been studied in various aspects. Putting together all the described aspects of the proposed algorithm has been projected in two mutually joint algorithms, Back shown in Algorithm (1) and Feed in Algorithm (2). In this section terms explanation and mathematical formulation of the algorithms have been described. Dark Price is considered as the iterative parameter between Back-Algorithm which is the customers attorney and Feed-Algorithm which is the multi-cloud lawyer. Feedback process will continue to settle down in profitable point for customers and providers in the pool of vDCs. Each active tasks send the initial bid to the broker. The broker makes a decision based on the difference of two consequent bids per task with the pre-specified gage as δ. If the absolute difference gets greater than δ the broker specify the new shadow price based on the utility function type. Each user receives the shadow price and solve its own optimization problem $v_k(n) = \max_{v_k}(\log(U_k(v_k)) - p\ v_k)$ for v_k which is used to calculate the new bid $w_k(n) = p(n) \cdot r_k(n)$. The broker collects all the bids sequentially and the process would be repeated until $|w_k(n) - w_k(n-1)|$ gets less than the pre-specified threshold δ The Back and Feed are the same as UE and eNodeB algorithm in [32] There are modifications which are applied in the algorithm to match them with Ranked Method. Scaling factor as $(V - z)$ to make the algorithm as an applicable process in the proposed Ranked Method. In the following chapter simulation results declare the convergence of the algorithm in scale which is the requirement of the sustainable computing.

5.4.3 Simulation Results

Ranking method is applied to the variety sets of utility functions. Convergence of the proposed algorithm is acknowledged by the simulation. In the simulations, to show the generality of the algorithm, it is supposed that instances of the providers are ranked from 1 to 100 which they are included in four clutches with the same size. So one can interpret each number to the instances with the specific parameters. In the first simulation, six utility functions corresponding to sigmoidal functions in figure 5.7. are simulated. Through classification of the instances it is assumed

Algorithm 5.1 Back ($k^{th} Utility Function$)

1. Initial its bid as $w_k(1)$ to Feed
2. **Loop**:

   Received shadow price $p(n)$ from the Feed

   if STOP from Feed **then**

   Calculate allocated rate $v_n^{opt} = \frac{w_k(n)}{p(n)}$

   STOP
3. **else**:

   Solve $v_k(n) = \max_{v_k}(log(U_k(v_k)) - p\ v_k)$

   Send new bid $w_k(n)$ to Feed-Algorithm
4. **endif**

Algorithm 5.2 Feed

1. **Loop**:

   input $(V - z)$ **and scale the functions by** $(V - z)^{-1}$

   if k^{th} utility function is activated

   – receive bids $w_k(n)$ from Back-Algorithm { Assert $w_k(0) = 0$ }

   elsif $|w_k(n) - w_k(n-1)| < \delta \forall$ **then**

   – allocate rates $v_k^{opt} = \frac{w_k(n)}{p(n)}$ to k^{th} utility function

   – STOP

   else:

   – Based on the specific group for p_k:

   $*$ calculate $p(n) = \frac{\sum_{k=1}^{M} w_k(n)}{V-z}$

   – Send new shadow price based on the users category

   endif:
2. **end Loop**

almost the same instances are put to the same clutch and for clearly illustrate of the algorithm only computing power is studied as the gauge to make distinguish between the instances. However, the algorithm has the potential to study all the instances properties such as computing power, memory, disk space and networking bandwidth factors at one simulation. In this case the classification methods should be used to weight the gauge. In the first simulation the computing power is used as the weighted parameter to provide the gauge. The necessitated manipulation to the algorithm is related to the $(V - z)$ as the maximum available resources. The pool of resources in vDC of the Multi-Clouds is the other explanation of the $(V - z)$ manipulation. In the algorithm V is the maximum number of resources and is the reserved resources for the specific tasks or customers. Since by increasing $(V - z)$ just the scaling of the functions are varied, the convergence of the optimization problem is not denied. The manipulation which is used for this simulation is as follow:

$$(V - z) = \sum_{i=1}^{I} \sum_{n}^{N} C_{i,n} P_i^n \tag{5.11}$$

in which I is the maximum number of providers, is the maximum number of the clutches $C_{i,n}$ is coefficient for the the provider in the the clutch and P_i^n is the corresponding provider advantageous over other providers in the lucrative specification terms. The scaling factor is $(V - z)^{-1}$ for all the functions. In the first simulation $P_i^n = 1$ which indicates similarity in the performance of the providers and $C_{i,n} = b_{i,n}$. Figure 5.7 describes the convergence of the algorithms by weighting the ranked clutches in the x axis and scaling all the functions based on the available calculated resources. $(V - z) = 240$ is considered with the iterations $n = 20$. The rates of different functions with the number of iteration are illustrated in 5.7. After giving enough iterations the algorithm is settle down for the sigmoidal functions as 13, 21, 38, 6, 89, and 76 correspondingly. This simulation shows for the 4^{th} sigmoidal function the allocated instances is placed in the lowest clutch. It means the algorithm provides the lowest priority for the 4_{th} function since the 4_{th} (with $\alpha = 0.12$) is spread throughout the operating boundaries.

In the second case the available resources are decreased to $(V-z) = 100$. From the suimulation results in figure 5.8 and referring to figure 5.4 it can be understood sigmoidal function number 5

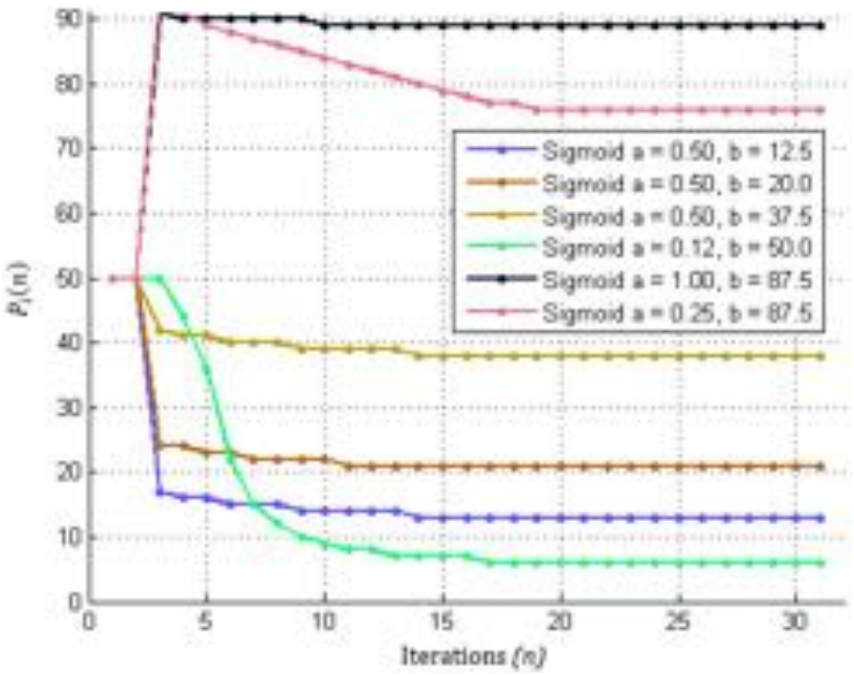


Figure 5.7: Allocated ranked providers convergence $P_i(n)$ with number of iterations for $(V - z) = 240$.

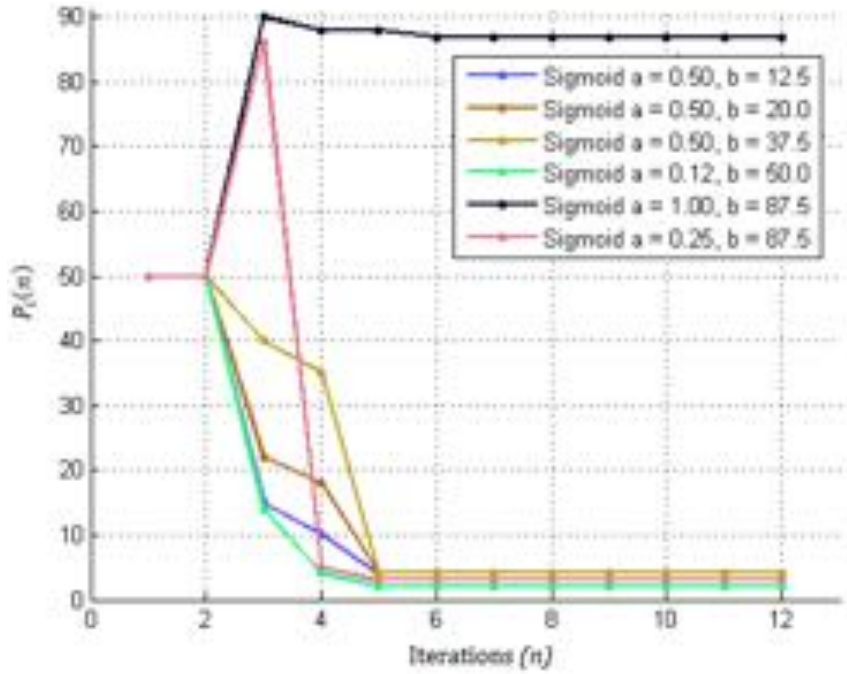


Figure 5.8: Allocated ranked providers convergence $P_i(n)$ with number of iterations for $(V - z) = 100$.

is satisfied with the appropriate instances which is matched with its request. However sigmoidal function number 6 is not confined in its primary clutch and it is downgraded to the small clutch. The reason for that is sigmoidal function number 6 has the value as 0.25 which spreads the function outside of its primary clutch despite of the sigmoidal function number 5 with the value as 1 which is strictly confined the sigmoidal function in its primary clutch. The same reason is applied for function number 4. Since it spreads widely throughout the operating boundaries it gets even less ranked instance than function number 1, 2 and 3. Finally, sigmoidal functions 1 and 2 get the same instances which means low ranked clutch has the opportunely to serve more customers. It means in a case of having higher requests than the available resources, petitions for the low ranked clutches arise dramatically.

