ENABLING AVQoS FOR ADAPTIVE STREAMING IN HETEROGENEOUS NETWORK ENVIRONMENT UTILIZING NON-INTRUSIVE BANDWIDTH INFORMATION

By

LIM SU JIN

A dissertation submitted to the Department of Electrical and Electronic Engineering,
Lee Kong Chian Faculty of Engineering and Science,
Universiti Tunku Abdul Rahman,
in partial fulfillment of the requirements for the degree of
Master of Engineering Science
October 2015

ABSTRACT

ENABLING AVQoS FOR ADAPTIVE STREAMING IN HETEROGENEOUS NETWORK ENVIRONMENT UTILIZING NON-INTRUSIVE BANDWIDTH INFORMATION

Lim Su Jin

Interactive communication is becoming crucial in our daily life. The form of communication between people from anywhere becomes much easier with the use of interactive communication, such as video conference system. The real-time interactive communication often faced challenges in network stability and also costly charges for bandwidth reservation. Hence, these motivate the idea to enable adaptive streaming by adjusting the quality of the content delivery.

In order to adapt the streaming content to the changing network condition, network performance metrics are used to adjust the quality of the video streams. The network performance metrics discussed in this thesis are available bandwidth and latency. Available bandwidth is the amount of data that can be transferred over the network with the presence of cross traffic. Latency is the time taken for a packet to travel round trip from source to destination, which is often referred as round-trip delay. The quality of the video content delivery depends highly on the network available bandwidth while the audio packets are relative to the latency between network nodes.

There are two ways to obtain network performance metrics, intrusive and non-intrusive measurement. Intrusive measurement is performed by incurring the least possible traffic to the network while non-intrusive measurement tends to measure to a subset of network nodes and predicting the performance of the rest of the network. In a large scale network, performing full mesh active measurement will incur large measurement overhead. Hence, non-intrusive measurement is adopted in this thesis with the aim to reduce measurement overhead and network congestion.

In this thesis, network performance prediction algorithm is performed in quantitative as well as qualitative approach. The first part of the algorithm, which is the quantitative prediction, formulates network metrics as matrix completion problem whereby some entries are measured and some are predicted. This method made an assumption that the network performance metrics are correlated. In the later part of the algorithm, available bandwidth and latency are formed as pairs for network link quality prediction.

One of the improvements made over the conventional matrix factorization algorithm is initialization via Singular Value Decomposition (SVD) over random initialization. This pre-processing step provides better and faster algorithm convergence to local minimum with a close-to-actual starting point before the stochastic updates take place. The proposed algorithm has been experimented and shown improvement of 20% over conventional random initialization. In the enhancement part of the algorithm, supervised learning algorithm, Support Vector Machine (SVM) is proved to be applicable in network link quality prediction. Furthermore, SVM can be easily extended to include more network features such as packet loss, jitter and throughput for link quality prediction.

The proposed network performance prediction algorithm is integrated into video conference system, namely AVMEDIC. AVMEDIC is currently under research by MIMOS Berhad for medical conferencing purposes consists of 4 parties which are communicating with each other. Network resource estimation (NRE) framework is developed to adapt the quality of the video content delivery based on dynamic network condition. This is done by adjusting the bit rate of the video based on available bandwidth obtained from the proposed algorithm. Therefore, a smooth communication can be guaranteed by adjusting the quality of the content delivery.

In conclusion, NRE has been developed and integrated into video conference system which enables adaptive streaming in heterogeneous network environment. In future, the practical use of supervised learning algorithm and online learning can be applied into real-time streaming application.

ACKNOWLEDGEMENT

I have experienced the ups and downs during my master's degree study. I would never have been able to finish my master's journey without the guidance of my supervisors, friends, and support from my family.

I would like to express my deepest gratitude to my supervisors, Prof. Dr. Lee Sze Wei and Dr. Simon Lau for their guidance on networking knowledge and practical issues beyond the textbooks. Moreover, they have been very supportive throughout my thesis amendment.

I have been very fortunate to have Boon Ping as my supervisor in MIMOS, who always willing to help and shared a lot of her research experiences with me. She has continuously provided me with inspiration, encouragement and advices.

I would like to thank my friends, especially Chin Jou and Peng Shen who always willing to help and encourage me throughout my master's journey. I would also like to thank Adrian, Farid, and other workers in the lab for helping me in data collection from the video conference system.

I would also like to thank my family who were always supporting me and encouraging me with their best wishes.

FACULTY OF ENGINEERING
UNIVERSITI TUNKU ABDUL RAHMAN

Date: __________________

PERMISSION SHEET

It is hereby certified that **LIM SU JIN** (ID No: **12UEM01305**) has completed this thesis/dissertation entitled “ENABLING AVQoS FOR ADAPTIVE STREAMING IN HETEROGENEOUS NETWORK ENVIRONMENT UTILIZING NON-INTRUSIVE BANDWIDTH INFORMATION” under the supervision of Prof. Dr. Lee Sze Wei (Supervisor) from the Department of Electrical and Electronic Engineering, Faculty of Engineering and Science.

I hereby give permission to my supervisor to write and prepare a manuscript of these research findings for publishing in any form, if I did not prepare it within six (6) months time from this date, provided, that my name is included as one of the authors for this article. Arrangement of names will depend on my supervisor.

Yours truly,

____
(LIM SU JIN)

APPROVAL SHEET

This dissertation/thesis entitled "**ENABLING AVQoS FOR ADAPTIVE STREAMING IN HETEROGENEOUS NETWORK ENVIRONMENT UTILIZING NON-INTRUSIVE BANDWIDTH INFORMATION**" was prepared by LIM SU JIN and submitted as partial fulfillment of the requirements for the degree of Master of Engineering Science at Universiti Tunku Abdul Rahman.

Approved by:

____
(Prof. Dr. LEE SZE WEI)
Date:…………………..
Professor/Supervisor
Department of Electrical and Electronic Engineering
Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

DECLARATION

I hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

Name ______________________________

Date _______________________________

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

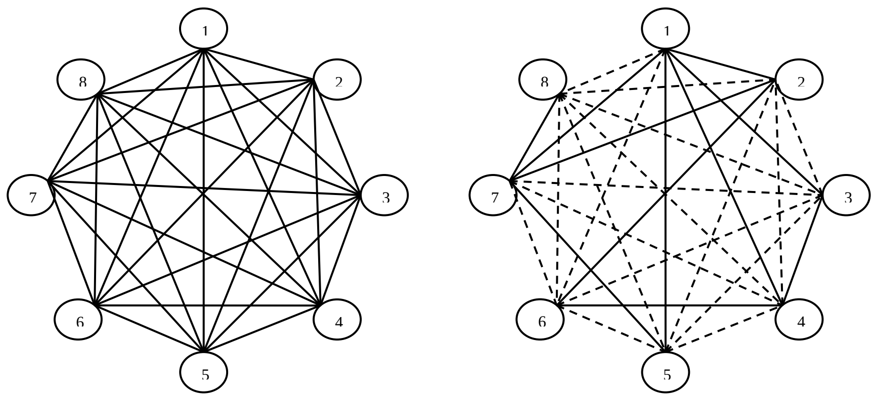
The rapid growth in data communication and information technology has lead to the increment of number of users and the usage of communication applications. This significant evolution is introducing higher demand on network bandwidth. Hence, the conventional client-server architecture often experiences congestions and instability due to server overload. As an attempt to this issue, decentralized architectures have been proposed and studied to provide better Quality of Service (QoS).

In the networking context, QoS can be measured and guaranteed by utilizing the network performance metrics more efficiently. Therefore, the knowledge of end-to-end network performance is crucial to achieve good QoS. The network performance metrics discussed in this research are available bandwidth and latency. Available network bandwidth is the amount of data that can be transferred over the network with the presence of other ongoing traffics. Latency is the time taken for a data packet to travel from a source network node to a destination node and then back to the source node, also known as the round-trip delay.

There are two main approaches to measure the network metrics, i.e. intrusive or non-intrusive measurement. Intrusive measurement is about measurement techniques that attempt to incur the least possible probing traffic but with good accuracy. Non-intrusive measurement methods meanwhile, aim to reduce measurement overheads by measuring a subset of network nodes and then predict the performance of the rest of the network. Intrusive measurement methods will introduce some probing traffics to the network while the non-intrusive ones tend to minimize the measurement overhead through prediction.

In general, some of the current key challenges in network performance prediction are:

- *Stability*: Network nodes join and leave frequently and hence affect the stability.
- *Metric diversity*: The network performance metrics (latency and available bandwidth) vary over time.
- *Costly measurement*: Intrusive measurement methods are costly due to probing traffics injected into the network.
- *Scalability*: Performing measurement in mesh network is highly infeasible with $O(n^2)$ path complexity.



(a) Intrusive measurement (b) Non-intrusive measurement

Figure 1.1 (a) Full mesh active measurement with *$O(n^2)$* complexity. (b) Non-intrusive measurement whereby the solid paths are measured and the dashed paths are predicted.

Mesh networks have topologies in which all network nodes are connected to each other. Performing full mesh active measurement in a large-scale network, as shown in Figure 1.1 (a), often incurs prohibitively high measurement overheads. Hence, the researchers have been trying to design scalable prediction schemes to reduce the measurement overhead. Figure 1.1 (b) illustrates one of the many approaches in which some paths are measured and the rest are predicted via prediction algorithms and thus, introducing less probing traffics.

Considering the potential overhead and probing traffics caused by intrusive measurement methods, the research reported here focuses on non-intrusive network performance prediction on large-scale networks.

1.2 Problem Definition

End-to-end network performance prediction is defined as follows: In a network with *n* nodes, a link is defined as the routing path between two end nodes. There will be $O(n^2)$ paths in a mesh network for the measurement among the *n* network nodes. In order to reduce the measurement overhead, a subset of paths is measured and the rest are predicted. The prediction problem is solved by machine learning method in an iterative manner through the minimizing of the loss function and convergence to the local minimum. In other words, the algorithm iterates over time to reduce the discrepancy between measured and predicted values.

Implication of machine learning methods constructs a simple model based on the input data which is able to make intelligent decisions. The concerns of using machine learning techniques are a) the learning model constructed based on the training data, b) the number of measured data which is representing the measurement overhead. The machine learning technique, namely Matrix Completion, used to produce such model is widely used in data mining and pattern recognition. The prediction accuracy is proportional to the measurement overhead. This thesis aims to achieve the best accuracy with the minimum measurement overhead, by measuring to a subset and predicting the rest of the network.

1.3 Problem Statement and Contributions

This thesis strives to address the following problem: Current empirical methods of network bandwidth estimation and measurement incur high error rate over lossy network and impose high network overhead with flooded measurement packets. We propose to address this issue via matrix factorization approach which yields higher accuracy with improved convergence relative to existing approaches. In order to further enhance the proposed algorithm, available bandwidth and latency are formed as pairs for network link quality prediction. The proposed algorithm was tested and evaluated on publicly available dataset of available bandwidth and latency information.

1.4 Research Objectives

This research is aimed to:

i. design and develop a predictive network resource estimation mechanism which enables non-intrusive network available bandwidth and latency prediction over heterogeneous network, and
ii. develop an optimized bandwidth prediction algorithm.

1.5 Thesis Outline

The thesis is organized as follows:

Chapter 2 discusses the progress and limitations of related work on network performance prediction and identifies the research gap.

Chapter 3 introduces the formulation of the proposed algorithm and describes the proposed algorithms and implementations for available bandwidth and latency prediction, binary and multiclass classification.

Chapter 4 presents the simulation results and discussion of the proposed algorithms.

Chapter 5 concludes this thesis with some proposed future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, a survey report on research work carried out in the past on latency and available bandwidth prediction methods is presented covering their features and limitations. Later, a comparative study on network resource estimation is presented in Section 2.6. Furthermore, a survey on application of network distance prediction is presented in the end of the chapter.

2.2 Euclidean Embedding for Latency Prediction

There are several accurate solutions that have been proposed such as Global Network Positioning (GNP) (Eugene Ng & Zhang, 2002) and Vivaldi (Dabek, et al., 2004) for latency estimation. GNP and Vivaldi methods are based on network embedding system which models the Internet as a geometric space and maps the nodes as coordinates. Vivaldi is the extended version of GNP in decentralized manner. Both of these approaches are purely meant for latency prediction. Furthermore, the Euclidean-based embedding applied by these approaches is not suitable for asymmetric network distance prediction.

2.2.1 Global Network Positioning (GNP)

The main concept of GNP (Eugene Ng & Zhang, 2002) is to model the Internet as a geometric space and characterizes the position of the network nodes as a coordinate. Latency, also known as network distance, is predicted as the distance between two nodes embedded on the network coordinate system. Landmark-based architecture is applied in GNP whereby a set of distributed nodes are selected as Landmark and serve as a reference for the new joining host.

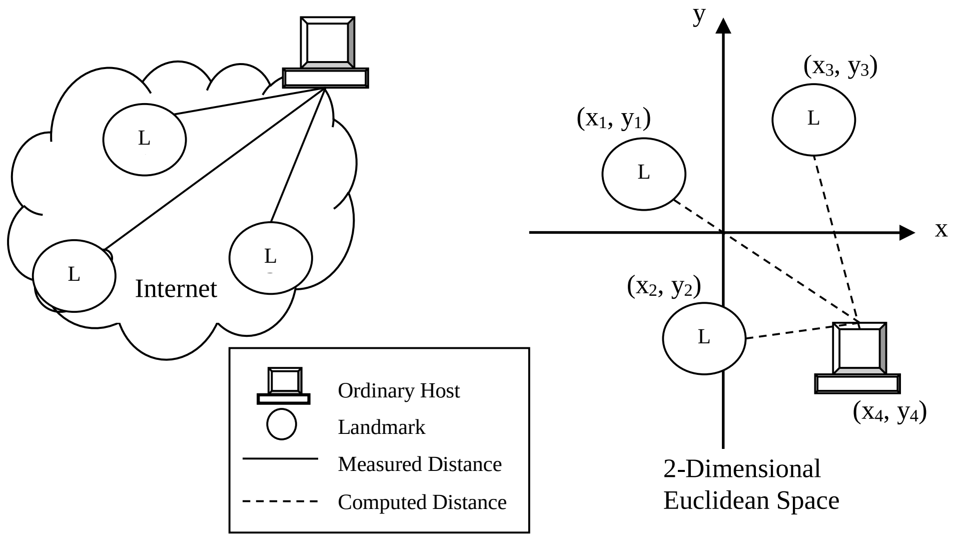


Figure 2.1 Ordinary host operations for GNP.

The architecture of GNP is divided into landmark operation and ordinary host operation. The distance between landmarks is first computed in the chosen coordinate system. These coordinates are then disseminated to the new joining host. As seen from Figure 2.1, each ordinary host will derives its own coordinates in relative to the coordinate of the landmark hosts. Euclidean based embedding suffer bad prediction accuracy when the network topology changes. The biggest limitation encountered in GNP is the selection of landmark nodes which led to limited scalability and lack of flexibility.