In multi-cloud this is the situation which competition between the providers arises to serve more number of customers by offering lower size of instances. Actually when there are enough available resources the algorithm provides the customers by the highest performance resources based on their desired utility functions. However, when the requests are getting higher than the available resources, based on the customers chosen model, customers with bounded utility functions maintain their primary resources with the small variation but the allocated resources for other types of functions are varied to satisfy each chosen model by the customers. In figure 5.9 The steady state rate of different sigmoidal functions with different $(V - z)$ has been illustrated. As mentioned before monitoring system in parallel with brokering strategies control the situation and decide the final decision. Study the utility function number 4 and 6 is interesting. Since the requested resources by this functions have not been satisfied until the rest functions reach their corresponding requested resources.

5.4.4 1.1 Implementation of Spot Instances with the Logarithmic Function

The distributed algorithm has the interesting application in spot instances pricing model to control the resource allocation. Spot instances is following the method which impose dynamic price based on supply and demand. Since maximizing revenue of requests is one of the Multi-Clouds aims, a

dynamic resource allocation has been simulated.

Figure 5.10 shows the sigmoidal and logarithmic functions with their characteristics which are used as reference for 5.11 and 5.12. Figure 5.11 illustrates the distributed algorithm gives the sigmoidal functions higher priority than the logarithmic function. This is because, the algorithm starts giving the resources to the logarithmic functions after the steady state rate of all the sigmoidal functions exceed their corresponding inflection points. Furthermore, the majority of resources are allocated to the tasks with sigmoidal functions.

To illustrates the application of logarithmic functions for spot instances 5.12 describes the characteristic of the algorithm; the allocation of resources for logarithmic functions have been carried on when the available resources are more than the summation of the inflection points. This feature of the algorithms has the potential to use as the spot instances dynamic model.

In the case of the available provider with the specific feature, the instances can be provisioned to the customer of the spot instances and when there is a request of permanent use of the resources on the other specific conditions the spot is return back to the system as the available resource.

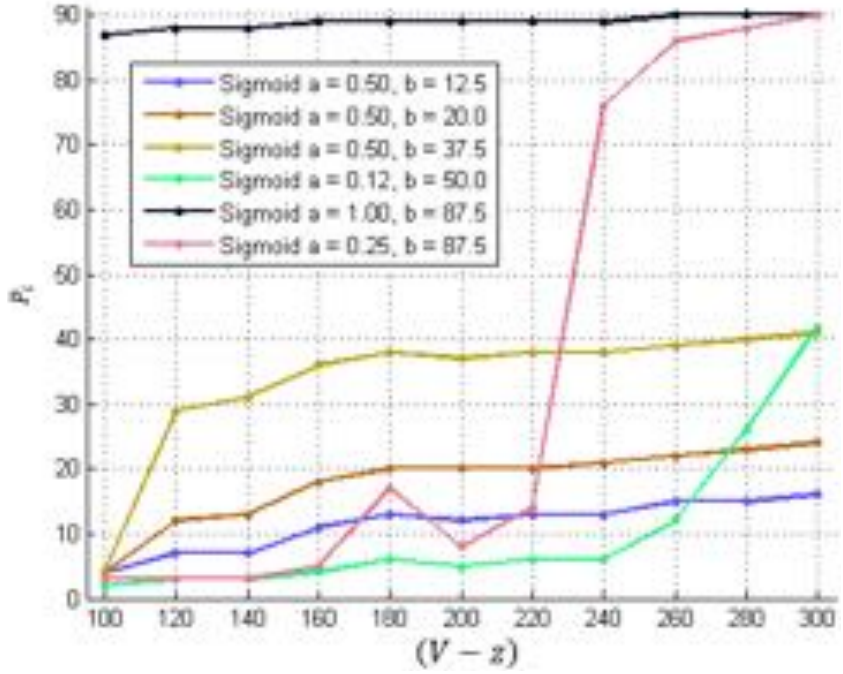


Figure 5.9: Allocated ranked providers for $100 \leq (V - z) \leq 300$.

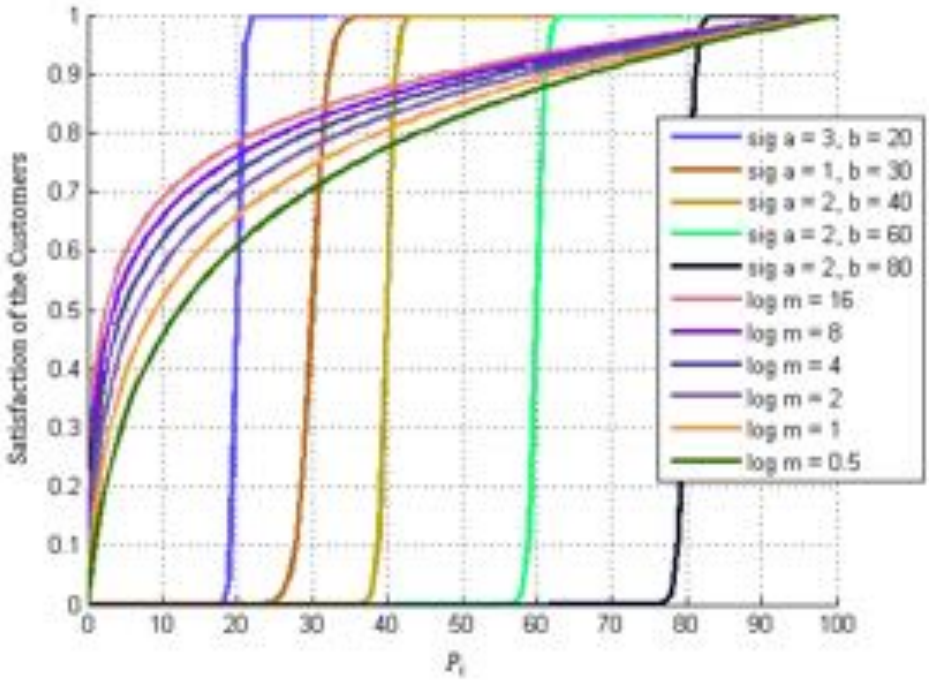


Figure 5.10: Illustration of the sigmoidal and the logarithmic utility functions.

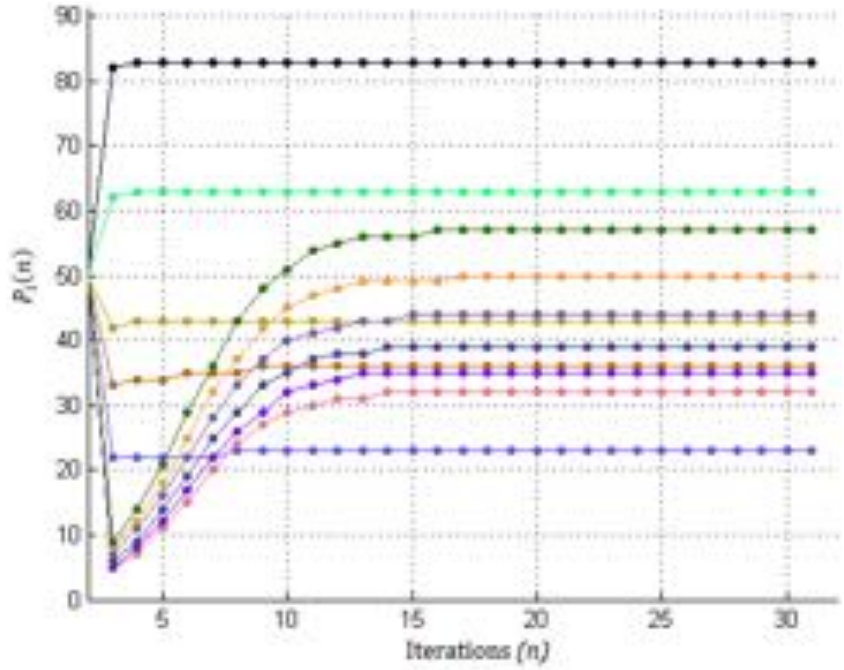


Figure 5.11: Allocated ranked providers convergence $P_i(n)$ sigmoidal and logarithmic functions for $(V - z) = 500$.

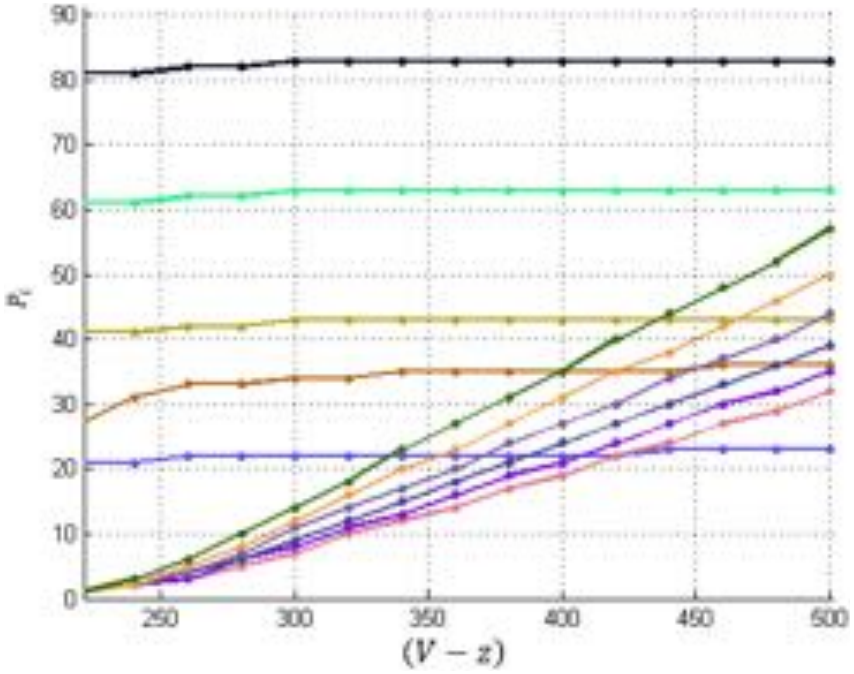


Figure 5.12: Allocated ranked providers convergence $P_i(n)$ sigmoidal and logarithmic functions for $220 \leq (V - z) \leq 500$.

5.5 Conclusion

The Ranking Method has been introduced as the elastic and inelastic tasks scheduler to support the heterogeneous requests and resources in multi-cloud environment. The convergence of the algorithm is studied by simulation for a variety sets of the sigmoidal and logarithmic functions while computations are spread over two mutually joint algorithms, Back algorithm (customers attorney) and Feed algorithm (multi-cloud lawyer). Sigmoidal function has been used to make a balance between requests and resources while always SLA requirements are guaranteed. To maximize the usage revenue, in a pool of resources, the combination of logarithmic and sigmoidal functions has been proposed to support the spot instances for the elastic tasks while guaranteed the sigmoidal functions always have the priority; which means the algorithm support the elastic and inelastic applications simultaneously. In addition, demonstration of the Ranked method elucidates the capabilities to handle the dynamic resource provisioning in scale in multi-cloud system. The substantial Ranked system is proposed to support the Ranked method in the heterogeneous multi-cloud environment. One of the main limitation in scheduling algorithm is the delay response of the system which is strongly suggested for the future research for the Ranked system.

Chapter 6: INTELLIGENT DECISION SUPPORT SYSTEMS FOR HETEROGENEOUS AUTONOMOUS AGENTS

This chapter has been published with modifications: Miraftabzadeh, Seyed Ali, Nicolas Gallardo, Nicholas Gamez, Karthikpai Haradi, Abhijith R. Puthussery, Paul Rad, and Mo Jamshidi. "Cloud robotics: A software architecture: For heterogeneous large-scale autonomous robots." In World Automation Congress (WAC), 2016, pp. 1-6. IEEE, 2016.

In this chapter, we propose an intelligent decision support system which Includes three subsystems in a cloud environment: Middleware Subsystem, Background Tasks Subsystem, and Control Subsystem. The architecture invokes cloud technologies such as cloud computing, cloud storage, and other networking platforms arranged to provide assistance with congregated infrastructure and shared services for robotics, such as a Robotic Operating System (ROS). Since work is looking for a reliable, scalable, and distributed system for the heterogeneous large-scale autonomous agents, Infrastructure as a Service (IaaS) is chosen among the cloud services. Three major tasks can be handled by the proposed software architecture: Computing, Storage, and Networking. Hadoop–MapReduce provides the appropriate framework in the cloud environment to process and handle these tasks.

6.1 Introduction

Robotic services are systems, devices, and robots with three functions: sensation, actuation, and control [67]. Providing robotic services to support intelligent artificial activities through socially conscious interactive behaviors is an emerging topic in robotics research to support, for instance, daily human activities [35] and IBM Smarter Cities `http://www.ibm.com/smarterplanet/us/en/smarter_cities/overview/`. To achieve this, robotic technologies can be integrated with advanced networking technologies [31] to foster the emergence of networked robotics.

A networked robotic system allows for communication between a group of robotic devices that are connected via a wired and/or wireless communication network http://www-users.cs.umn.edu/~isler/tc/. Networked robotics, especially a multi-robot system, distribute the workload of sensing, actuating, communication, control, and computation among a group of participating robots. These systems have been put in place with great success for industrial applications, intelligent transportation systems, and security applications. However, the advancement of networked robotics is restricted by resource, information, and communication constraints inherent in the existing framework [60]. These drawbacks are especially significant for mobile robotics networks, and may lead to severe performance degradation.

Networked robotics serves as a stepping stone [60] towards cloud robotics. For instance, cloud-enabled network robotics leverages emerging cloud computing technologies to enhance networked robotics, removing analytical duties off the robot.

The design objective is to overcome the limitations of networked robotics using elastic resources provisioned by an ever-present cloud infrastructure. Cloud computing uses a sophisticated networked ecosystem but presents a clear, concise interface to extend the capabilities of networked robotics.

In 2010, Kuffiner J. J. penned the term "Cloud Robotics" and described a number of potential advantages over conventional robotics [78]. Cloud robotics is an emerging field of robotics embedded in cloud computing, cloud storage and cloud networking. While providing the advantage of powerful computational, storage, and communications resources of modern data centers, they also allow robots to benefit from the platform, which includes infrastructure and shared services. Robots connected through the cloud platform can access services running on remote servers. Furthermore, one byproduct of cloud robotics is the allocation of background tasks, or services not related to the core function of the robot, to the cloud platform; for example, Big Data management and implementation of advanced Machine Learning algorithms [42, 69] and heavy image enhancement algorithms [6, 129] or software [43]. The National Institute of Standards and Technology (NIST) definition of cloud computing [95] enables remarkable elasticity in designing and

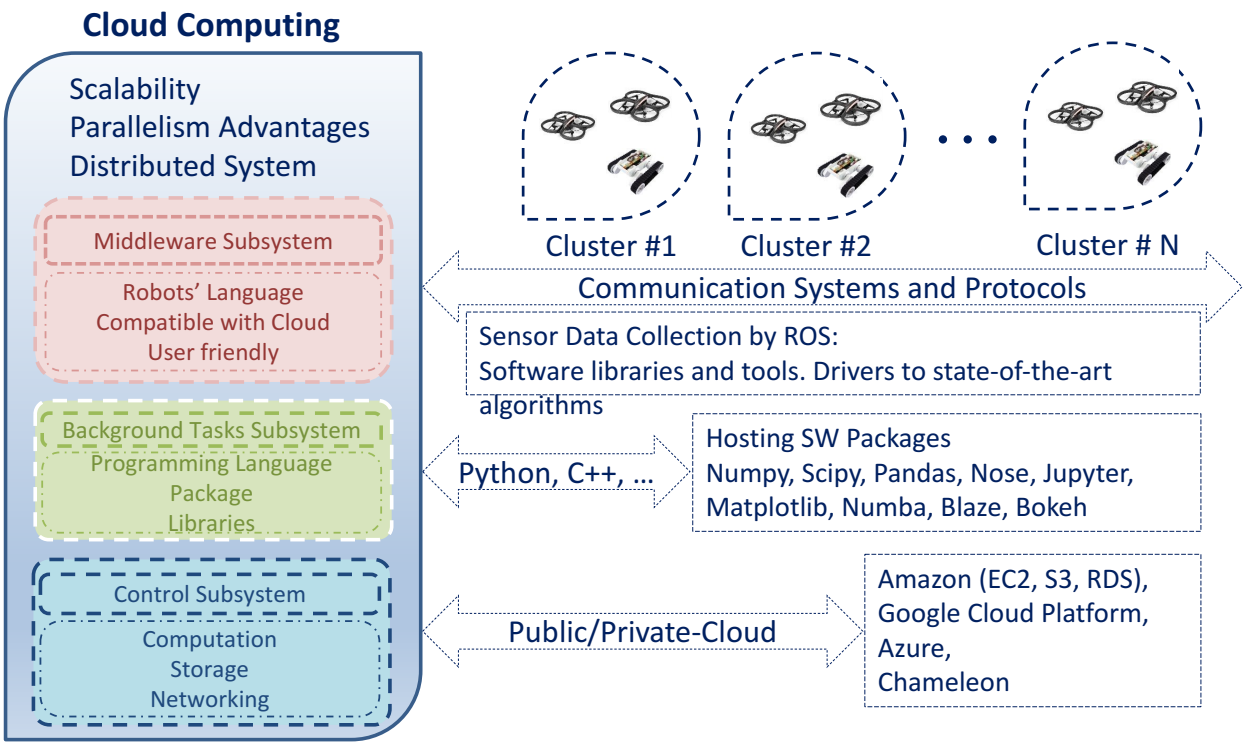


Figure 6.1: Architecture of the proposed framework to host applications in cloud.

implementing new applications for networked robotics.

Several researchers have begun to study the architecture of cloud technologies in robotic applications. For instance, a research group at Singapore ASORO laboratory announced a software framework that merges the scalability and parallelism advantages of cloud computing with large-scale environments of robots. The system is implemented through Hadoop clusters and ROS communication networks [4]. The DustBot project [132] is an example of such a sophisticated network within a robotic system. In this work, two types of robots perform cooperative tasks while benefitting from the use of an external sensory system. Two tasks have been implemented within the system: garbage collection and street cleaning (sweeping). Z. Du et. al. [28] introduced the concept of Robot as a Service and the initial Robot Cloud Center system. To the best of our knowledge, substantial works have only a few distinct design directions and implementations, especially in the area of cloud robotics, that utilize emerging advanced technologies in the cloud research area, such as OpenStack `https://www.openstack.org` and Docker `https://www.docker.com`.

The major aspect of the research presented here expounds on the creation of a software archi-

tecture that will enable large-scale systems of autonomous agents, including robots, in a heterogeneous environment, publishing sensor data throughout the cloud and also capturing data for on demand processing and post processing for computationally intense algorithms. The information flows from robots to instances within the cloud servers and back to the robots after analytical work is performed in the cloud. This architecture includes the advantage of ROS handling the modular communication mechanism among the nodes. More specifically, ROS provides the distributed communication for not only the multi-types of robots but also for the cloud servers. The elastic properties of a cloud environment have the potential to provide a scalable system. IaaS provides the governor of the software and hardware, which are needed for this scalable system. OpenStack accomplishes the management of the provisioning for the resources in the cloud environment. In the proposed architecture, Hadoop is used as the powerful cloud-based data management tool that serves to store the sensor data.