2.2.2 Vivaldi

Vivaldi (Dabek, et al., 2004) is an extended version of GNP in a decentralized manner whereby the landmark nodes are eliminated. Latency is predicted based on the distance between hosts on the coordinate system. The new joining host computes its coordinate after collecting latency information from a small set of hosts. This procedure is performed on every node and it keeps the node’s coordinates updated over time. This provides the Vivaldi method with the ability to tolerate high number of erroneous nodes and the metric diversity.

The shortcoming of the Vivaldi method is that slow convergence is experienced if a large number of nodes join the network at the same time. Furthermore, the method is not applicable to available bandwidth prediction. It was mainly designed for latency estimation.

Two important properties are applicable to the distances on the metric embedding spaces:

- Symmetry: $d(A, B) = d(B, A)$;
- Triangle Inequality: $d(A, B) + d(B, C) \geq d(A, C)$

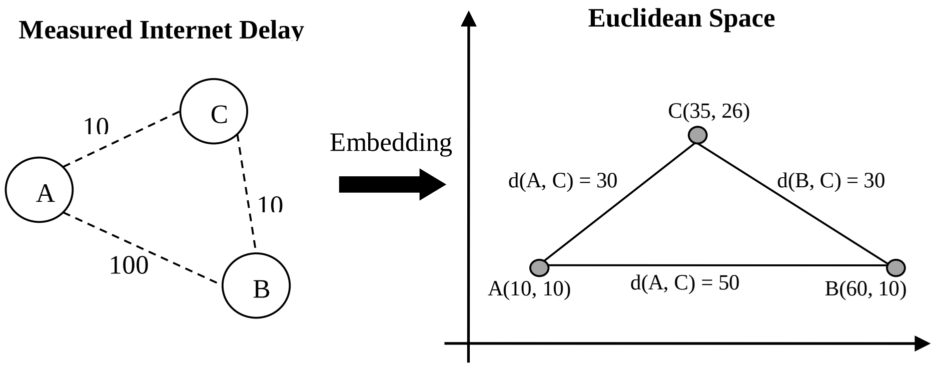


Figure 2.2 Three network nodes are embedded on the Euclidean space. The triangle of Δ*ABC* shows the constraint of the triangle inequality as d(*A*, *C*) < d(*A*, *C*) + d(*B*, *C*).

The network distances are not necessarily symmetric especially for one-way delays (Pathak, et al., 2008). Euclidean embedding is not applicable on available bandwidth prediction due to its asymmetrical behaviour. The violation of triangle inequality presents a big issue to the Euclidean based embedding. With the presence of triangle inequality, the edges on the embedding space are forced to shrink or to stretch accordingly and hence causing the prediction accuracy to decrease. There have been various approaches such as (Dong, et al., 2010) and (Kaafar, et al., 2009) been proposed to address the issue.

2.3 Available Bandwidth Prediction Approaches

This section mainly presents available bandwidth prediction approaches via route observation.

2.3.1 BRoute

BRoute (Hu & Steenkiste, 2005) is a route sharing based model proposed for available bandwidth estimation. The route sharing model is leveraged by the fact that most Internet bottlenecks are on path edges, which are shared by many different paths. The authors use autonomous systems (AS) level source and sink tree information to infer network path edges. A common-AS is used to identify the end-segments between source and sink tree. Border Gateway Protocol (BGP) is used to obtain the information between AS on the Internet. The BRoute server will return the smaller available bandwidth value between source to common-AS and destination to common-AS as the predicted value. The limitation of this model is that measurement overhead is linear with the number of end nodes in the system.

2.3.2 PathGuru

PathGuru (Xing, et al., 2009) is a landmark-based system for available bandwidth estimation. In PathGuru, each node obtains an outgoing and an incoming bandwidth vector using measurement data to and from landmarks. This method relies on the observation that, in certain circumstances, Internet's available bandwidth forms an ultra-metric space (Buneman, 1974). An ultra-metric space satisfies the Three-Points Condition (3PC), whereby for every three points $x, y, z \epsilon X$, the distances between the points, $D(x, y) \leq D(x, z) \leq D(y, z)$ and the fact $D(x, z) = D(y, z)$ holds. So, in this condition, the distance between node x and y, $D(x, y)$, is predicted as smaller or equals to the maximum of $D(x, z)$ and $D(z, y)$. For each prediction, the landmarks are selected based on ultra-metric space formation that holds highest approximation to the nodes to be predicted, and then the measurement results are applied to these landmarks to predict the available bandwidth between two ordinary nodes. The accuracy of this method is limited by the landmark node selection whereby the node may leave the network.

2.3.3 Last-mile Model

Last-mile model (Beaumont, et al., 2011) generalized the use of a "last-mile" (end-host) approximation, which reflects the overall performance of the complete end-to-end path. In Last-mile model, each node x is represented by two different bandwidth capacities, one for upload, β_x^{out}, and one for download, β_x^{in}. The end-to-end performance is assumed to be limited by the upload and download capacities. Hence, the predicted bandwidth P between two nodes x and y is given by $min(\beta_x^{out}, \beta_y^{in})$.

The initial value for Last-mile algorithm is very crucial as one bogus measurement will intensify more prediction error. Hence the authors introduced α value to remove a few outliers before prediction in order to deal with the missing and erroneous measurements. The initial outgoing and incoming capacity of x is computed as $(1 - \alpha)$-th percentile of its measurements to the neighbour nodes. Then, iterative procedure is applied to improve the initial calculation. The accuracy of Last-mile model decreases when there is limited number of available measurement. Furthermore, the “last-mile” assumption is not always satisfied in practice, since some nodes might share the same bottleneck link.

2.4 Unified Approach via Prediction Tree

A unified approach for latency and available bandwidth prediction called Sequoia (Ramasubramanian, et al., 2009), is a light-weight system which embeds nodes on tree structures via virtual nodes. The paths measured are assumed to be symmetric and the triangle inequality holds as well. Sequoia constructs *prediction trees* based on Gromov product through virtual nodes embedding, as illustrated in Figure 2.3. The requirement to construct prediction tree is that at least two nodes must already exist and measured in the tree.

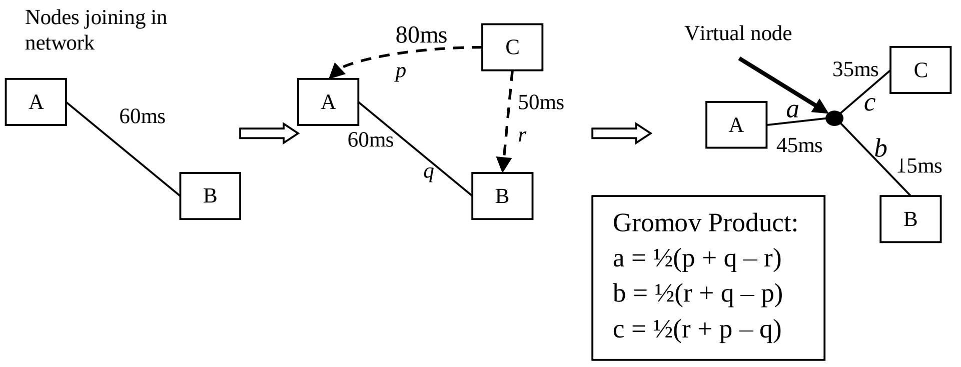


Figure 2.3 The tree embedding structure based on Gromov Product in simplified form.

For the example in Figure 2.3, latency between node *A* and node *B* is measured. When node *C* wants to join the network, it measures to node *A* and node *B*. By applying Gromov Product, the three nodes are embedded together via virtual node as filled in dark circle in Figure 2.3. The node selection is very crucial in *prediction tree* approach as it is linearly proportional to the prediction accuracy. The prediction accuracy drops when the number of nodes joining increases. Furthermore, Sequoia is unable to provide asymmetric prediction and also sensitive to the violations of triangle inequality.

A recent attempt to solve the symmetrical constraint on prediction tree construction was reported in (Schober, et al., 2013). Due to the highly asymmetric features in available bandwidth measurement, the authors presented a direction-aware embedding, by separating upstream from downstream properties of hosts. The idea is to embed nodes for each direction separately and embedding it twice on the prediction tree, one for its upstream and the other for downstream properties. This approach is still susceptible to the violation of triangle inequality.

2.5 Matrix Factorization

The occurrence of triangle inequality violation has led the research community to focus on a rather tolerant approach, called matrix factorization. The pairwise network metrics are treated as the matrix element which enables prediction in large-scale network. The assumption made in this approach is that the network performance metrics in such network have to be correlated. Models based on matrix factorization are able to overcome the constraints that exist in the Euclidean embedding system. Furthermore, it is implementable with other network performance metrics as long as they are correlated.

The model in matrix factorization is formulated as, a matrix, *D*, of size $n \times n$ is approximated by a low-rank matrix, $\widehat{D}$ of rank *r*, where $r << n$, is factorized into the product of two smaller matrices, *X* and *Y*, of size $n \times r$, as follows:

$$D \approx \widehat{D} = XY^T \tag{1.1}$$

The model is further illustrated in Figure 2.4, with some elements in matrix *D* are measured (*blue boxes*), and the rest are unmeasured (*grey boxes*). The predicted values (*yellow boxes*) are approximated by the two smaller matrices, *X* and *Y*. The diagonal elements of the matrix are left empty since it is not of concern in the analysis here.

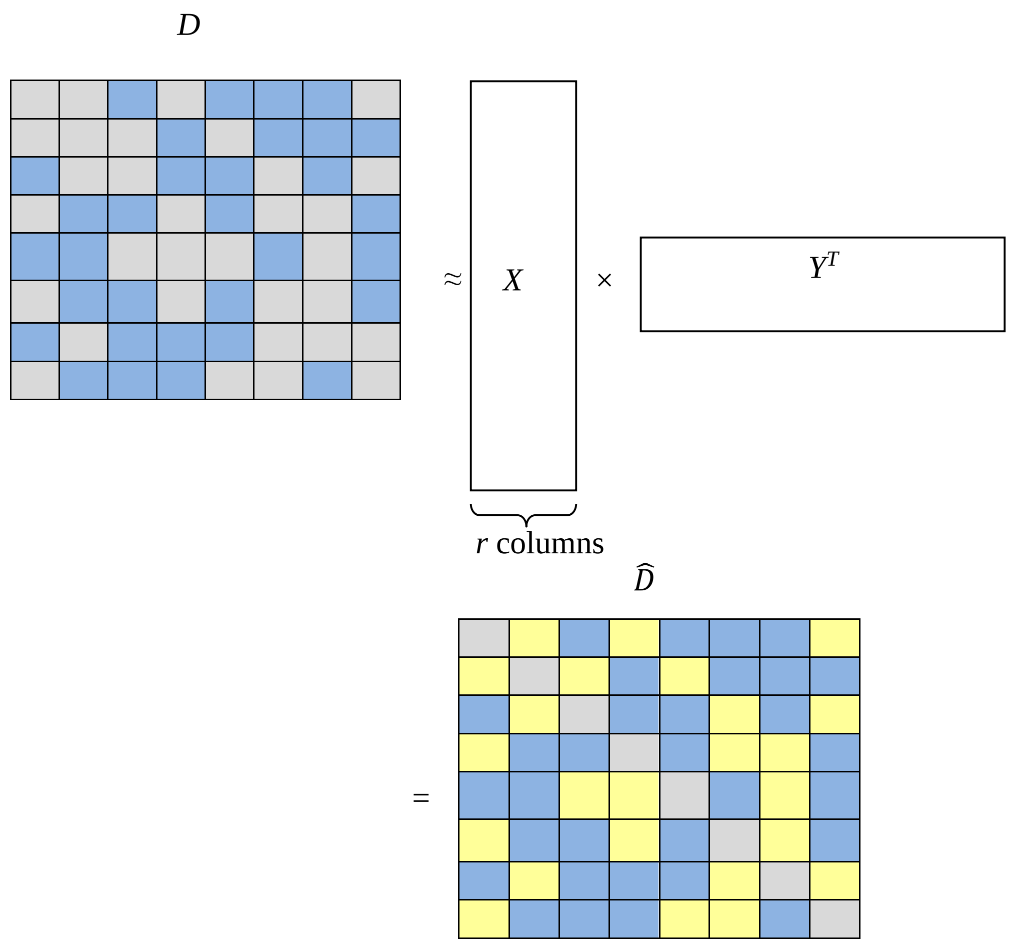


Figure 2.4 The model of matrix factorization.

After the matrix D is factorized into two smaller matrices, each node i is associated with two vectors, x_i and y_i, corresponding to the i^{th} row of X and of Y. The pairwise metric value, from node i to node j, is then predicted as the dot product of x_i and y_j. The details of the matrix factorization operations will be discussed in Chapter 3. There are three approaches that adopt matrix factorization: Internet Distance Estimation Service (IDES) (Yun, et al., 2006), Decentralized Matrix Factorization (DMF) (Liao, et al., 2010), and Decentralized Matrix Factorization via Stochastic Gradient Descent (DMFSGD) (Liao, et al., 2012).

2.5.1 Internet Distance Estimation Service (IDES)

IDES (Yun, et al., 2006) is the first system formulated network distance prediction as a matrix completion problem. The network distances are modelled as the dot product of two smaller matrices using two algorithms, namely Singular Value Decomposition (SVD) (Klema & Laub, 1980) and Non-negative Matrix Factorization (NMF) (Lee & Seung, 2001). IDES applies landmark-based system similar to GNP which was discussed in section 2.2.1. The difference is that in IDES a new joining host does not have to measure with respect to all the landmark nodes. The example illustrated in Figure 2.5 shows how prediction is performed on a newly joined node even without measuring it with respect to all landmark nodes.

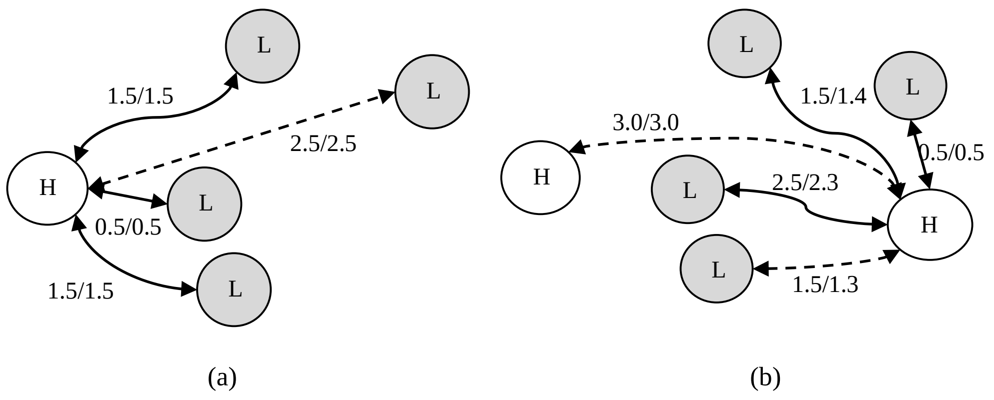


Figure 2.5 In (a) new joining host H_1 is measured with respect to L_1, L_2 and L_4 (denoted in solid lines) and predicted for its value with respect to L_3 (denoted in dashed line). In (b), when another ordinary host H_2 joining, it will be measured with respect to a few available nodes and predicted for its value with respect to the other nodes via matrix factorization.