The rest of this chapter is organized as follows. First, we outline various applications of the concepts required for cloud robotics and how cloud computing can overcome critical challenges in large-scale robotics. We will also categorize the robots' tasks in order to get the most benefit from the cloud. Next, we describe the proposed cloud robotics architecture, and elaborate on three key enabling subsystems. Then, we present the implementation of the system, including two types of robots, the Kobuki turtlebot 2 as a ground robot and the Parrot Bebop as an aerial robot. We address technical challenges in designing and operating these cloud robotics architectures. In section five the advantages of the proposed software architecture for cooperative robotics are described in detail, focusing on the use of OpenStack. Finally, we conclude and summarize this chapter.

6.2 Cloud Functionality and the Robots Tasks

Networked robotics, like standalone robots, express intrinsic native constraints. For instance, robots have smaller computer architecture for mobile purposes leading to inadequate computing capability, since all computations must be conducted onboard the robots. Data access is also constrained to the limited storage of the network. Fast-developing wireless communications and recent

cloud computing technologies can be utilized to overcome some of these restrictions through the concept of cloud robotics. This creates a computationally more intelligent, well-organized and less expensive robotic network. To create this network, important robotic tasks are categorized based on the types of cloud resources needed. Examples of robotic tasks include: obstacle avoidance, vision processing, localization, path planning and environment mapping. From a practical point of view, having all tasks performed by multi agents or swarms of robots imposes significant down sides such: dedicated power supplies, good shock protection for hard disk, cost limitations, duplication and redundancy efforts, and solving networking problems. Cloud environments overcome these unnecessary and inefficient concerns by shifting the paradigm of computing resources off the physical robots. In this platform, we propose an infrastructure that includes computing, storage and a network in the datacenter. To break down the objects, we categorized these tasks as follows:

- The computing resources (VM or vDC) allocated for each agent
- Storage resources for Data Analytic research activities (vSLAM, world-maps, etc.)
- Network resources for swarm performing cooperative missions (shared information)

By utilizing this order of tasks, each node of the system is treated with respect to all three aspects: computation, storage, and networking. The next section includes a description of the proposed architecture, which was developed to handle all the aforementioned tasks simultaneously.

6.3 Proposed Software Architecture

A cloud-based architecture for large-scale autonomous robots has been proposed in figure 6.1. The architecture consists of three subsystems: (1) Middleware Subsystem, which can be counted as the main carrier for the platform; (2) Background Tasks Subsystem, which is the framework for batch processing including, software packages; and, (3) Control Subsystem, which is the brain of the platform. However, the subsystems are ordered horizontally, and the computation, storage, and networking tasks are processed vertically. Serialization of the three latter functions is processed in the following order: Networking, Storage, and Computation. The order of the storage and

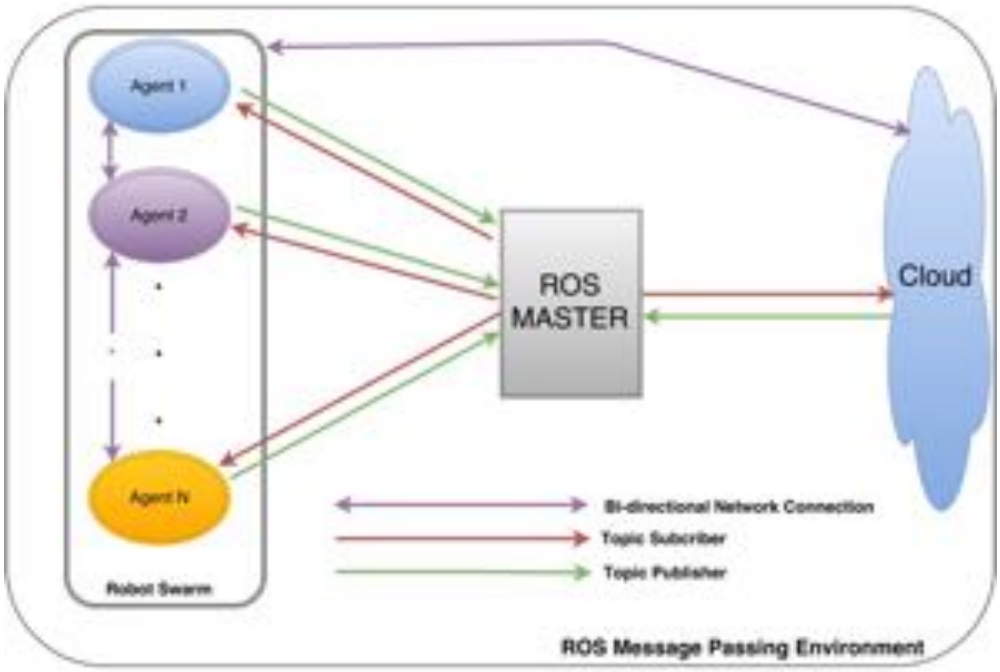


Figure 6.2: Illustration of the ROS messaging in the platform.

computation tasks is debatable, however, we suggest storage come first, since provisioning the control resources is more valuable for intense computational tasks. In the next three sections of this chapter, we will dive into the details of the subsystems:

6.3.1 Middleware Subsystem

The first subsystem provides the compatibility between robots and protocols of the communication networks. This subsystem is implemented at the lowest level of the system, connecting the robots and the cloud platform with bidirectional communication. In addition, the middleware subsystem provides all nodes of the platform with the ability to access all types of sensor data. To do so, ROS plays a large role in the first layer of our system. Figure 6.2 illustrates data communication network in the proposed system. Our proposed system requires a reliable communication network that allows many agents to pass information back and forth to each other quickly and easily. ROS uses a data passing model based on the publisher-subscriber relationship. This relationship makes sending and receiving data intuitive for the software developers writing programs in ROS.

For example, if program A requires information from program B, program B would publish this

data to the ROS master node, while program A would subscribe to the data and retrieve it when available from the ROS master node. This is a direct reason for ROS's modularity and immunity to dynamic network infrastructure, where communication reliability is poor or intermittent. ROS is able to handle these situations because it is made up of many functional nodes, or a set of processes that are publishing and subscribing to the ROS master node. Each node is only responsible for a certain task, which correlates to the developer's choice of responsibilities for each node. Consequently, if each node's functionality is a small enough portion of the whole, the loss of one or two nodes only effects a small portion of the system's functional processes and will not degrade the entire system to failure, increasing its robustness. However, ROS is a common language among multi robots, effectively used by/for critical resources among threads [22], [23] which can control the transient behaviors of the systems.

ROS MULTIMASTER-FKIE: Furthering communication reliability is an ROS package named MULTIMASTER-FKIE. This package allows more that one ROS Master Node to be run within the same ROS network. It works by syncing each ROS Master to a heartbeat. The heartbeat allows the system to know when an ROS Master has dropped from the network, and waits for the heartbeat to show up again. This idea is similar to separating the system out into nodes; the more you break your system up, the more resilient it becomes. Using ROS MULTIMASTER-FKIE gives software developers an easy way to add robustness to the system.

6.3.2 Background Tasks Subsystem

This subsystem provides the framework for performing/conducting the batch processing from a software point of view, including programming language, packages and libraries. The goal of this subsystem is to feed the control subsystem with required data. Handling the information is carried on in this stage. Hence, powerful programming enriches the system with less bugs and faster processing time. The useful programming languages in this area are Python, C++ and Java. And, among them, Python is more common these days because of its power and free libraries and packages such as Anaconda. Anaconda includes Python, SciPy, NumPy, Pandas, IPython,

Matplotlib, Numba, Blaze and Boken, all useful for computation and mathematical programming. Furthermore, the Hadoop Map/Reduce framework for conducting/performing the batch processing of sensor data is efficient for data storage and processing in this step. We describe the Hadoop-MapReduce implementation later in this chapter.

6.3.3 Control Subsystem

The final subsystem is for controlling the three aforementioned tasks: Computation, Storage and Networking. IaaS is chosen for the cloud service since we want to take advantage of scalable hardware and software for managing our hardware based on the required computational needs. Hadoop is installed on top of the platform to capture the data and podcast it for the appropriate agent. A Hadoop Distributed File System (HDFS) provides the powerful capability needed to handle the huge amount of information that has been acquired and store it in the proposed platform. HDFS also allows the use of Google's map-reduce algorithm to search and find the specific data needed. The architecture gets the advantage of the Hadoop framework, which allows for the distributed processing of large data sets across clusters of agents using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage). Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of agents is able to be directly monitored in the third subsystem.

6.4 Implementation

6.4.1 Connecting Multiple agents (Kobuki and Bebop)

ROS is an open source software developing platform for any robotic system. ROS includes many powerful packages, functionality wise, which are available for free and legal to use as the developer chooses. With that, the Kobuki ROS package and Bebop ROS driver were used in order to operate each agent in ROS. These packages output data topics pertaining to the agents' angular velocities, linear velocities, camera data, control topics, and many other state topics. This data is then used

to perform various co-operative tasks. The Kobuki agents along with the Bebop drones use Wi-Fi communication and a network router to achieve bi-directional sharing of data. Utilizing ROS multimaster fkie, each agent is made an a ROS Master, allowing for each robot to pass in and out of the Wi-Fi communication network without fatal communication or system errors.

6.4.2 Connecting Each Agent to the Cloud Using ROS

The final piece of the network layer for the system is the communication between each agent and the cloud. Each Kobuki relies on an ODroid microprocessor that runs Samsung Exynos5422 Cortex-A15 2Ghz and Cortex-A7 Octa core CPUs with 2 Gigs of RAM. This microprocessor runs an Android version of Linux and is, essentially, a credit card sized Linux computer. Despite its impressive specifications, the ODroid falls short with computationally heavy functionality, such as Simultaneous Localization and Mapping (SLAM). To solve this issue a cluster of servers running the OpenStack cloud computing software are used in order to perform the necessary resource demanding functions such as image processing with SLAM. OpenStack cloud computing software adds dynamic resource scheduling functionality to the servers. This means processing resources such as hard disk space and RAM can be allocated in real time according to the demands on the server cluster. By doing this, resources become elastic and power is saved by sending unused servers into low power mode when they are not being used. OpenStack allows the user to create an instance, which is a virtual machine that has a certain allocation of RAM and hard disk space, and allocate a floating IP address to the instance. Each agent is wirelessly connected to the cloud through a wireless router, allowing bi-directional communication. With all of the agents on the same network, ROS is able to pass agent data to the cloud by using the instance floating IP.