In Figure 2.5, the values denoted on the edge between nodes are the outgoing and incoming vectors for the respective host. SVD is used to factor the landmark distance to obtain the two smaller matrices of which the details will be discussed in Chapter 3. The use of landmark nodes is the shortcoming of IDES whereby the landmark nodes selection will affect the overall accuracy.

2.5.2 Decentralized Matrix Factorization (DMF)

The decentralized version of IDES was proposed by Liao, Geurts and Leduc (2010), and the matrix factorization problem is solved using Alternating Least Square (ALS) method. The matrix with missing elements is factorized and estimated iteratively by having each node retrieving a small number of distance measurements. DMF does not require any special nodes for landmark usage or central nodes to collect and store data. The authors applied non-negative matrix factorization (NMF) to make sure the predicted values are all positive. NMF is enhanced with iterative optimization methods such as gradient descent to improve speed of convergence. The shortcoming of this algorithm is that no guarantee on the global minimum as it depends on the initial node selection for convergence.

2.5.3 Decentralized Matrix Factorization by Stochastic Gradient Descent (DMFSGD)

DMFSGD (Liao, et al., 2012) is decentralized in the sense that no extra matrix construction is needed and only the exchange of vector information between nodes is involved. One of the limitations faced in DMFSGD is that the speed of convergence is decreased with the increase in the amount of neighbour information retrieved for prediction. The prediction accuracy is proportional to the number of probed neighbour nodes.

This algorithm was further extended to qualitative prediction by replacing the quantitative value with 1 for *good* and -1 for *poor* (Liao, et al., 2011). The network performance is classified through threshold, τ which is determined based on the application requirements. The authors suggested that threshold can be directly applied on round-trip-time where the measurements are cheap. For available bandwidth, direct classification measurement data can be obtained via *pathload* (Jain & Dovrolis, 2002) and *pathChirp* (Ribeiro, et al., 2003) with little modification. For example, direct classification of available bandwidth is done via *pathload* by sending UDP trains at a constant rate of τ, and classifies the path as *good* if no congestion is observed and *poor* otherwise. This might be inaccurate when the metrics on the path is close to τ.

2.6 Comparative Study

As introduced earlier in this chapter, the methods proposed for latency and available bandwidth prediction are mainly consist of four categories, i.e., Euclidean embedding, route observation, prediction tree and matrix factorization. This section summarizes the most outstanding algorithm from each of the categories. Table 2.1 summarises the features, pros and cons for comparison.

Table 2.1 Comparison between various prediction approaches.

	Vivaldi	**Last-mile**	**Sequoia**	**DMFSGD**
Predict	Latency	Bandwidth	Latency & Bandwidth	Latency & Bandwidth
Main feature	Update node's coordinate at each timestamp	Characterize each node with outgoing and incoming bandwidth	Construct prediction tree	Factorize distance matrix
Require landmark	No	No	No	NA
Weakness	Slow convergence when large number of nodes joining	Accuracy do not increase with the number of neighbors	Inability to provide asymmetric prediction	Each node is characterized with a larger number of parameters (k, *l*)
Asymmetric prediction?	No	Yes	No	Yes

From the comparative study on network resource estimation, matrix factorization approach is selected as the main algorithm. The details of the algorithms which are proposed based on matrix factorization have been studied and presented in Section 2.5. The comparison table in Table 2.2 shows the differences between the proposed algorithm and the existing approaches.

Table 2.2 Network resource estimation approaches via matrix factorization.

	IDES	**DMF**	**DMFSGD**	**Proposed Algorithm**
Matrix factorization algorithm	SVD and NMF	Decentralized version of IDES	Low-rank matrix approximation	SVD and SGD
Features	Proceed with missing entries by eliminating the rows and columns in distance matrix that contain them.	Random initialization and updated continuously with respect to random selected neighbours.	Random update via SGD. Replace quantitative prediction with (-1, 1) for *good* or *poor* network path prediction.	Collect historical data and initialize via SVD. Further enhanced with link quality prediction
Qualitative Prediction	NA	NA	Binary classification	Binary and multiclass classification
Landmark	Required	Not Required		

The proposed algorithm is an optimization to the existing approaches with enhanced features which will be discussed in the following chapter.

2.7 Application of Network Performance Prediction

Network distance prediction poses a significant role in providing end-to-end network performance information. This information is very useful to Internet applications, such as overlay multicast and server selection. For example, in a large scale network, the predicted distances are used to guide the construction of multicast tree in order to reduce the measurement overhead. The following table shows the application of network performance prediction.

Table 2.3 Application of Network Performance Prediction

Application	**Description**	**How application facilitates the prediction algorithm?**
Overlay Network	Network that creates virtual topologies based on node-content attributes.	Next hop selection
Server Selection	Clients have to select servers without querying of server location or network topology.	Select the server based on lowest latency or highest available bandwidth

Overlay network creates a structured virtual topology on top of the physical topology (Doval, D. & O'Mahony, D., 2003) which offers automatic load balancing and self-organization. In the standard graph-traversal problem, each hop brings the query closer to the target node. Hence, network performance prediction can be used to define next hop selection. As a result, the network load and response time can be reduced.

The server selection problem often arises in content downloading, multimedia streaming and file transfer. Round-trip latency is important to minimize the response time for clients during content downloading while higher available bandwidth implies faster data transfer time. These network resources can be predicted via network performance prediction algorithms which in turn able to reduce measurement overhead in large scale network.

2.8 Conclusion

This chapter discussed the related works on latency and available bandwidth prediction with their features and limitations. The approaches presented focus on different aspects, with some assumptions made to meet certain criteria. The algorithm developed in this thesis does not rely on the network structural information and the qualitative-based approach is applied as well.

CHAPTER 3

END-TO-END NETWORK PERFORMANCE PREDICTION

3.1 Introduction

Predicting network bandwidth of large systems based on a few pairs of network nodes is essential to reduce measurement overhead. This chapter presents an algorithm which predicts network performance metrics, i.e. latency or available bandwidth based on matrix factorization and its further enhancement with network link quality prediction. In the first part of this chapter, the proposed SSGD algorithm is presented. Later in the second part of this chapter, the SSGD algorithm is further enhanced with network link quality prediction.

3.2 Network Performance Prediction Algorithm Overview

In this thesis, network performance metrics of the mesh network are formulated as matrix completion problem with measured and unmeasured entries. The formulation is made possible because of the strong correlations among network metrics. Furthermore, it is able to overcome the drawbacks of existing approaches based on Euclidean embedding.

A new algorithm, namely Singular value decomposition – Stochastic Gradient Descent (SSGD), is proposed to solve the network performance prediction via matrix completion. The algorithm is fully decentralized whereby each node only stores local information and exchange messages with each other, without the needs of landmarks or central servers. Different from the existing approaches; the conventional gradient descent is initialized with Singular Value Decomposition (SVD) to enhance better convergence. The proposed algorithm is illustrated in Figure 3.1 where SSGD algorithm is further enhanced with network link quality prediction.

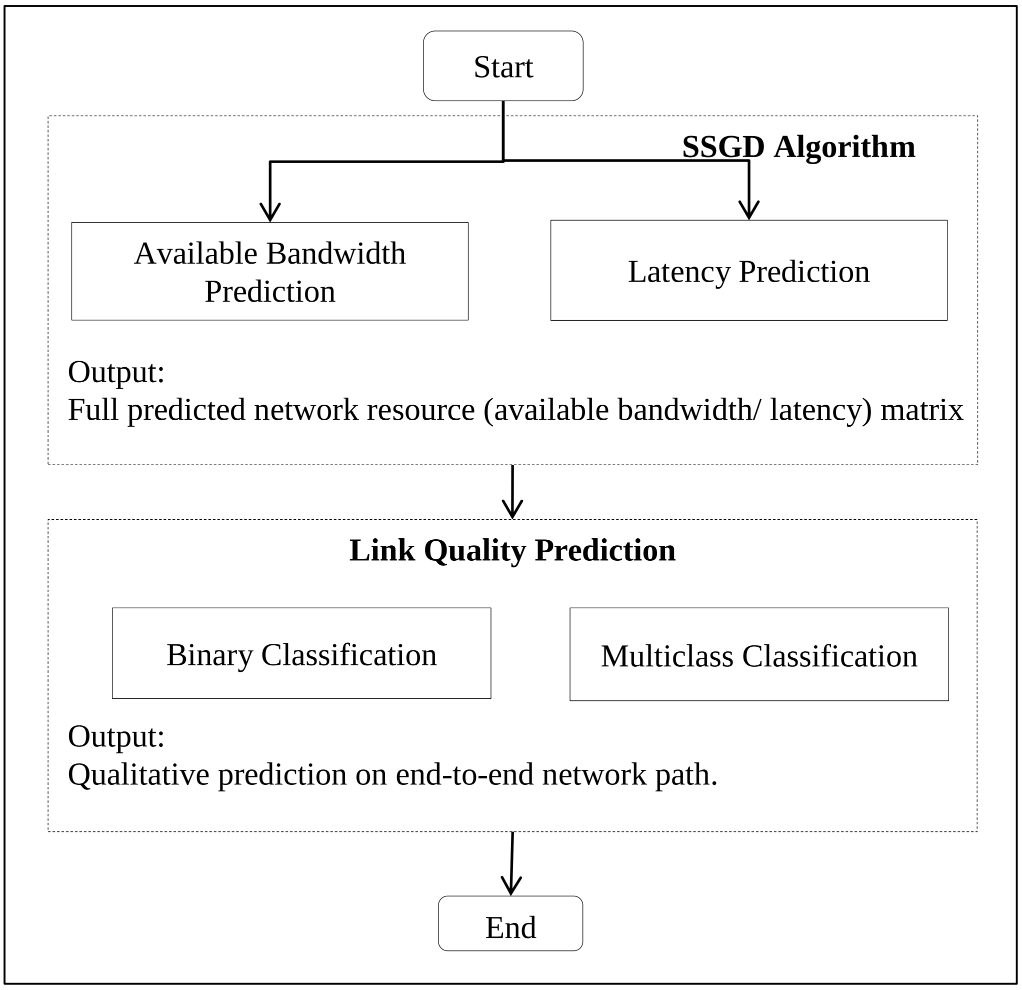
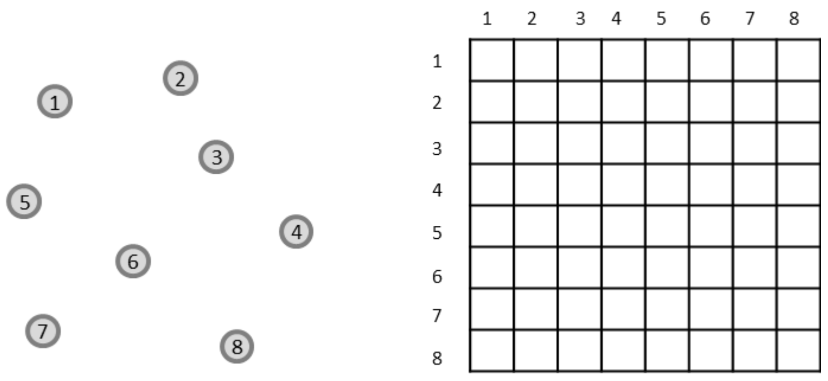

Start
SSGD Algorithm
Available Bandwidth Prediction
Latency Prediction
Output:
Full predicted network resource (available bandwidth/ latency) matrix
Link Quality Prediction
Binary Classification
Multiclass Classification
Output:
Qualitative prediction on end-to-end network path.
End

Figure 3.1 The illustration of the proposed algorithm.

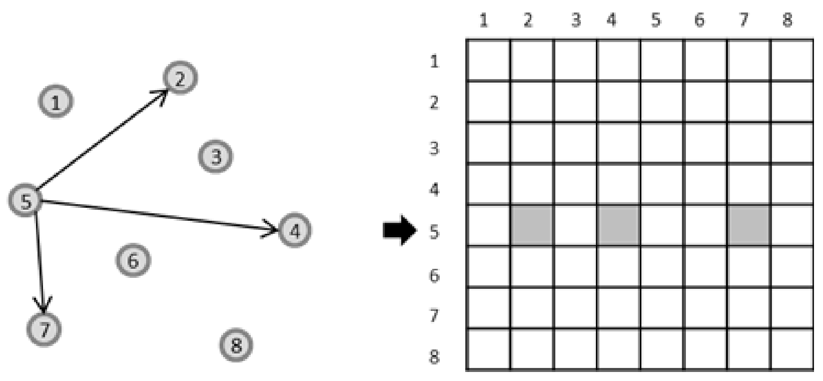
The metric pair, is extracted from SSGD to form the input for link quality classification. These enable the prediction to be done in both quantitative and qualitative way. Link quality prediction served as an enhancement to the quantitative prediction which can be easily extend to other network metrics such as packet loss, jitter, and throughput for classification.

3.3 SSGD Algorithm for Network Metrics Prediction

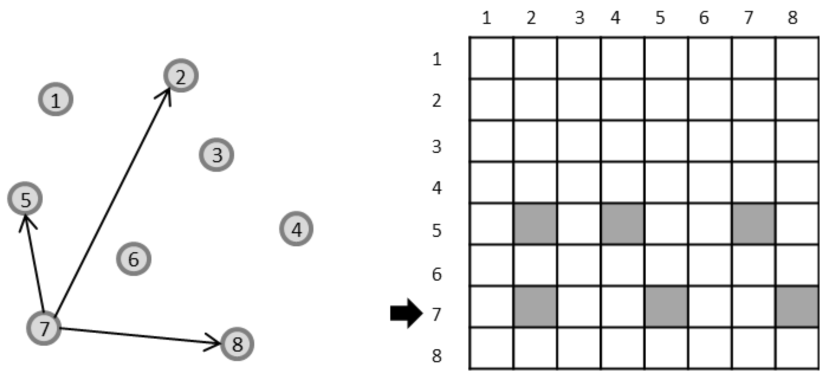
The proposed algorithm is divided into two phases: *initialization* and *stochastic update*. For a matrix with n nodes, the network will form $n \times n$ distance matrix with some distances between nodes measured and others unmeasured.



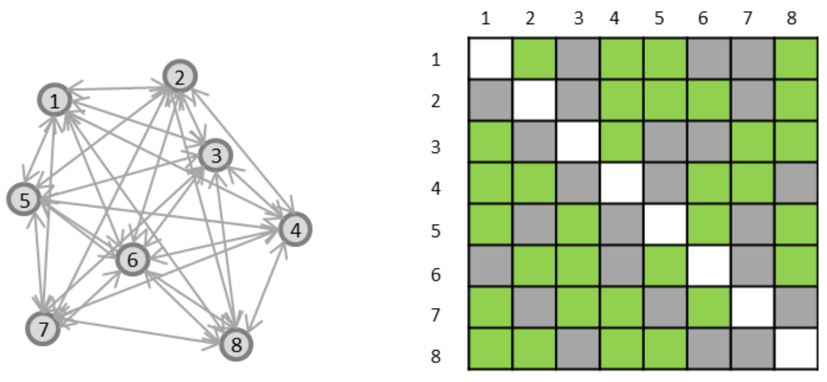
(a) Matrix of 8 network nodes.



(b) Node 5 measures network metrics to node 2, 4, and 7.



(c) Node 7 measures network metrics to node 2, 5, and 8.



(d) A matrix with measured and unmeasured entries is constructed.

Figure 3.2 The formulation of network performance prediction. (Note: The constructed matrix in (d) contains measured value in grey and green entries are missing elements to be predicted. The diagonal entries are left empty as the performance of a node to itself is not a concern.)

The proposed SSGD algorithm can be adopted in a decentralized network architecture with no central server. At the initial stage of the SSGD algorithm, SVD initialization is applied before the stochastic update. SVD is a matrix factorization method which serves as a method for data reduction whereby the best approximation can be found by using the number of data points lower than the number of network nodes (Baker, 2013). The following pseudocode shows the flow of the proposed SSGD algorithm.

```
Algorithm 1 The proposed SSGD algorithm flow
Input: D: measurement matrix
    Perform row-column interpolation
    $k$: number of neighbours for each node
    $\lambda$: regularization coefficient
    $r$: number of dimensions
    $i$: number of iterations
    $\eta$: learning rate
Output: $U_x$, $V_y$, and $J(\theta)$
    perform Singular Value Decomposition initialization to obtain $U_x$, $V_j$
    for $i$ number of iterations
        select a random node $x$ with $k$ number of neighbours
        minimize the cost function $\mathrm{J}(\theta)$: $\min \mathrm{J}(\theta) = (D_{i,j} - U_x V_y)$
    end for
```

SSGD algorithm is further elaborated in the following sections.

3.3.1 SSGD Algorithm: *Initialization*

The initial matrix D is constructed by filling it in with measurement carried out on a subset of network nodes and some unmeasured entries. Prior to the prediction stage, pre-processing which is the initialization through SVD is performed. SVD is based on a linear algebra theorem which explains that a rectangular matrix D can generated from the product of three matrices, an orthogonal matrix U, a diagonal matrix S, and the transpose of an orthogonal matrix V. The theorem is presented as:

$$D_{mn} = U_{mn} S_{nm} V_{nn}^{T} \tag{3.1}$$

where $U^T U = I$, $V^T V = I$; the columns of U are orthonormal eigenvectors of DD^T, the columns of V are orthonormal eigenvectors of $D^T D$, and S is a diagonal matrix containing the square roots of eigenvalues from U or V in descending order.

Since SVD cannot handle missing elements (Brand, 2002), row and column interpolation is performed to fill the missing entries as follows:

$$\widehat{D}_{i,j} = \min(\widehat{D}_{i^{th}\,row}, \widehat{D}_{j^{th}\,col}) \tag{3.2}$$

where $\widehat{D}_{i,j}$ is the missing value from node i to node j, $\widehat{D}_{i^{th}\,row}$ is the mean of ith row and $\widehat{D}_{j^{th}\,col}$ is the mean of jth column. In order to obtain a low-rank approximation, the number of singular values extracted represents the rank of

D. Only r numbers of rank in S are kept and the rest are replaced by zero. These three smaller matrices are formed into two smaller initialization matrices X and Y for stochastic updates. Let S_r be the new S, $X = US_r^{1/2}$ and $Y^T = S_r^{1/2}V^T$, the predicted distance matrix $\widehat{D} = XY^T$ is then the optimal low-rank approximation to *D*.

3.3.2 SSGD Algorithm: *Stochastic Update*

The initial predicted distance matrix is optimized based on stochastic gradient descent (SGD), a method which is particularly suitable to solve the matrix factorization problem (Koren, et al., 2009). Whenever there is a random measured value, d_{ij}, node i and node j update their coordinates to minimize the loss in the expression below

$$l\left(d_{ij}, x_i y_j^T\right) + \lambda x_i x_i^T + \lambda y_j y_j^T \tag{3.3}$$

where λ is the regularization coefficient (Bottou, 1998) and that $x_i y_j^T = \hat{d}_{ij}$ approximates d_{ij} better. The loss function *l* is applied and defined as

$$l\left(d, \hat{d}\right) = (d - \hat{d})^2 \tag{3.4}$$

The updates are along the negative gradient as expression 3.3. Figure 3.2 illustrates the stochastic gradient descent optimization for matrix factorization.

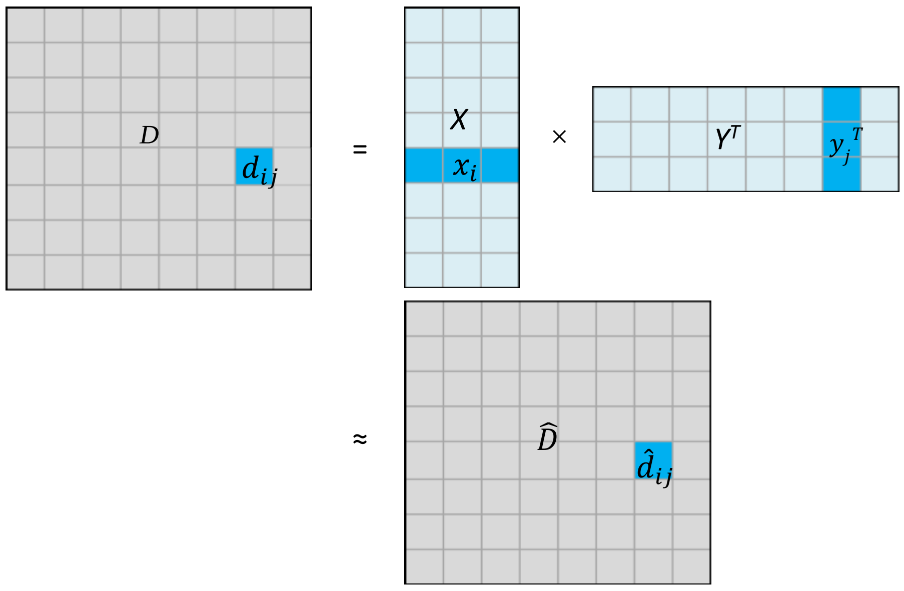


Figure 3.3 Stochastic gradient descent optimization for matrix factorization. The predicted value $\hat{d}_{ij}$ can be updated whenever the measured value d_{ij} is available. The ith row of X and jth row of Y can be updated so that $x_i y_j^T = \hat{d}_{ij} \approx d_{ij}$.

By using SGD to solve expression 3.3, the algorithm loops through all the samples d_{ij}, for $i, j < n$ and select random sample to update the respective x_i and y_j so that $d_{ij} \approx x_i y_j^T$. The algorithm runs through t number of iterations to minimize the discrepancies between measured and predicted values, as follows:

$$l_{ij} = (d_{ij} - x_i y_j^T)^2 + \lambda x_i x_i^T + \lambda y_i y_i^T \qquad (3.5)$$

The x_i and y_j are updated based on a learning rate η, or step size, in the negative gradients (Bottou, 1998), giving

$$x_i = (1 - \eta\lambda)x_i + \eta(d_{ij} - x_i y_j^T)y_j \quad (3.6)$$

$$y_i = (1 - \eta\lambda)x_j + \eta(d_{ij} - x_i y_j^T)x_i \quad (3.7)$$

The algorithm convergence is observed at each iteration which minimizes the difference between d_{ij} and $\hat{d}_{ij}$. The convergence are said to be improved when the difference between d_{ij} and $\hat{d}_{ij}$ was reduced after an iteration. SGD involves only simple update rules with vector operations and therefore it is able to deal with large-scale dynamic measurements.

3.4 End-to-end Network Link Quality Prediction

The quantitative prediction via matrix completion introduced in the Section 3.3 of this chapter is further enhanced with link quality prediction, whereby available bandwidth and latency are both taken into consideration to predict the quality of the link between two network nodes. This can be easily extended to other network metrics such as packet loss, jitter or throughput. The quality of the link between frequent communicating nodes is classified based on supervised learning classification. Network link quality is classified into binary (*good* or *poor*) and multiclass (*very good*, *good*, *moderate*, *poor*) to adapt to the dynamic network condition by adjusting quality of the content delivery. Available bandwidth and latency are predicted via the proposed SSGD algorithm in the previous section. The metrics pairs are extracted and serve as training data for link quality prediction via Logistic Regression (Menard, 2010) and Support Vector Machine (Cortes & Vapnik, 1995).

Machine learning algorithms can be categorized into unsupervised learning and supervised learning. In unsupervised learning, the algorithm finds the structure of the training samples with unknown labels. The structure of the training samples is often referred to as a group formed based on their similarity (Komarek, 2004). In this thesis, the training samples are available bandwidth and latency, which are obtained from SSGD algorithm. Therefore, the labels of the pairs (*good* or *poor*) are known and applied using supervised learning based algorithm. In supervised learning, the system acquires knowledge from previous training samples and adapts the system with a new model for prediction on new input data. The most widely used supervised learning algorithms are Support Vector Machines (Cortes & Vapnik, 1995), logistic regression (Menard, 2010), and k-nearest neighbor algorithm (Mack, 2010). Support Vector Machines (Cortes & Vapnik, 1995) and logistic regression (Menard, 2010) are implemented as these two are closely related and to identify which method is better fit for the learning pattern in this thesis.

3.4.1 Logistic Regression for Link Quality Prediction

Logistic Regression (LR) is frequently used to estimate qualitative response models in which the dependent variable is a dichotomy, such as email spam filtering (Chang, et al., 2008), fraudulent detection for online transactions (Chae, et al., 2007), and tumor malignancy classification (Timmerman, et al., 2005). This method is applied to predict the quality between two network nodes with respect to their metrics pair (available bandwidth and latency). LR classifies the samples based on the probability of the sample to be *positive* or *negative*. For binary classification via LR, the algorithm uses the training samples, which is the metric pair, to predict the probability of the network link to be 0 (*good*) or 1 (*poor*). The following pseudo codes show the overview of link quality prediction via logistic regression.

Algorithm 2 Logistic Regression for Link Quality Prediction

```
Input: Training Data, X: Latency and Available Bandwidth pairs (x, y) with dichotomous outcome
        C: regularization coefficient
        m: number of training samples
Output: Link quality prediction: $p(y = 1| x; \theta)$
        for m samples
            minimize the cost function J(θ)
            for $j = 0, 1, \dots, m$
                perform gradient descent $\theta_j := \theta_j - \propto \frac{d}{d\theta_j} J(\theta)$
            end for
        end for
```

Let X be a dataset with dichotomous outcome, $y = \{0, 1\}$. For each training samples x_i in X, the outcome is either $y_i = 1$ or $y_i = 0$. The experiments outcome with $y_i = 1$ are said to have *good* link quality, while $y_i = 0$ for *poor* quality. The input dataset is learnable via a differentiable function instead of using a two line segment. The logistic regression model (Menard, 2010) is built based on the hypothesis as follows:

$$h_\theta(x) = p(y = 1|x;\theta) = g(\theta^T x) \tag{3.8}$$

which is also referred to as the probability that $y = 1$ given x, parameterized by θ, where g is the sigmoid function

$$g(z) = \frac{1}{1+e^{-z}} \tag{3.9}$$

The training pairs $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})\}$ with m samples where $x \in [x_1, x_2]^T$ is the set of input features are used to obtain the fitting parameter, θ to minimize the cost function *J(θ)* as follows:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right) - \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)] \tag{3.10}$$

In order to optimize the algorithm, gradient descent is applied to improve convergence as follows:

Optimization algorithm:

```
i.    Compute cost function J(θ).
ii.   To min_θ J(θ), perform gradient descent:
iii.  Repeat {
          $\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta)$      (for j = 0, 1, …, n)
      }
```

The gradient of the cost function is defined as

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m} h_\theta((x^{(i)}) - y^{(i)})x_j^{(i)} \tag{3.11}$$

The trained dataset will result in a model with decision boundary and trained parameter for new input samples. For the non-linearly separable dataset, features are mapped into higher dimension with higher polynomial terms of x_1 and x_2 to fit it, and regularization term θ is used to do parameter tuning. The polynomial is expanded up to the sixth power which is best fit in this case.

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \cdots) \tag{3.12}$$

The learning problem can be difficult if the dimension is too high. Features x_1 and x_2 correspond to the available bandwidth and latency on the link between two network nodes. The cost function with regularization term as follows builds a more impressive classifier and prevent over-fitting,

$$J(\theta) = -[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} log h_\theta(x^{(i)}) + (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))] \\ + \frac{\lambda}{m}\sum_{j=1}^{n} \theta_j^2 \tag{3.13}$$

The gradient of the cost function with regularization is defined as follows:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j \tag{3.14}$$

for $j = 1, 2, 3, \ldots, n$.

To enable multiclass classification, “one-against-all” logistic regression is applied by picking the class i that maximizes $max_i h_\theta^i(x)$. The hypothesis of multiclass logistic regression is different from that of binary logistic regression while the cost function is similar for both of the cases.

The hypothesis of *N*-class multiclass logistic regression is

$$h_\theta^i(x) = p(y = i|x; \theta) \tag{3.15}$$

where a logistic regression classifier $h_\theta^i(x)$ is trained for each class *i* to predict the probability that $y = i$. The prediction on each new sample is made by picking the class *i* that maximizes the hypothesis.

3.4.2 Support Vector Machine for Link Quality Prediction

Support Vector Machines (SVM) constructs decision boundaries based on different kernel function which separates a set of samples having different classes. The very basic example of SVM is the linear classifier, i.e. a classifier that separates a set of samples into their respective groups with a line. However, most classification problem is not linearly separable. The basic idea of SVM is to map the samples, using mathematical functions, known as kernels, into a higher dimensional space (Cortes & Vapnik, 1995). The following pseudo codes show the overview of link quality prediction via SVM.