6.4.3 Hadoop and ROS Interaction

The robotic agents each publish their own image topics consisting of the video feed. The images are collected and published as a topic in the ROS network. The Hadoop cluster and the robots, land and air, are connected through this ROS network allowing the transfer of topics from robots to the

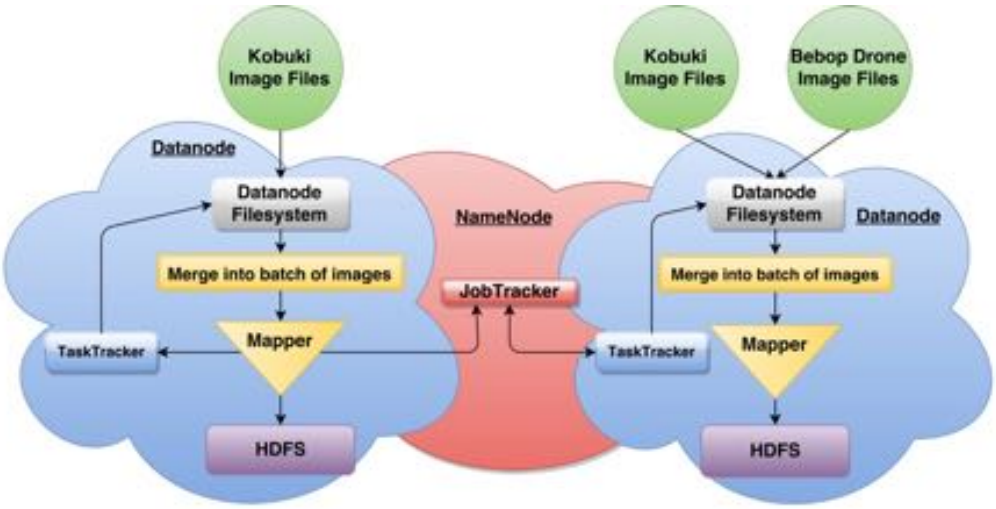


Figure 6.3: Illustration of the ROS messaging in the platform.

cloud Hadoop cluster. A collection of images are aggregated to create one file to be sent into the HDFS. The library HIPI, Hadoop image processing interface, allows for images to be concatenated into one file for the Hadoop distributed filesystem, HDFS, and the created image bundles can be manipulated by OpenCV a popular image processing library. OpenCV is first used to convert the ROS images into a format that can be modified by a script. Next, the OpenCV software can perform modifications to the image, such as drawing on the image, given its pixel location. This can be used to identify obstacles or other irregularities the robots may see. The master node of the Hadoop cluster, or the name-node, has a process called the JobTracker which handles the jobs reqauested by the user. This job tracker assigns the map and/or relays tasks to other nodes, mainly data nodes, for execution. Another process, the TaskTracker, is running on the corresponding data nodes of the cluster. The TaskTracker executes the map or reduce algorithm. Once finished, the TaskTracker sends the status back to the JobTracker. The cluster name-node then sends the job of mapping the input files into {key:value} pairs, which is then output into the HDFS from the data-node.

6.4.4 Storing Images

Since we want the majority of data processing off the robots, we add the HIPI (`HIPI-Hadoophttp://hipi.cs.virginia.edu/`) software to the cloud instances. Therefore, the datanodes are running their own small ROS program saving images onto a folder located within the datanode. Once a request is received to map the images from the JobTracker, an HIPI program will create the bundled images file and run the mapper as well. The results are added to the HDFS creating a set of images which can be manipulated and analyzed with openCV.

6.5 Software Architecture and Cooperative Robotics

Two hot issues in cooperative robotics are (1) reliable networked robotic systems and (2) formal models and methods for cooperative tasks. Cooperative robotics requires data sharing, cooperative perception, and collective intelligence. While networked robotics suffer from resource, information, learning, and communication constraints, the proposed architecture has the advantage of OpenStack-Nova, which has a strong networked ecosystem for provisioning elastic computational and storage resources. OpenStack-Neutron serves as a heavy-duty networking tool for handling the communication among the systems and, to finally overcome associated security challenges, by persistently controlling the virtual machines (VM).

The proposed platform consists of the coupled system comprised of the OpenStack and Hadoop, refer to figure 6.1 for the illustration. OpenStack functions as the shared data center that can be controlled through the dashboard as a user interface. In addition, the decision to offload a specific task requires a unified framework that can handle a list of complex issues. This important task is handled by a Hadoop-MapReduce system. First, the offloading strategy should consider various factors, including the amount of data exchanged, and the delay deadline to complete the task. Second, the decision should also consider whether it is more advantageous to execute the task within the group of networked robots hardware, given the presence of cloud resources and computation requirements. Finally, given a pool of cloud resources spread across different data centers, it is a challenge to allocate virtual machines optimally to execute the offloaded task and to

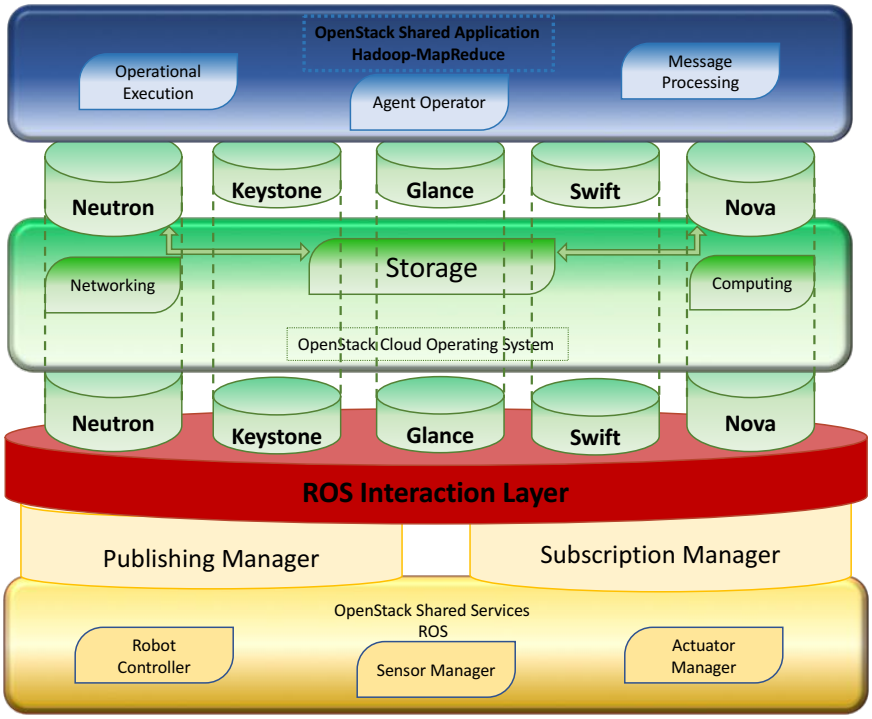


Figure 6.4: Illustration of the Software Architecture for implementing the Cooperative Robots on OpenStack.

manage live VM migrations. The other services by OpenStack facilitate solving the problem. For instance, Glance holds the created disk images, allowing for quick provisioning of new instances with all the required software in the saved images. Keystone serves as an access point for the other OpenStack services with authentication and authorization between the endpoints of each service. Swift implements a RESTful API to send commands for storing and receiving data from the OpenStack through HTTP requests. The illustration of this software architecture for cooperative robots is depicted in figure 6.4.

To overcome the communication challenge, the choice of standalone or cloud execution depends on the sensitivity to delay of the task. The communication delay introduced in sending the computation request to the cloud has to be factored into each decision. Packet delivery failures and communication outages are inherent in any wireless communication system. The additional increase in failure rate depends on the network topology. The use of the IaaS cloud as a super node in the network effectively controls this rate.

Trust and security issues are major considerations in cloud robotics. The VM environment must be secure from outside interference. A malicious VM can subtly sabotage an important task without the robot being aware of the damage. A robot needs to trust the VM is not compromised in order to launch task delegation on a public cloud, especially when the computation and network traffic have monetary costs. The computing environments in the cloud should be verified by a user or a trusted party. Confidential data may be stored in public cloud storage systems, while logically private to cloned devices. Therefore, strong integrity and confidentiality protection are needed to secure application data.

All the aforementioned capabilities make a system, consisting of a multitude of networked robots and other devices, which, as a whole, is capable of interacting with the environment through the use of perception and action for the performance of tasks. Furthermore, two current technologies for cooperation of robots, localization and learning, have shown an ability to be implemented in the software architecture.

Localization for mobile robots is a basic requirement, when it comes to cooperative spatial perception [3]. Meanwhile, the localization problem is predominantly addressed together with mapping, which leads to the paradigm of Simultaneous Localization and Mapping (SLAM) [10]. Most current approaches to SLAM are based on probabilistic (Bayesian) approaches, typically employing Kalman filter methods, particle filters, or expectation maximization techniques. While SLAM is well established for 2D mapping by single land robots, multi-robot mapping is considered to be a very important, but also still largely an open problem. In a cooperative robotics context, Petri nets have been used for modeling multi-robot plans, and a multi-robot coordination algorithm for environment exploration.

Learning can be successfully applied in behavior-based systems to adapt task assignment in heterogeneous robot teams, dealing with individual robot capabilities that change over time. Unsupervised learning methods, such as evolutionary algorithms, are popular for multi-robot task optimization.

Having the capacity to implement both localization and learning technologies enriches the soft-

ware architecture to implement formal models and methods for cooperative robots. These methods include, but are not limited to: logic-based knowledge representation and planning, decentralized decision-making, graph-based control methods, game theory, Bayesian networks, discrete and hybrid system models, or models of natural systems applicable to robotics. Therewith, the venue is provided for hosting the cooperative planning and execution, cooperative perception, and cooperative learning and evaluation.