Algorithm 3 Support Vector Machine for Link Quality Prediction

```
Input: Training Data, X: Latency and Available Bandwidth pairs (x, y) with dichotomous outcome
    $C$: regularization coefficient
    $\gamma$: margin of the support vectors
    $l$: number of training samples
    $\omega$: feature vector
    $\phi(x_i)$: kernel function
    $b$: bias
    Linear kernel function: $\boldsymbol{K(x_i, x_j) = x_i^T x_j}$ (for SVM with linear kernel)
    RBF kernel function: $\boldsymbol{K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \ \gamma > 0}$ (for SVM with RBF kernel)
Output: Link quality prediction: class label
    for $l$ samples
        minimize the decision function
        $\boldsymbol{sgn(\omega^T \phi(x) + b) = sgn(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x_j) + b)}$
    end for
```

SVM is able to give high accuracy with small training set size by mapping samples into higher dimensional feature spare. In this thesis, C-support vector classification (C-SVC) with linear and radial basis function (RBF) kernel is studied to classify the samples into binary and multiclass classification. The main parameter in C-SVC is *C*, the regularization parameter, which plays the role similar to $1/\lambda$, the regularization parameter in regularized logistic regression.

The following figure shows the concept of SVM with the decision boundary of $w.x + b > 0$ if $y = good$ and $w.x + b < 0$ if $y = poor$. Support vectors are the points touching the margin.

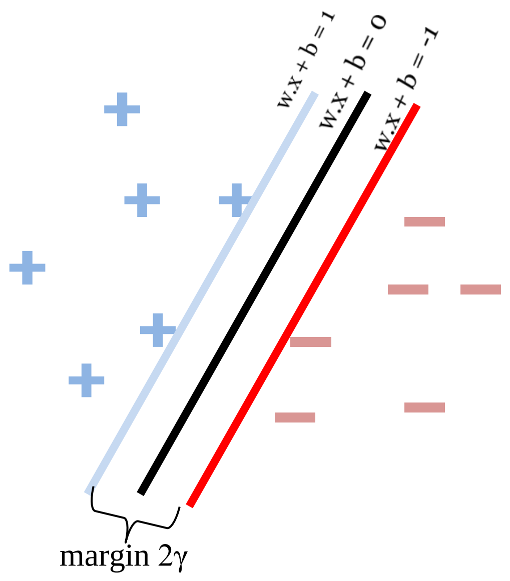


Figure 3.4 Illustration of SVM with support vectors.

The training features with l samples are interpreted as vectors in C-SVC, such that $x_i \epsilon R^n$, $i = 1,\ldots, l$, and the label vector $y \in R^l$ such that $y_i \in \{0,1\}$ to solve the following primal optimization problem as introduced in (Hsu, et al., 2010),

$$\min_{\omega,b,\xi} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l}\xi_i$$

$$\text{subject to} \quad y_i(\omega^T\phi(x_i) + b) \geq 1 - \xi_i, \tag{3.16}$$

$$\xi_i \geq 0, \;\; i = 1, \ldots, l$$

where the kernel function, $\phi(x_i)$, maps xi into a higher-dimensional space, $C > 0$ is the regularization parameter, b is a bias, ω is the feature vector, ξ_i is the "slack variables" and $\sum_{i=1}^{l} \xi_i$ is the sum of errors in addition to $\omega^T\omega$. For non-linearly separable data, "slack variables" are introduced in SVM as a trade off to allow outliers or noisy samples to be on the wrong side. The optimization problem can be tuned based on C whereby higher C minimizes mis-classifications while smaller C maximizes the margin. Due to the possible high dimensionality of vector variable ω, usually the dual problem is solved as follows (Hsu, et al., 2010):

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \\ \text{subject to} \quad & y^T \alpha = 0, \\ & 0 \leq \alpha_i \leq C, \quad i = 1, \dots, l \end{aligned} \tag{3.17}$$

where e = [1, …, l]^T is the vector of all ones, Q is an 1 by l positive semi-definite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is the kernel function. After solving the equation in 4.12 via the primal-dual relationship, the optimal ω satisfies

$$\omega = \sum_{i=1}^{l} y_i \alpha_i \phi(x_i) \tag{3.18}$$

and the decision function is

$$sgn(\omega^T \phi(x) + b) = sgn(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x_j) + b). \tag{3.19}$$

The training model consists of class label names, support vectors, and kernel parameters are used for prediction. The kernels function for linear and RBF kernel is shown below (Hsu, et al., 2010):

Linear kernel: $$K(x_i, x_j) = x_i^T x_j \tag{3.20}$$

RBF kernel: $$K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \ \gamma > 0 \tag{3.21}$$

The regularization parameter C and kernel parameters are then chosen according to the selected kernel. RBF kernel is able to classify non-linear attributes by mapping samples into higher dimensional space, while linear kernel is best fit for linear relation between class labels and attributes.

3.5 Conclusion

The overall idea of the proposed algorithms is presented in this chapter. The first part of the algorithm, SSGD is elaborated in section 3.3 while the enhancement is introduced in section 3.4 for link quality prediction. In the next chapter, simulation results for network performance prediction via SSGD algorithm and link quality prediction are presented.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the simulation of the proposed algorithm, which includes the test results, impact of parameter tuning and followed by performance comparison with other existing approaches. The enhancement to the proposed algorithm via qualitative approach for network link quality prediction is presented in Section 4.3.

4.2 Evaluating End-to-end Network Performance Prediction

Network performance metrics such as available bandwidth and latency are beneficial to overlay network construction and server selection. The prediction matrix consists of mesh bandwidth information with measured and predicted value.

There are mainly two stages in the proposed SSGD algorithm: pre-processing stage and prediction stage.

The pre-processing stage aims to locate good starting points for the nodes' convergence to local minima. In other words, the initial approximation is better guaranteed with the proposed algorithm to converge to a close enough solution during pre-processing stage. In essence, the outgoing, x_i and incoming, y_i vector of the node is assigned based on Singular Value Decomposition (SVD) instead of being randomly initialized which will result in the need for more iterations for convergence to local minimum. The flow of simulation setup is illustrated in Figure 4.1.

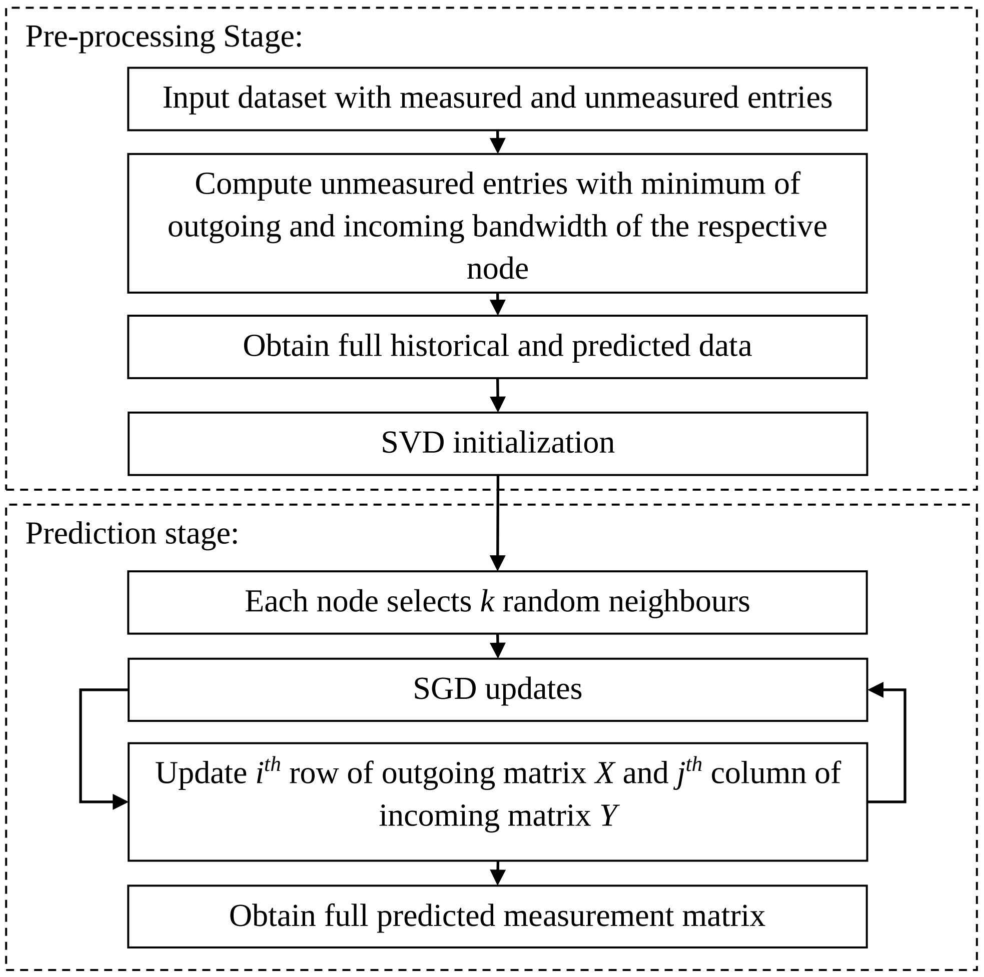

Pre-processing Stage:
Input dataset with measured and unmeasured entries
Compute unmeasured entries with minimum of outgoing and incoming bandwidth of the respective node
Obtain full historical and predicted data
SVD initialization
Prediction stage:
Each node selects k random neighbours
SGD updates
Update i^{th} row of outgoing matrix X and j^{th} column of incoming matrix Y
Obtain full predicted measurement matrix

Figure 4.1 Simulation setup for SSGD algorithm.

In the simulation, the unmeasured data are first predicted as the minimum of outgoing and incoming bandwidth of the corresponding network node. With this full historical bandwidth data, SVD initialization is performed to obtain the two smaller matrices *X* and *Y* as shown in Figure 4.2.

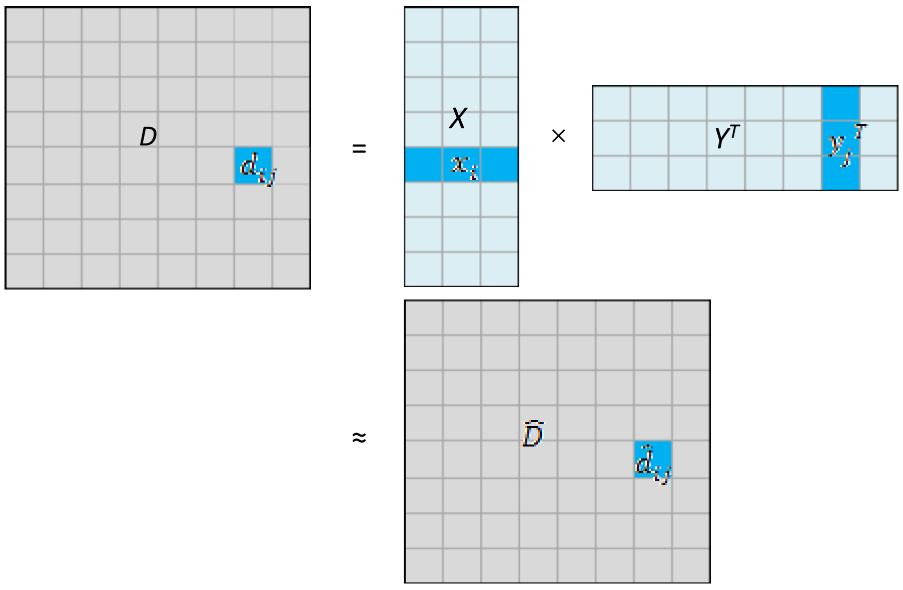


Figure 4.2 Illustration of matrix factorization operation.

In the prediction stage of the algorithm, stochastic update is performed on the measurement matrix through Stochastic Gradient Descent (SGD) as illustrated in Figure 4.2. A random node is selected at each epoch and the corresponding row and column of matrix *X* and *Y* along negative gradient are updated. The initial approximation of the function at the current point is updated along negative gradient to minimize the discrepancy between measured and predicted value. SGD is applied to estimate the predicted value to be closer to the measured value. Finally, a full measurement matrix is constructed with the predicted bandwidth information. The learning parameters involved in SGD are regularization parameter, λ, learning rate, η, approximation rank, *r*, and number of neighbors, *k*. The parameters obtained from the training set are cross validated and tested with different sets of data of the same size.

4.2.1 SSGD Algorithm Simulation Setup

The input datasets used for simulation were taken from the *Bedibe* project (Eyraud-Dubois & Uznanski, 2012) and they contain network available bandwidth measurements between 98 of the most responsive nodes, with the presence of some missing measurements. Properties of the datasets are presented in Table 4.1 with three sets of available bandwidth measurements captured on different dates, including a set of data for latency prediction. The datasets are asymmetric in which two-way measurements were performed.

Table 4.1 Properties of the datasets.

Measurement	**Dataset**	**Nodes**	**Asymmetry**
Available Bandwidth	HPS3 dataset (2011-11-21)	98	Yes
Available Bandwidth	HPS3 dataset (2011-11-22)	98	Yes
Available Bandwidth	HPS3 dataset (2011-11-24)	98	Yes
Latency	HPS3 dataset (2011-11-24)	98	Yes

The SSGD algorithm makes an assumption that available bandwidths and latencies among network nodes are highly correlated and that the network nodes are able to maintain a list of historical measurement.

4.2.2 Evaluation Metric

Quality of the prediction algorithm is measured by the cumulative distribution function (CDF) of relative error for all pairs of hosts. Relative error is defined as

$$RE_{x,y} = \frac{|M_{x,y} - P_{x,y}|}{M_{x,y}} \tag{4.1}$$

where $M_{x,y}$ is the measured value and $P_{x,y}$ is the predicted value. The CDF plots of relative errors indicate the higher the corresponding plot is, the better estimation performance.

The overall fitness of the embedding is measured by *stress* measure, which is used to illustrate the convergence of the algorithm, defined as follows

$$stress = \sqrt{\frac{\sum_{x,y=1}^{n} (M_{x,y} - P_{x,y})^2}{\sum_{x,y=1}^{n} M_{x,y}^2}} \tag{4.2}$$

Dataset with high variance is undesirable which will lead to low convergence of the prediction algorithm. High prediction errors are normally due to over-estimation or under-estimation of the bandwidth information. Impacts of parameter tuning are shown in the following section.

4.2.3 Impact of Parameter Tunings

The main parameters in the proposed SSGD algorithm are approximation rank, r, regularization coefficient, λ, learning rates, η, and number of neighbor, k. The *stress* of the proposed algorithm is observed for 100 iterations on 98 network nodes.