6.6 Conclusion

A software architecture in a cloud environment consisting of three subsystems has been proposed. It has been implemented on an IaaS platform using OpenStack to add scalability to the system. By using ROS Multimaster FKIE the distributed platform can correctly serve a heterogeneous large-scale swarm of autonomous robots that have encountered serious problems, for instance if a robot agent loses communication. Introducing any type of robot just entails installing the related package onto the robot and network. Finally, with the advantage of Hadoop, data storage is carried out in a series of packages, allowing for large-scale processing of the massive amount of data received from the robotic agents. The proposed architecture is reliable, scalable, and powerful in the three functions discussed: Computing, Storage, and Networking. Hadoop–MapReduce provides the appropriate framework in cloud to process and handle these tasks. Finally, the advantages of the proposed software architecture for cooperative robotics with an authentic shared data center for communication and a trustworthy platform have been presented.

BIBLIOGRAPHY

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Sos Agaian, Paul Rad, Rahul Rajendran, and Karen Panetta. A novel technique to enhance low resolution ct and magnetic resonance images in cloud. In *Smart Cloud (SmartCloud), IEEE International Conference on*, pages 73–78. IEEE, 2016.

[3] H Levent Akin, Andreas Birk, Andrea Bonarini, Gerhard Kraetzschmar, Pedro Lima, Daniele Nardi, Enrico Pagello, Monica Reggiani, Alessandro Saffiotti, Alberto Sanfeliu, et al. Two "hot issues" in cooperative robotics: network robot systems, and formal models and methods for cooperation. *A white paper, EURON Special Interest Group on Cooperative Robotics*, 2008.

[4] Rajesh Arumugam, Vikas Reddy Enti, Liu Bingbing, Wu Xiaojun, Krishnamoorthy Baskaran, Foong Foo Kong, A Senthil Kumar, Kang Dee Meng, and Goh Wai Kit. Davinci: A cloud computing framework for service robots. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3084–3089. IEEE, 2010.

[5] Muhammad Zubair Asghar, Aurangzeb Khan, Shakeel Ahmad, and Fazal Masud Kundi. A review of feature extraction in sentiment analysis. *Journal of Basic and Applied Scientific Research*, 4(3):181–186, 2014.

[6] Mehrab Ghanat Bari, Fatemeh Ghanat Bari, and Jianqiu Zhang. The positive and random impulse noise reduction using ann and gaussian recursive filter. In *World Automation Congress (WAC), 2014*, pages 763–767. IEEE, 2014.

[7] Luiz Andre Barroso. Warehouse-scale computing: Entering the teenage decade. In *ACM SIGARCH Computer Architecture News*, volume 39. ACM, 2011.

[8] Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle. The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis lectures on computer architecture*, 8(3):1–154, 2013.

[9] Anton Beloglazov and Rajkumar Buyya. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1366–1379, 2013.

[10] Patrick Benavidez, Mohan Muppidi, Paul Rad, John J Prevost, Mo Jamshidi, and Lutcher Brown. Cloud-based realtime robotic visual slam. In *Systems Conference (SysCon), 2015 9th Annual IEEE International*, pages 773–777. IEEE, 2015.

[11] James O Benson, John J Prevost, and Paul Rad. Survey of automated software deployment for computational and engineering research. In *Systems Conference (SysCon), 2016 Annual IEEE*, pages 1–6. IEEE, 2016.

[12] Hrishikesh Bhaumik, Siddhartha Bhattacharyya, Mausumi Das Nath, and Susanta Chakraborty. Hybrid soft computing approaches to content based video retrieval: A brief review. *Applied Soft Computing*, 46:1008–1029, 2016.

[13] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[14] Marina Blanton and Mehrdad Aliasgari. Secure outsourced computation of iris matching. *Journal of Computer Security*, 20(2-3):259–305, 2012.

[15] Andrew Boles and Paul Rad. Voice biometrics: Deep learning-based voiceprint authentication system. In *System of Systems Engineering Conference (SoSE), 2017 12th*, pages 1–6. IEEE, 2017.

[16] Peter Burnap and Matthew Leighton Williams. Hate speech, machine classification and statistical modelling of information flows on twitter: Interpretation and communication for policy decision making. 2014.

[17] Andrew Byde, Mathias Sallé, and Claudio Bartolini. Market-based resource allocation for utility data centers. *HP Lab, Bristol, Technical Report HPL-2003-188*, 2003.

[18] Erik Cambria and Amir Hussain. *Sentic computing: a common-sense-based framework for concept-level sentiment analysis*, volume 1. Springer, 2015.

[19] Octavia Camps, Mengran Gou, Tom Hebble, Srikrishna Karanam, Oliver Lehmann, Yang Li, Richard J Radke, Ziyan Wu, and Fei Xiong. From the lab to the real world: re-identification in an airport camera network. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(3):540–553, 2017.

[20] Iván Cantador, Peter L Brusilovsky, and Tsvi Kuflik. Second workshop on information heterogeneity and fusion in recommender systems (hetrec2011). 2011.

[21] Cunjian Chen, Antitza Dantcheva, and Arun Ross. An ensemble of patch-based subspaces for makeup-robust face recognition. *Information Fusion*, 32:80–92, 2016.

[22] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.

[23] IEEE RAS Technical Committee et al. Ieee society of robotics and automation's technical committee on: Networked robots, 2012.

[24] Md Daiyan, Dr SK Tiwari, Manish Kumar, and M Aftab Alam. A literature review on opinion mining and sentiment analysis. *International Journal of Emerging Technology and Advanced Engineering*, 5(4), 2015.

[25] Kushal Dave, Steve Lawrence, and David M Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, pages 519–528. ACM, 2003.

[26] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[27] Changxing Ding and Dacheng Tao. A comprehensive survey on pose-invariant face recognition. *ACM Transactions on intelligent systems and technology (TIST)*, 7(3):37, 2016.

[28] Zhihui Du, Weiqiang Yang, Yinong Chen, Xin Sun, Xiaoying Wang, and Chen Xu. Design of a robot cloud center. In *Autonomous Decentralized Systems (ISADS), 2011 10th International Symposium on*, pages 269–275. IEEE, 2011.

[29] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[30] Nationality Dutch. *Evert de Haan*. PhD thesis, University of Groningen, 2011.

[31] Morad Eghbal and Mehdi Shadaram. Tandem-modulator generated w-band ocdma radio-over-fiber system. In *Transparent Optical Networks (ICTON), 2016 18th International Conference on*, pages 1–3. IEEE, 2016.

[32] Erik Elmroth, Fermin Galan Marquez, Daniel Henriksson, and David Perales Ferrera. Accounting and billing for federated cloud infrastructures. In *Grid and Cooperative Computing, 2009. GCC'09. Eighth International Conference on*, pages 268–275. IEEE, 2009.

[33] Zekeriya Erkin, Martin Franz, Jorge Guajardo, Stefan Katzenbeisser, Inald Lagendijk, and Tomas Toft. Privacy-preserving face recognition. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 235–253. Springer, 2009.

[34] Qingxiang Feng, Chun Yuan, Jeng-Shyang Pan, Jar-Ferr Yang, Yang-Ting Chou, Yicong Zhou, and Weifeng Li. Superimposed sparse parameter classifiers for face recognition. *IEEE transactions on cybernetics*, 47(2):378–390, 2017.

[35] Terrence Fong, Illah Nourbakhsh, and Kerstin Dautenhahn. A survey of socially interactive robots. *Robotics and autonomous systems*, 42(3):143–166, 2003.

[36] Tim Forell, Dejan Milojicic, and Vanish Talwar. Cloud management: Challenges and opportunities. In *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*, pages 881–889. IEEE, 2011.

[37] Ian Foster and Carl Kesselman. *The Grid 2: Blueprint for a new computing infrastructure*. Elsevier, 2003.

[38] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE'08*, pages 1–10. Ieee, 2008.

[39] Longxiang Gao, Tom H Luan, Shui Yu, Wanlei Zhou, and Bo Liu. Fogroute: Dtn-based data dissemination model in fog computing. *IEEE Internet of Things Journal*, 4(1):225–235, 2017.

[40] W Gentzsch and B Yenier. The ubercloud experiment: tech. comp. in the cloud-2nd compendium of case studies. Technical report, Tech. rep., Tabor Communications, Inc, 2014.

[41] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber. Learning precise timing with lstm recurrent networks. *Journal of machine learning research*, 3(Aug):115–143, 2002.

[42] Mehrab Ghanat Bari, Xuepo Ma, and Jianqiu Zhang. Peaklink: a new peptide peak linking method in lc-ms/ms using wavelet and svm. *Bioinformatics*, 30(17):2464–2470, 2014.

[43] Mehrab Ghanat Bari, Nelson Ramirez, Zhiwei Wang, and Jianqiu Michelle Zhang. Mzdasoft: a software architecture that enables large-scale comparison of protein expression levels over multiple samples based on liquid chromatography/tandem mass spectrometry. *Rapid Communications in Mass Spectrometry*, 29(19):1841–1848, 2015.

[44] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[45] Atul Gohad, Nanjangud C Narendra, and Parathasarthy Ramachandran. Cloud pricing models: A survey and position paper. In *Cloud Computing in Emerging Markets (CCEM), 2013 IEEE International Conference on*, pages 1–8. IEEE, 2013.

[46] Shaogang Gong, Marco Cristani, Shuicheng Yan, and Chen Change Loy. *Person re-identification*, volume 1. Springer, 2014.

[47] Kevin Guinn, Peyman Najafirad, and Bharath Vasudevan. System and method for the implementation of an adaptive cache policy in a storage controller, April 18 2005. US Patent App. 11/108,521.

[48] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 1735–1742. IEEE, 2006.