The regularization coefficient, λ reduces over-fitting in the algorithm and drift of the coordinates. The algorithm is simulated with $\lambda = \{0.1, 1, 10, 100\}$, $\eta = 3\text{x}10^{-5}$ and $r = \{10, 30, 50, 80\}$. The values of λ selected for testing are of multiples of 10. η is selected based on the result presented in Figure 4.5. The values of r used in the simulation are of increments of 20 or 30 whereby larger r involves lesser prediction with more active measurement which yields better accuracy. However, the aim of this research is to minimize active measurement via prediction algorithm. Therefore, r is selected to be as low as possible to achieve acceptable accuracy. It is shown that in Figure 4.3, the proposed algorithm achieved best convergence at $\lambda = 1$ and $r = 50$.

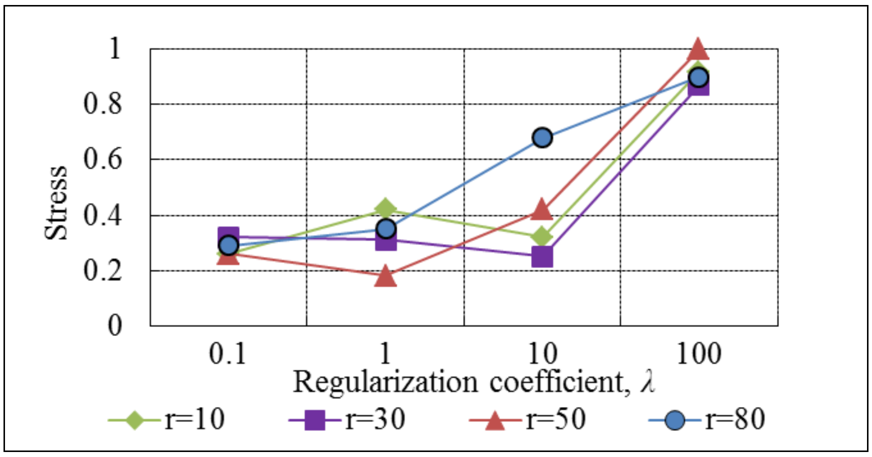


Figure 4.3 Impact of parameter for r versus λ, with $\eta = 3\text{x}10^{-5}$.

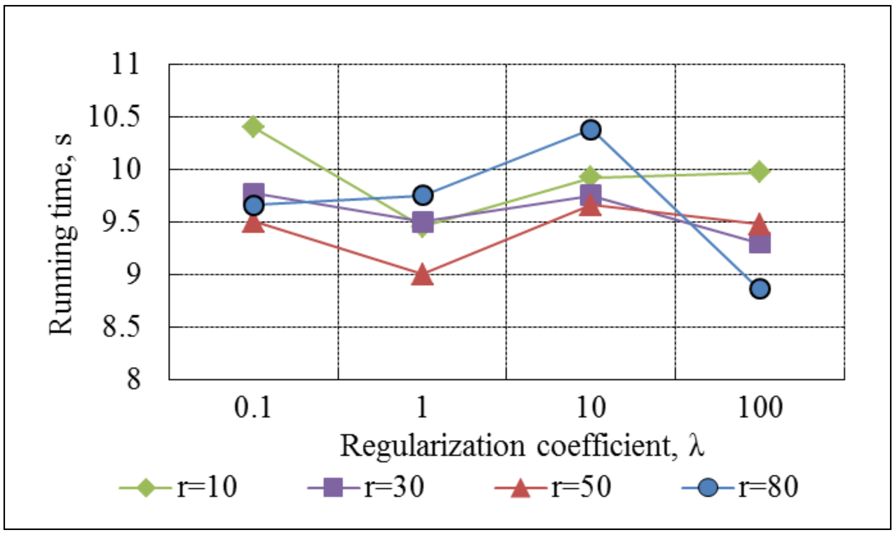


Figure 4.4 Impact of λ and r on algorithm computation time.

The rank constraint in SVD affects the performance of the algorithm by changing the number of singular values, which is represented by the approximation rank, r. It may affect the algorithm running time. As shown in Figure 4.4, with r = 50 and λ = 1 selected from Figure 4.3, the computation time is shorter when the parameter values are best suited for the respective dataset. Although r = 80 with λ = 100 achieved lower computation time, higher r yields more active probing which is unwanted in this research.

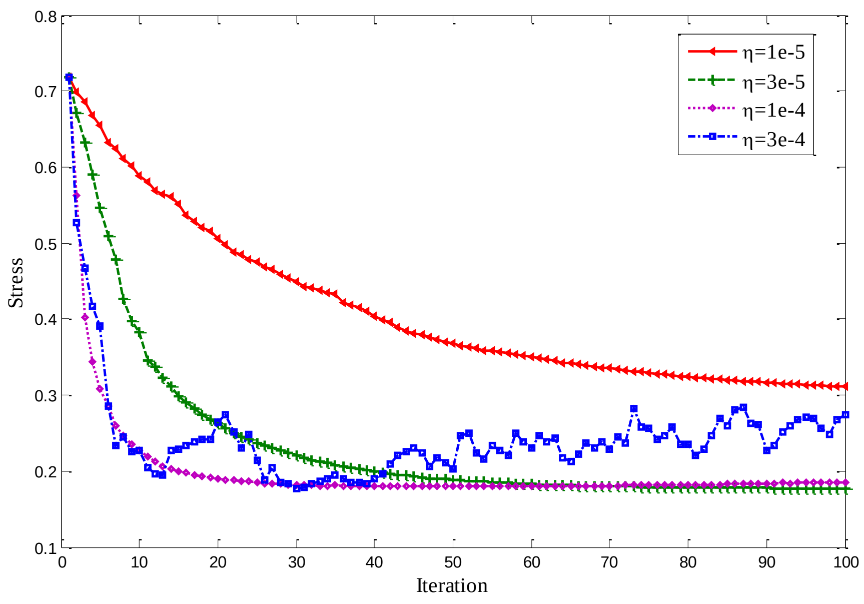


Figure 4.5 Impact of learning rate, η with $k = 32$, $r = 50$, $\lambda = 1$.

Learning rate or more often referred to as *step size* affects the speed of convergence of the algorithm. Larger values of η would also lead to divergence in the algorithm. The impact of η is tested with $\eta = \{3x10^{-4}, 1x10^{-4}, 3x10^{-5}, 1x10^{-5}\}$. It is important to observe the algorithm convergence and divergence. As shown in Figure 4.5, for $\eta = 1x10^{-4}$, the algorithm tend to diverge after 70th iteration while $\eta = 3x10^{-4}$ gave fluctuating results. Therefore, η is set to be $3x10^{-5}$ according to results in Figure 4.5 in order to prevent divergence.

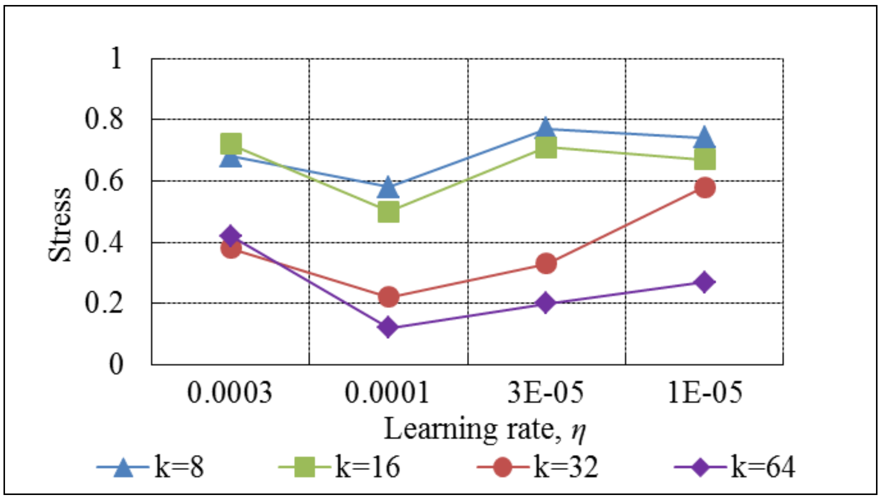


Figure 4.6 Impact of η versus number of neighbor, k, with λ = 1, r = 50.

Number of neighbor, k, represents the amount of active probing which is directly proportional to the algorithm accuracy. Larger values of k involve more active measurement with larger measurement overhead. The aim of this research is to minimize active measurement via prediction algorithm. The proposed algorithm is simulated with k = {8, 16, 32, 64} and η = {$3x10^{-4}$, $1x10^{-4}$, $3x10^{-5}$, $1x10^{-5}$} are selected for testing. The number of neighbor used for simulation is selected based on the sequence of 2^n with n = 3, 4, 5, and 6. From Figure 4.6, the *stress* of the algorithm is the highest at k = 64 and η = $1x10^{-4}$. However, η = $1x10^{-4}$ are shown to be diverged after 70th iteration as presented in Figure 4.5. Hence, η = $3x10^{-5}$ and k = 32 are selected based on the results in Figure 4.6. The active measurement is calculated as 32 over 98, which is 33% of the data are used to achieve *stress* as low as 0.3.

The tuning parameters involved in the proposed algorithm depend highly on the network environment from where the data is taken and the number of network nodes involved. Therefore, the values of the parameters should not be generalized and be applied on other datasets. Hence, the optimum training parameters obtained above should be cross-validated and tested with other compatible datasets.

4.2.4 Cross-validation of Training Set

In this section, the learning parameters obtained from the training set is cross-validated and tested with another compatible set of data. The size of the data and the network nodes selected is similar as in training set. Stress plots of the proposed SSGD algorithm and the conventional SGD algorithm with random initialization is also shown in this section. The parameters were set heuristically as $\lambda = 1$, $\eta = 3\text{x}10^{-5}$, $r = 50$ and $k = 32$ based on simulations in Section 4.2.3.

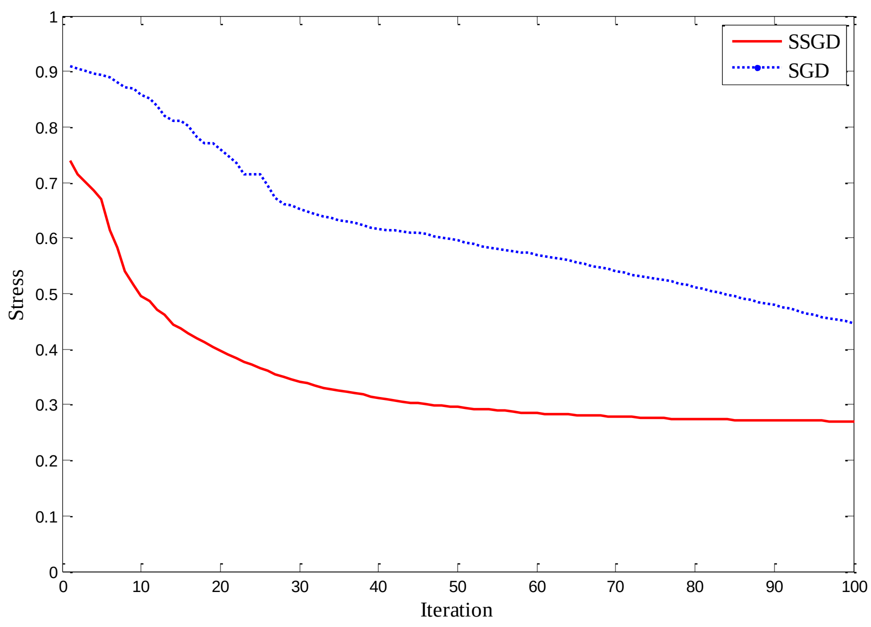


Figure 4.7 Stress plot of SSGD algorithm versus conventional SGD algorithm with random initialization for cross validation set.

It is shown in Figure 4.7 that the proposed SSGD algorithm with SVD initialization yields better convergence compared to conventional SGD with lower number of iteration. Initialization via SVD leads the node's coordinate to start at better initial point rather than random placement. Furthermore, the required number of iteration for convergence is lower compared to conventional SGD. The cumulative distribution function (CDF) plot shows that the plot closer to the upper left corner of the graph represent better estimations with smaller relative errors.

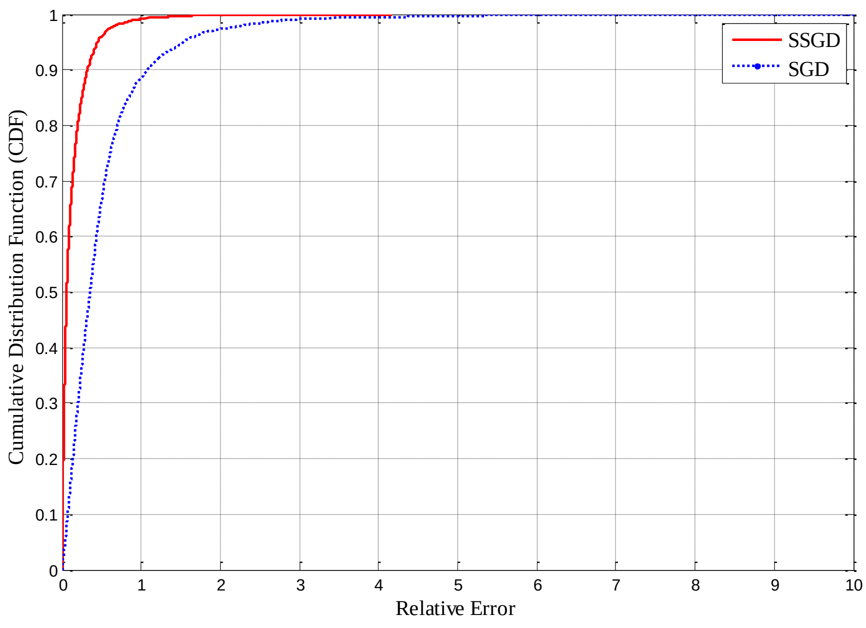


Figure 4.8 Performance comparisons of SSGD and conventional SGD for cross validation set.

On the cross-validation dataset, the proposed SSGD algorithm clearly outperforms conventional SGD in terms of algorithm convergence and prediction accuracy. From Figure 4.8, it can be observed that 90% of all source/destination pairs are predicted with an error below 0.5.

The training parameters are further simulated with another test set. On the test set, SSGD is able to provide better convergence and prediction accuracy compared to conventional SGD, as shown in Figure 4.9 and Figure 4.10.

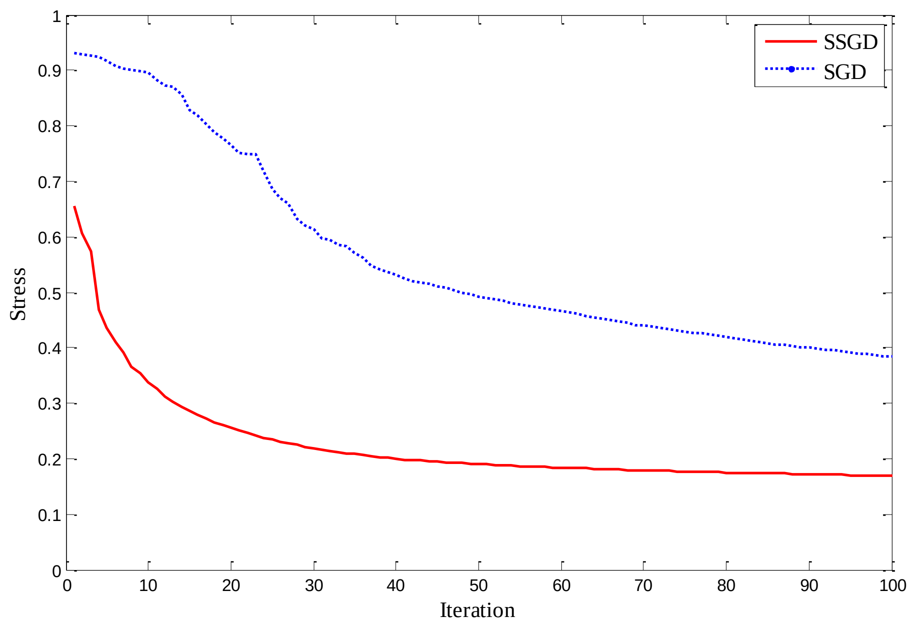


Figure 4.9 Stress plot of SSGD algorithm versus conventional SGD algorithm with random initialization for test set.

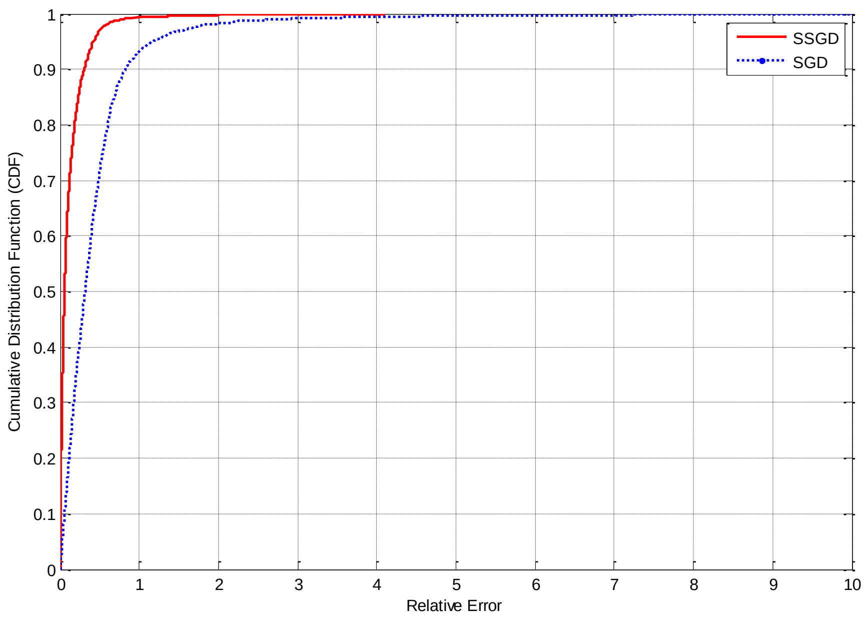


Figure 4.10 Performance comparisons of SSGD and conventional SGD for test set.

It is shown in Figure 4.10 the CDF of relative error is improved whereby 95% of the predicted bandwidth information having relative error less than 0.5. The training parameters are cross-validated and tested and proved to be accurate and suited for prediction in similar network environment.

4.2.5 Performance Comparisons of SSGD, Last-Mile, Sequoia, and DMF

Last-mile (Beaumont, et al., 2011), Sequoia (Ramasubramanian, et al., 2009), and DMF (Liao, et al., 2010) are selected for performance comparison with the proposed algorithm based on literature study presented in Chapter 2.

The parameter constraints in both algorithms are set to the best fitting value for a fair comparison. The approximation rank, *r* is set to be 50 for DMF and SSGD since both algorithms are facing the rank constraint. The accuracy and algorithm convergence are proportional to the number of measurement probed.

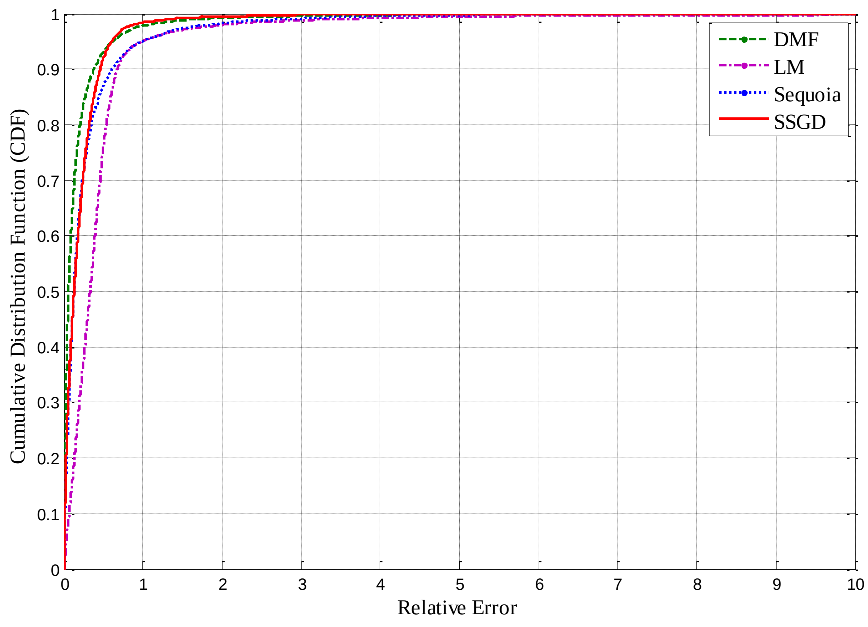


Figure 4.11 Performance comparisons of DMF, Last-Mile (LM), Sequoia, and SSGD.

Last-mile, Sequoia, DMF and SSGD algorithms are tested with the same dataset from *bedibe* project. The plot nearer to the upper left corner of the graph has a better accuracy. In Figure 4.11, it is shown that SSGD has better accuracy compared to Last-Mile and Sequoia and slightly better than DMF with approximately 98% of the source/destination pairs having prediction error between 0 and 1.

4.2.6 SSGD for Latency Prediction

In order to show the applicability of the proposed algorithm to other network performance metrics, such as latency, this section presents the simulation results for latency prediction. The dataset used are the latency measurement from *bedibe* project (Eyraud-Dubois & Uznanski, 2012), consisting of 98 network nodes. The laerning parameters are $\lambda = 1$, $\eta = 3\text{x}10^{-5}$, $r = 50$ and $k = 32$. The experiment is executed for 100 iterations.

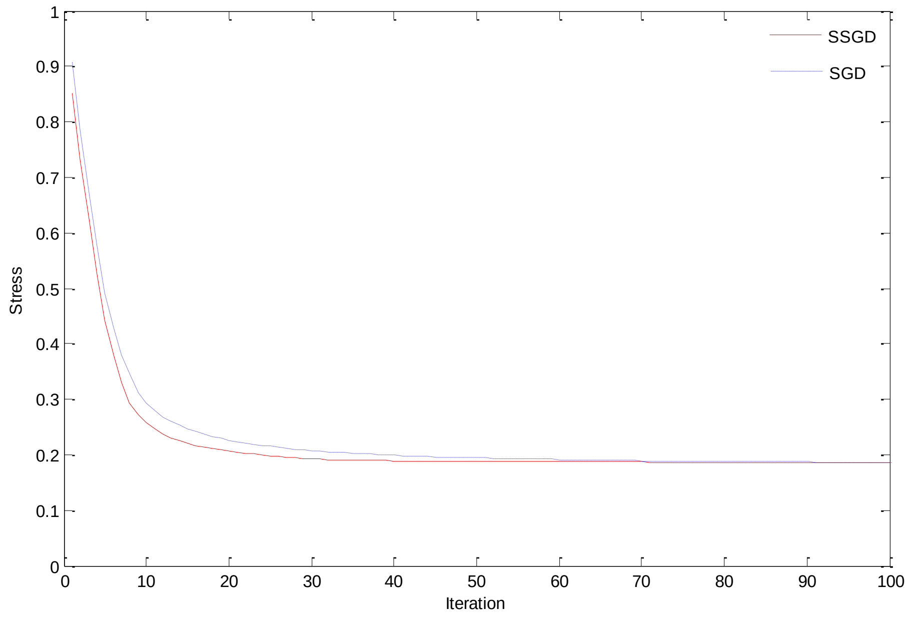


Figure 4.12 The stress plot of algorithm embedding for SSGD versus conventional SGD for latency prediction.

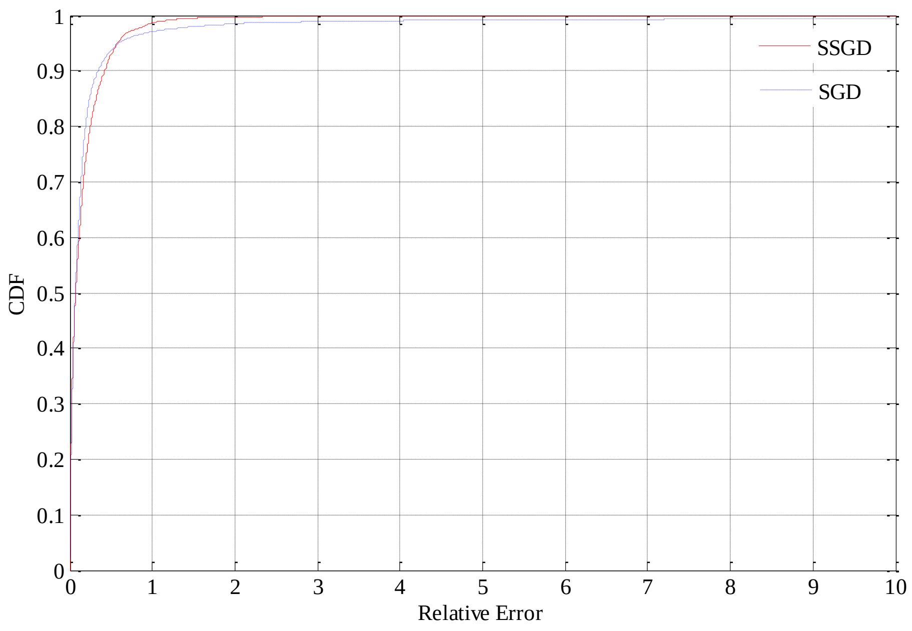


Figure 4.13 CDF plot of the proposed SSGD algorithm for latency prediction.

The stress plot of the SSGD algorithm presented in Figure 4.12 is able to converge to good prediction accuracy at 0.18. The CDF of SSGD shown in Figure 4.13 produces low relative error with approximately 95% of the node pairs have error smaller than 0.5.

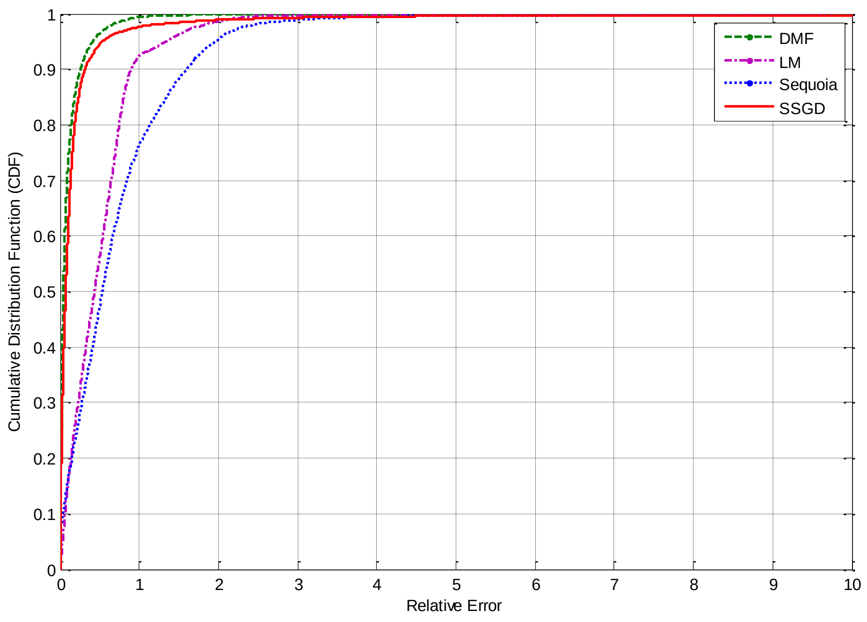


Figure 4.14 Performance comparisons of DMF, Last-mile, Sequoia, and SSGD for latency prediction.

Last-mile, Sequoia, DMF and SSGD algorithms are tested with *bedibe* latency dataset with same network nodes as of available bandwidth dataset. In Figure 4.14, it is shown that SSGD has better accuracy compared to Last-mile and Sequoia and comparable with DMF with approximately 95% of the node pairs having prediction error smaller than 0.5. This proved that SSGD is compatible for both available bandwidth and latency prediction.

4.3 End-to-end Link Quality Prediction

In this section, end-to-end network link quality is classified based on the measured and prediction results obtained in the first part of the algorithm. This provide us with an *O(nxn)* look-up table with *n* number of network links as opposed to one single link. Link quality prediction in this thesis is performed via Logistic Regression (LR) and Support Vector Machine (SVM) to classify the network path in binary and multiclass, and suggest next hop or neighbor selection based on the dynamic network condition. The performance of both classification algorithms was evaluated and it is found that SVM outperformed LR in link quality classification by 20%.

4.3.1 Evaluation Methodology

The aim of machine learning algorithm is to train a set of training samples to obtain the best fitting parameter which can be used as a model to the other compatible dataset. The accuracy of learning algorithms is evaluated as:

$$Accuracy = \frac{P_{correct}}{m} \times 100\% \tag{3}$$

where $P_{correct}$ is the number of correctly predicted data, m is the total number of data. The datasets used for this evaluation were extracted from the *bedibe* project (Eyraud-Dubois & Uznanski, 2012), same as the one used in first part of the proposed algorithm. The available bandwidth and latency on the respective paths are formed as pairs. The dataset is divided in the ratio of 600:200:200 for training set, cross-validation set and test set. The parameter obtained in training set is cross-validated and tested with another compatible set of data.

Tuning parameters such as regularization parameter are essential in machine learning algorithm to obtain a best fit training model. The regularization parameter, λ in regularized logistic regression (RLR) is experimented with different values to obtain the best learning model. The learning model generated from the training set was cross-validated and tested for its accuracy.

There are two training parameters for SVM with RBF (radial basis function) kernel, which is C and γ. This pair of parameter is obtained via grid search and the pair with best cross-validation accuracy is picked. This thesis applied 5-fold cross-validation and LIBSVM (Chang & Lin, 2001) is deployed to run the experiment. In SVM with linear kernel, large values of C tend to minimize misclassification while smaller C values would maximize the margin between boundaries. This in turns too large C values will lead to over-fitting.