[49] James Hamilton. Cost of power in large-scale data centers. *Blog entry dated*, 11:28, 2008.

[50] David Y Hancock, Craig A Stewart, Jeremy Fischer, John Michael Lowe, Paul Rad, and Matthew Vaughn. Resource management from hpc to the cloud: Do you manage resources or do they manage you? 2016.

[51] Colin Harrison, Barbara Eckman, Rick Hamilton, Perry Hartswick, Jayant Kalagnanam, Jurij Paraszczak, and Peter Williams. Foundations for smarter cities. *IBM Journal of Research and Development*, 54(4):1–16, 2010.

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[54] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[55] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[56] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[57] Alexander Hogenboom, Bas Heerschop, Flavius Frasincar, Uzay Kaymak, and Franciska de Jong. Multi-lingual support for lexicon-based sentiment analysis guided by semantics. *Decision support systems*, 62:43–53, 2014.

[58] Ines Houidi, Marouen Mechtri, Wajdi Louati, and Djamal Zeghlache. Cloud service delivery across multiple cloud platforms. In *Services Computing (SCC), 2011 IEEE International Conference on*, pages 741–742. IEEE, 2011.

[59] Jeremy Hsu. Finding one face in a million [news]. *IEEE Spectrum*, 53(8):11–12, 2016.

[60] Guoqiang Hu, Wee Peng Tay, and Yonggang Wen. Cloud robotics: architecture, challenges and applications. *IEEE network*, 26(3), 2012.

[61] Peiyun Hu and Deva Ramanan. Finding tiny faces. *arXiv preprint arXiv:1612.04402*, 2016.

[62] Gary B Huang, Honglak Lee, and Erik Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2518–2525. IEEE, 2012.

[63] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[64] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[65] Katarzyna Janocha and Wojciech Marian Czarnecki. On loss functions for deep neural networks in classification. *arXiv preprint arXiv:1702.05659*, 2017.

[66] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.

[67] Koji Kamei, Shuichi Nishio, Norihiro Hagita, and Miki Sato. Cloud networked robotics. *IEEE Network*, 26(3), 2012.

[68] SM Azharul Karim, John J Prevost, and Paul Rad. Efficient real-time mobile computation in the cloud using containers. *Int. J. Com. Dig. Sys*, 5(1):21–30, 2016.

[69] Ben Kehoe, Sachin Patil, Pieter Abbeel, and Ken Goldberg. A survey of research on cloud robotics and automation. *IEEE Transactions on automation science and engineering*, 12(2):398–409, 2015.

[70] Timothy L Keiningham, Bruce Cooil, Tor Wallin Andreassen, and Lerzan Aksoy. A longitudinal examination of net promoter and firm revenue growth. *Journal of Marketing*, 71(3):39–51, 2007.

[71] Brian Kelley, John J Prevost, Paul Rad, and Aqsa Fatima. Securing cloud containers using quantum networking channels. In *Smart Cloud (SmartCloud), IEEE International Conference on*, pages 103–111. IEEE, 2016.

[72] Terence Kelly et al. Utility-directed allocation. In *First Workshop on Algorithms and Architectures for Self-Managing Systems*, volume 20, 2003.

[73] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4873–4882, 2016.

[74] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[75] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[76] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[77] James Kuffner. Cloud-enabled humanoid robots. In *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on, Nashville TN, United States, Dec.*, 2010.

[78] JJ Kuffner and Cloud-Enabled Robots. Ieee-ras international conference on humanoid robotics, 2010.

[79] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[80] Ravi Kumar and Peyman Najafirad. System and method for managing hung cluster nodes, April 25 2005. US Patent App. 11/113,759.

[81] Ravi Kumar and Peyman Najafirad. System and method for managing node resets in a cluster, January 31 2006. US Patent App. 11/343,777.

[82] Alicja Kwaśniewska, Jacek Rumiński, and Paul Rad. Deep features class activation map for thermal face detection and tracking. In *Human System Interactions (HSI), 2017 10th International Conference on*, pages 41–47. IEEE, 2017.

[83] Erik Learned-Miller, Gary B Huang, Aruni RoyChowdhury, Haoxiang Li, and Gang Hua. Labeled faces in the wild: A survey. In *Advances in face detection and facial image analysis*, pages 189–248. Springer, 2016.

[84] GBHE Learned-Miller. Labeled faces in the wild: Updates and new reporting procedures. *Technical Report UM-CS-2014-003*, 2014.

[85] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[86] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[87] Gunho Lee. *Resource allocation and scheduling in heterogeneous cloud environments*. University of California, Berkeley, 2012.

[88] Jang-Won Lee, Ravi R Mazumdar, and Ness B Shroff. Downlink power allocation for multi-class wireless systems. *IEEE/ACM Transactions on Networking*, 13(4):854–867, 2005.

[89] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5325–5334, 2015.

[90] Zhifeng Li, Dihong Gong, Xuelong Li, and Dacheng Tao. Aging face recognition: A hierarchical learning model based on local patterns selection. *IEEE Transactions on Image Processing*, 25(5):2146–2154, 2016.

[91] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[92] Jingtuo Liu, Yafeng Deng, Tao Bai, Zhengping Wei, and Chang Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv preprint arXiv:1506.07310*, 2015.

[93] Chaochao Lu and Xiaoou Tang. Surpassing human-level face verification performance on lfw with gaussianface. *arXiv preprint arXiv:1404.3840*, 2014.

[94] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM, 2013.

[95] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[96] Marian Mihailescu and Yong Meng Teo. Dynamic resource pricing on federated clouds. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 513–517. IEEE Computer Society, 2010.

[97] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[98] Seyed Ali Miraftabzadeh, Paul Rad, Kim-Kwang Raymond Choo, and Mo Jamshidi. A privacy-aware architecture at the edge for autonomous real-time identity re-identification in crowds. *IEEE Internet of Things Journal*, 2017.

[99] Seyed Ali Miraftabzadeh, Paul Rad, and Mo Jamshidi. Efficient distributed algorithm for scheduling workload-aware jobs on multi-clouds. In *System of Systems Engineering Conference (SoSE), 2016 11th*, pages 1–8. IEEE, 2016.

[100] Seyed Ali Miraftabzadeh, Paul Rad, and Mo Jamshidi. Distributed algorithm with inherent intelligence for multi-cloud resource provisioning. In *Intelligent Decision Support Systems for Sustainable Computing*, pages 77–99. Springer, 2017.

[101] Seyed Ali Miraftabzadeh, Paul Rad, and Mo Jamshidi. Temporal face embedding as biometric tokenization for decentralized iot. In *IEEE ICMLA 2018 Conference proceedings*. Institute of Electrical and Electronics Engineers, 2018.

[102] Seyed Ali Miratabzadeh, Nicolas Gallardo, Nicholas Gamez, Karthikpai Haradi, Abhijith R Puthussery, Paul Rad, and Mo Jamshidi. Cloud robotics: A software architecture: For heterogeneous large-scale autonomous robots. In *World Automation Congress (WAC), 2016*, pages 1–6. IEEE, 2016.

[103] Tom M Mitchell et al. Machine learning. wcb, 1997.

[104] Mohan Muppidi, Paul Rad, Sos S Agaian, and Mo Jamshidi. Container based parallelization for faster and reliable image segmentation. In *Imaging Systems and Techniques (IST), 2015 IEEE International Conference on*, pages 1–6. IEEE, 2015.

[105] Peyman Najafirad. *YottaCloud: A cloud architecture for secure big image analytics*. PhD thesis, The University of Texas at San Antonio, 2016.

[106] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.

[107] Arman Khadjeh Nassirtoussi, Saeed Aghabozorgi, Teh Ying Wah, and David Chek Ling Ngo. Text mining for market prediction: A systematic review. *Expert Systems with Applications*, 41(16):7653–7670, 2014.

[108] Hong-Wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 343–347. IEEE, 2014.

[109] Margarita Osadchy, Yann Le Cun, and Matthew L Miller. Synergistic face detection and pose estimation with energy-based models. *Journal of Machine Learning Research*, 8(May):1197–1215, 2007.

[110] Rameswar Panda, Amran Bhuiyan, Vittorio Murino, and Amit K Roy-Chowdhury. Unsupervised adaptive re-identification in open world dynamic camera networks. *arXiv preprint arXiv:1706.03112*, 2017.

[111] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

[112] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *BMVC*, volume 1, page 6, 2015.

[113] Przemyslaw Pawluk, Bradley Simmons, Michael Smit, Marin Litoiu, and Serge Mankovski. Introducing stratos: A cloud broker service. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 891–898. IEEE, 2012.

[114] Ranjith Purushothaman and Peyman Najafirad. Cluster failover from physical node to virtual node, November 14 2003. US Patent App. 10/713,379.

[115] Lianyong Qi, Wanchun Dou, and Xuyun Zhang. An inverse collaborative filtering approach for cold-start problem in web service recommendation. In *Proceedings of the Australasian Computer Science Week Multiconference*, page 46. ACM, 2017.

[116] Darren Quick and Kim-Kwang Raymond Choo. Big forensic data reduction: digital forensic images and electronic evidence. *Cluster Computing*, 19(2):723–740, 2016.

[117] Darren Quick and Kim-Kwang Raymond Choo. Big forensic data management in heterogeneous distributed systems: Quick analysis of multimedia forensic data. *Software: Practice and Experience*, 47(8):1095—1109, 2017.

[118] PAUL RAD. A software defined networking architecture for high performance clouds1 paul rad 2, mo jamshidi 2, gilad berman 3, and john j. prevost 2. 2015.

[119] Paul Rad, Sos Agaian, Mehdi Roopaei, and Saeed Sedighi. Image enhancement via cloud cascade control based sub-image-clipped histogram equalization. In *Smart Cloud (SmartCloud), IEEE International Conference on*, pages 69–72. IEEE, 2016.

[120] Paul Rad, Rajendra V Boppana, Palden Lama, Gilad Berman, and Mo Jamshidi. Low-latency software defined network for high performance clouds. In *System of Systems Engineering Conference (SoSE), 2015 10th*, pages 486–491. IEEE, 2015.

[121] Paul Rad, AT Chronopoulos, P Lama, Pranitha Madduri, and Cameron Loader. Benchmarking bare metal cloud servers for hpc applications. In *Cloud Computing in Emerging Markets (CCEM), 2015 IEEE International Conference on*, pages 153–159. IEEE, 2015.

[122] Paul Rad, Van Lindberg, Jeff Prevost, Weining Zhang, and Mo Jamshidi. Zerovm: secure distributed processing for big data analytics. In *World Automation Congress (WAC), 2014*, pages 1–6. IEEE, 2014.

[123] Ioan Raicu, Yong Zhao, Catalin Dumitrescu, Ian Foster, and Mike Wilde. Falkon: a fast and light-weight task execution framework. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, page 43. ACM, 2007.

[124] Frederick F Reichheld. The one number you need to grow. *Harvard business review*, 81(12):46–55, 2003.

[125] Charles Reiss, Alexey Tumanov, Gregory R Ganger, Randy H Katz, and Michael A Kozuch. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In *Proceedings of the Third ACM Symposium on Cloud Computing*, page 7. ACM, 2012.

[126] Steffen Rendle, Dennis Fetterly, Eugene J Shekita, and Bor-yiing Su. Robust large-scale machine learning in the cloud. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1125–1134. ACM, 2016.

[127] Antonio Reyes and Paolo Rosso. On the difficulty of automatically detecting irony: beyond a simple case of negation. *Knowledge and Information Systems*, 40(3):595–614, 2014.

[128] Benny Rochwerger, David Breitgand, Eliezer Levy, Alex Galis, Kenneth Nagin, Ignacio Martín Llorente, Rubén Montero, Yaron Wolfsthal, Erik Elmroth, Juan Caceres, et al.

The reservoir model and architecture for open federated cloud computing. *IBM Journal of Research and Development*, 53(4):4–1, 2009.

[129] Mehdi Roopaei, Morad Khosravi Eghbal, Mehdi Shadaram, and Sos Agaian. Noise-free rule-based fuzzy image enhancement. *Electronic Imaging*, 2016(13):1–5, 2016.

[130] Mehdi Roopaei, Paul Rad, and Kim-Kwang Raymond Choo. Cloud of things in smart agriculture: Intelligent irrigation monitoring by thermal imaging. *IEEE Cloud Computing*, 4(1):10–15, 2017.

[131] Mehdi Roopaei, Paul Rad, and Mo Jamshidi. Deep learning control for complex and large scale cloud systems. *Intelligent Automation & Soft Computing*, 23(3):389–391, 2017.

[132] Pericle Salvini, Cecilia Laschi, and Paolo Dario. Do service robots need a driving license?[industrial activities]. *IEEE Robotics & Automation Magazine*, 18(2):12–13, 2011.

[133] Vivek Vijay Sarkale, Paul Rad, and Wonjun Lee. Secure cloud container: Runtime behavior monitoring using most privileged container (mpc). In *Cyber Security and Cloud Computing (CSCloud), 2017 IEEE 4th International Conference on*, pages 351–356. IEEE, 2017.

[134] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[135] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[136] Uday D Shet, Peyman Najafirad, and Ramesh S Rajagopalan. System and method for dynamically adjusting the caching characteristics for each logical unit of a storage array, February 22 2011. US Patent 7,895,398.

[137] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[138] Sumankumar Singh, Timothy Abels, and Peyman Najafirad. Server cluster having a virtual server, January 12 2005. US Patent App. 11/034,384.

[139] Sumankumar Singh and Peyman Najafirad. System and method for recovering from a failure of a virtual machine, July 18 2005. US Patent App. 11/183,697.

[140] Sumankumar A Singh and Peyman Najafirad. Systems and methods for accessing a shared storage network using multiple system nodes configured as server nodes, March 17 2009. US Patent 7,506,009.

[141] Yang Song, Murtaza Zafer, and Kang-Won Lee. Optimal bidding in spot instance market. In *INFOCOM, 2012 Proceedings IEEE*, pages 190–198. IEEE, 2012.

[142] Flávio Souza, Diego de Las Casas, Vinícius Flores, SunBum Youn, Meeyoung Cha, Daniele Quercia, and Virgílio Almeida. Dawn of the selfie era: The whos, wheres, and hows of selfies on instagram. In *Proceedings of the 2015 ACM on conference on online social networks*, pages 221–231. ACM, 2015.

[143] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[144] Ross Stagner. The cross-out technique as a method in public opinion analysis. *The Journal of Social Psychology*, 11(1):79–90, 1940.

[145] Chi Su, Fan Yang, Shiliang Zhang, Qi Tian, Larry S. Davis, and Wen Gao. Multi-task learning with low rank attribute embedding for person re-identification. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[146] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[147] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep convolutional network cascade for facial point detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3476–3483, 2013.

[148] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1891–1898. IEEE, 2014.

[149] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2892–2900, 2015.

[150] Smitha Sundareswaran, Anna Squicciarini, and Dan Lin. A brokerage-based approach for cloud service selection. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 558–565. IEEE, 2012.

[151] Ilya Sutskever, James Martens, George E Dahl, and Geoffrey E Hinton. On the importance of initialization and momentum in deep learning. *ICML (3)*, 28:1139–1147, 2013.

[152] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.

[153] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[154] Richard Szeliski. Locally adapted hierarchical basis preconditioning. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 1135–1143. ACM, 2006.

[155] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[156] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2), 2012.

[157] Adel Nadjaran Toosi. *On the Economics of Infrastructure as a Service Cloud Providers: Pricing, Markets and Profit Maximization*. PhD thesis, University of Melbourne, Department of Computing and Information Systems, 2015.

[158] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, 2014.

[159] George Tychogiorgos, Athanasios Gkelias, and Kin K Leung. Utility-proportional fairness in wireless networks. In *Personal indoor and mobile radio communications (PIMRC), 2012 IEEE 23rd international symposium on*, pages 839–844. IEEE, 2012.

[160] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[161] William E Walsh, Gerald Tesauro, Jeffrey O Kephart, and Rajarshi Das. Utility functions in autonomic systems. In *Autonomic Computing, 2004. Proceedings. International Conference on*, pages 70–77. IEEE, 2004.

[162] Lizhe Wang, Jiabin Zhang, Peng Liu, Kim-Kwang Raymond Choo, and Fang Huang. Spectral-spatial multi-feature based deep learning for hyperspectral remote sensing image classification. *Soft Computing*, 21:213–221, 2017.

[163] Wei Wang, Baochun Li, and Ben Liang. Towards optimal capacity segmentation with hybrid cloud pricing. In *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on*, pages 425–434. IEEE, 2012.

[164] Yandong Wen, Zhifeng Li, and Yu Qiao. Latent factor guided convolutional neural networks for age-invariant face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4901, 2016.

[165] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.

[166] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 529–534. IEEE, 2011.

[167] Hong Xu and Baochun Li. Dynamic cloud pricing for revenue maximization. *IEEE Transactions on Cloud Computing*, 1(2):158–171, 2013.

[168] Yong Xu, Zheng Zhang, Guangming Lu, and Jian Yang. Approximately symmetrical face images for image preprocessing in face recognition and sparse representation based classification. *Pattern Recognition*, 54:68–82, 2016.

[169] Yan Yan, Elisa Ricci, Ramanathan Subramanian, Gaowen Liu, Oswald Lanz, and Nicu Sebe. A multi-task learning framework for head pose estimation under target motion. *IEEE transactions on pattern analysis and machine intelligence*, 38(6):1070–1083, 2016.

[170] Jian Yang, Lei Luo, Jianjun Qian, Ying Tai, Fanlong Zhang, and Yong Xu. Nuclear norm based matrix regression with applications to face recognition with occlusion and illumination changes. *IEEE transactions on pattern analysis and machine intelligence*, 39(1):156–171, 2017.

[171] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5525–5533, 2016.

[172] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[173] Stefanos Zafeiriou, Cha Zhang, and Zhengyou Zhang. A survey on face detection in the wild: past, present and future. *Computer Vision and Image Understanding*, 138:1–24, 2015.

[174] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[175] Kuan Zhang, Jianbing Ni, Kan Yang, Xiaohui Liang, Ju Ren, and Xuemin Sherman Shen. Security and privacy in smart city applications: Challenges and solutions. *IEEE Communications Magazine*, 55(1):122–129, 2017.

[176] Linquan Zhang, Zongpeng Li, and Chuan Wu. Dynamic resource provisioning in cloud computing: A randomized auction approach. In *INFOCOM, 2014 Proceedings IEEE*, pages 433–441. IEEE, 2014.

[177] Yongshan Zhang, Jia Wu, Zhihua Cai, Peng Zhang, and Ling Chen. Memetic extreme learning machine. *Pattern Recognition*, 58:135–148, 2016.

[178] Zhenzhu Zheng and Guodong Quo. A joint optimization scheme to combine different levels of features for face recognition with makeup changes. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 3001–3005. IEEE, 2016.

[179] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

[180] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

VITA

After receiving his M.Sc. in Electrical and Computer Engineering from Sharif University of Technology, Tehran, Iran, in 2010, he involved in several projects by ERICSSON, HUAWEI, and MCI to solve real-word problems in new generation telecommunication networks. Ali's scientific and industry background mainly focus on design, develop, and apply mathematics models for complex real-word problems like as Smart Scheduling Algorithm for Multi-Cloud, Mutual-Coupling Compensation in Maxwell's equations, and Unique Neural Network Model built on Residual Layers for Extracting face information. Ali's Ph.D. research is focused on Artificial Intelligent Systems and Machine Learning for real-word applications in scale. He has been recently explored multi purposes deep learning frameworks for conceptual understanding of image and video contents.