4.3.2 Classification Threshold

The quality of video content may be adjusted based on changing network condition. The available bandwidth classification threshold is classified based on the sample distribution in the dataset. Latency is crucial in ensuring quality of audio-video in interactive communication. The acceptable values for one-way delay are within 150ms to 400ms (Karlson, 1996). For instance, scenario with latency larger than 500ms but 5Mbps available bandwidth is still classified as *poor* due to the delay in audio packets. In this condition, images may have been received with out-of-sync audio message under the respective scenario.

With video conferencing being the interest of this thesis, the threshold is set according to Table 4.4 for simulation purposes. For example, with a *very good* quality predicted, the video content is delivered with encoder maximum bit rate of 4096bps.

Table 4.4 Threshold setting for binary and multiclass classification

Classification	Link Quality	Available bandwidth	Latency	Encoder Maximum Bitrate (bps)
Binary	Good	≥ 1Mbps	≤ 0.05s	4096
	Poor	< 1Mbps	> 0.05s	512
Multiclass	Very Good	≥ 3Mbps	≤ 0.03s	4096
	Good	≥ 1Mbps	≤ 0.05s	2048
	Moderate	≥ 100kbps	≤ 0.5s	1024
	Poor	< 100kbps	> 0.5s	512

4.3.3 Logistic Regression (LR) for Classification

This section shows simulation accuracy of LR. The simulation results for binary classification through LR are shown in Table 4.5 below.

Table 4.5 Binary classification by LR

	Training Set	**Cross Validation Set**	**Test Set**
Accuracy (%)	98.17	99.00	99.50

The binary classification obtained through LR presents high accuracy as shown in Table 4.5, with 98.17% of training data correctly classified. The training set data is plotted as shown in Figure 4.15 to illustrate the binary link quality classification.

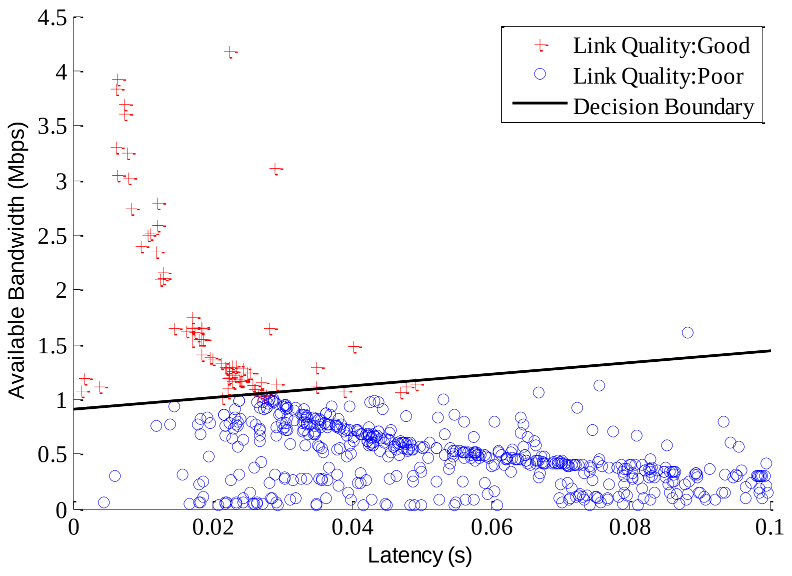


Figure 4.15 Binary classification by LR.

The two axes in Figure 4.15 are the corresponding performance metrics (available bandwidth and latency), which act as the input features to the learning algorithm. The points scattered above the decision boundary are classified as *good* while points distributed below the decision boundary are classified as *poor*.

For regularized logistic regression (RLR), different value of regularization coefficient λ is tested and the accuracy is shown in Table 4.6 respectively.

Table 4.6 Binary classification by RLR tested with different λ

Regularization coefficient, λ	**Accuracy (%)**
	Training Set
1	97.33
10	95.33
100	92.67

Results in Table 4.6 show that $\lambda = 1$ yields the best accuracy. The regularization parameter is included in the training set to prevent over-fitting of the algorithm. Hence, the simulation shows that training accuracy for RLR is slightly lower than normal LR for binary classification. The selected λ is further cross-validated and tested and the results are shown in Table 4.7.

Table 4.7 Binary classification by RLR

	Training Set	**Cross Validation Set**	**Test Set**
Accuracy (%)	97.33	96.50	98.00

The decision boundary generated for RLR as shown in Figure 4.16 produces a straight line from the training model.

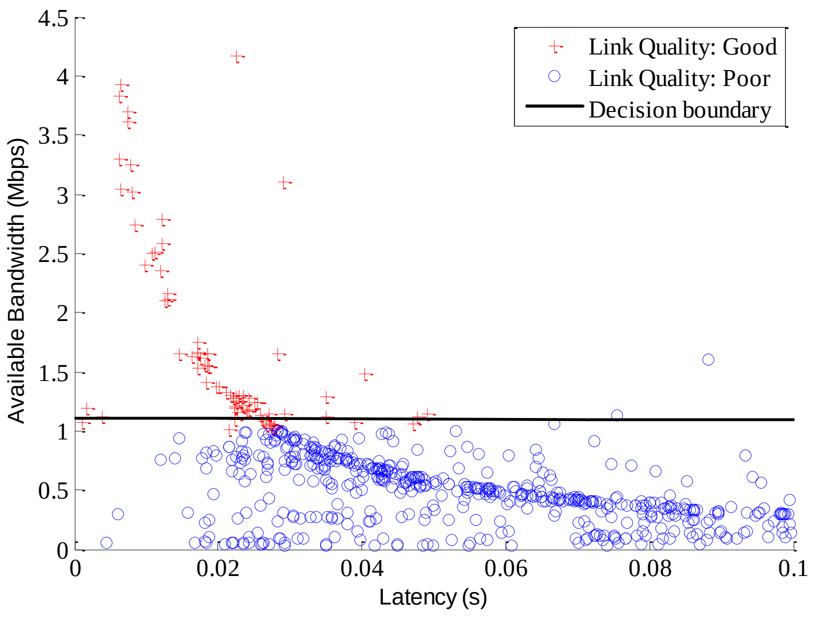


Figure 4.16 Classification by RLR with $\lambda = 1$.

As observed in Figure 4.16 above, decision boundary plotted tend to mis-classify a few outliers and points around the boundary. This is to prevent over-fitting by allowing more misclassification to occur with lesser bias to either side of the classification.

In a multiclass learning algorithm, the goal is to train a classifier that predicts one of *n* classes for each training samples. The multiclass classification via regularized logistic regression is performed through *one-against-all* method. The regularization parameter, $\lambda = 1$ for this simulation is selected based on the best accuracy shown in Table 4.8.

Table 4.8 Multiclass classification by RLR tested with different λ

Regularization coefficient, λ	Accuracy (%)
	Training Set
1	78.33
10	78.16
100	76.50

The accuracy for multiclass RLR is further cross-validated and tested with λ = 1 and the results are presented in Table 4.9.

Table 4.9 Multiclass classification by RLR.

	Training Set	Cross Validation Set	Test Set
Accuracy (%)	78.33	78.00	73.50

The illustration of multiclass classification by RLR is plotted in Figure 4.17. The points were classified based on latency and available bandwidth on the respective network link. The classification threshold is set in accordance to Table 4.4.

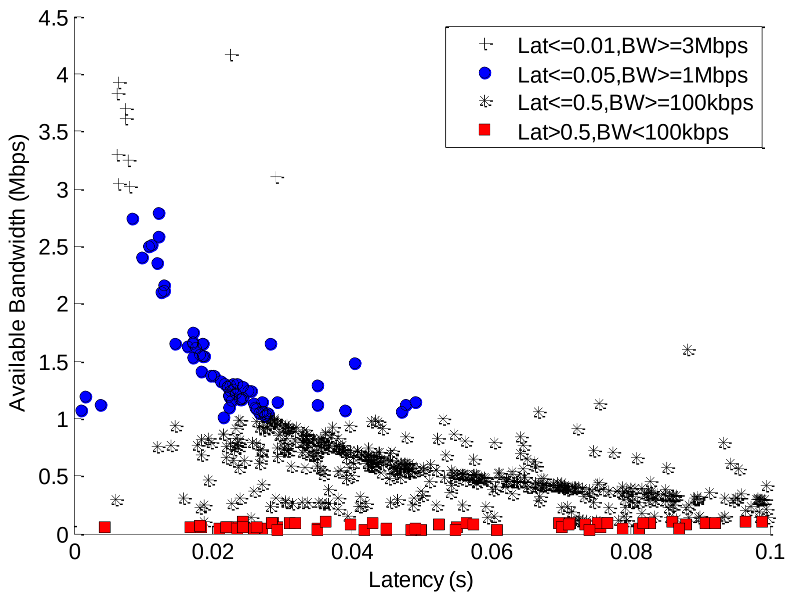


Figure 4.17 Multiclass classification by one-against-all RLR with λ = 1.

This section presents binary and multiclass classification implementation via LR together with their respective accuracies. In conclusion, the training parameters obtained from the training samples produces high accuracy for both the test set and cross-validation set in binary classification. In order to prevent over-fitting, λ is included as regularization parameter. While for multiclass classification, best achievable accuracy obtained from training samples is 78.33%.

4.3.4 Support Vector Machine (SVM) Classification

This section presented the simulation results for binary and multiclass classification via SVM. In this thesis, only SVM with linear and RBF (radial basis function) kernel are studied to examine if the sample distribution is linearly or non-linearly separable. Linear kernel is best applied on linearly separable data while RBF kernel uses a non-linear function to provide better accuracy on non-linearly separable data.

4.3.4.1 Simulation Results (SVM with Linear Kernel)

The parameter C acts similarly with the regularization parameter in LR, and experimented with different value shown in Table 4.10.

Table 4.10 SVM with linear kernel experimented with difference values of *C*

log2c	**Training Set Accuracy (%)**
-10	86.67
0	97.50
5	99.00
10	99.50
15	99.67
20	99.67

The simulation results show that C = 2^{15} and 2^{20} achieved highest training accuracy of 99.67%. It is to note that too large value of C will lead to over-fitting of the algorithm for small training data size. In order to prevent over-fitting, the parameter C is set to 2^{5} and applied in cross-validation set and test set.

Table 4.11 Simulation results for SVM with linear kernel

	Training Set	**Cross-validation Set**	**Test Set**
Accuracy (%) (Binary)	99.00	99.50	98.50
Accuracy (%) (Multiclass)	98.83	98.00	99.00

Table 4.11 shows that SVM with linear kernel is able to achieve high accuracy of 99% for binary and 98.83% for multiclass classification in training set. Cross-validation and test results yield high accuracy of 98% of the samples are correctly classified.

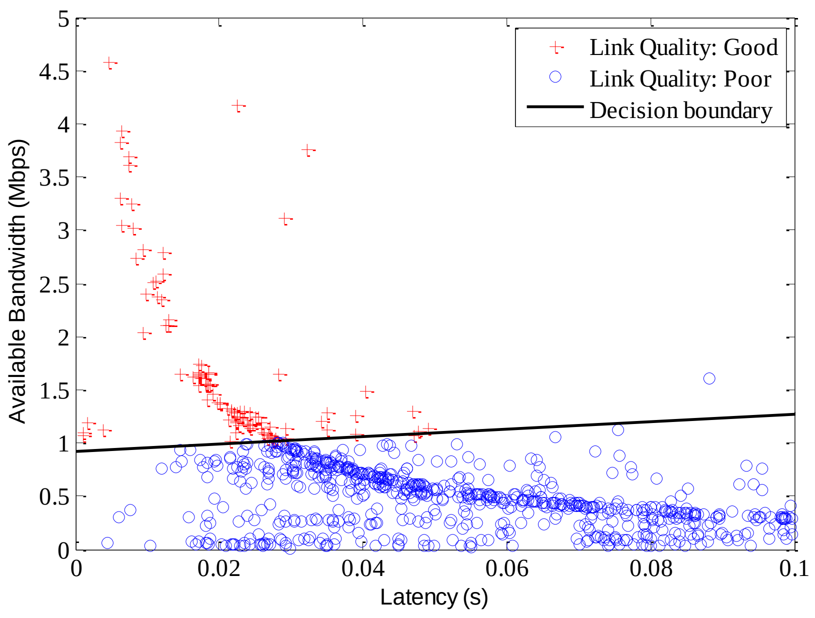


Figure 4.18 Decision boundary plotted for SVM with linear kernel.

The decision boundary for SVM with linear kernel is plotted as in Figure 4.18. The points scattered above the decision boundary are classified as *good* while the points below the decision boundary are considered as *poor*. In other words, the sample can be labeled as poor but yet it falls above the decision boundary. Therefore, 1% of misclassification is observed from the simulation results in Table 4.11. It is shown that the training data is not linearly separable, thus RBF kernel is applied to classify the data in a higher dimensional space in the following section.

4.3.4.2 Simulation Results (SVM with RBF Kernel)

There are two important parameters (C, γ) in SVM with RBF kernel simulation. Grid search is performed to find the best suited pairs and the result is shown in contour plot in Figure 4.19 and Figure 4.20 for binary and multiclass classification respectively. Five-fold grid search is applied as 5 distributions of lines can be observed in the following figures.

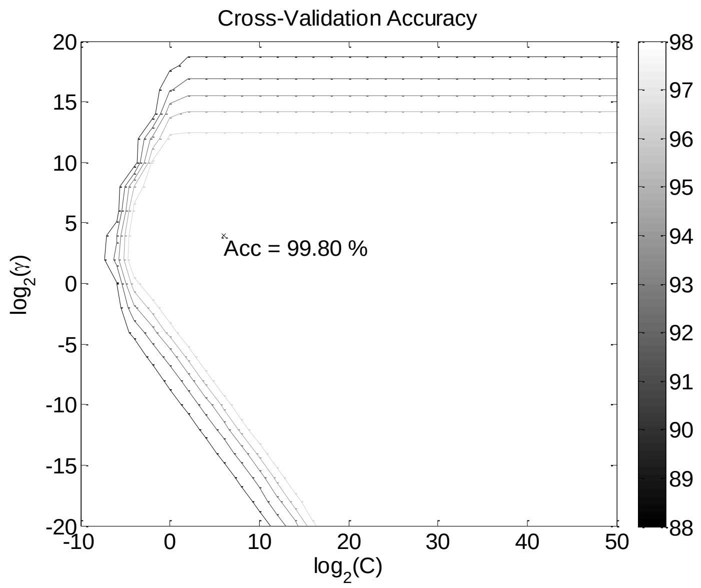


Figure 4.19 Contour of cross-validation accuracy via grid-search for binary classification with SVM.

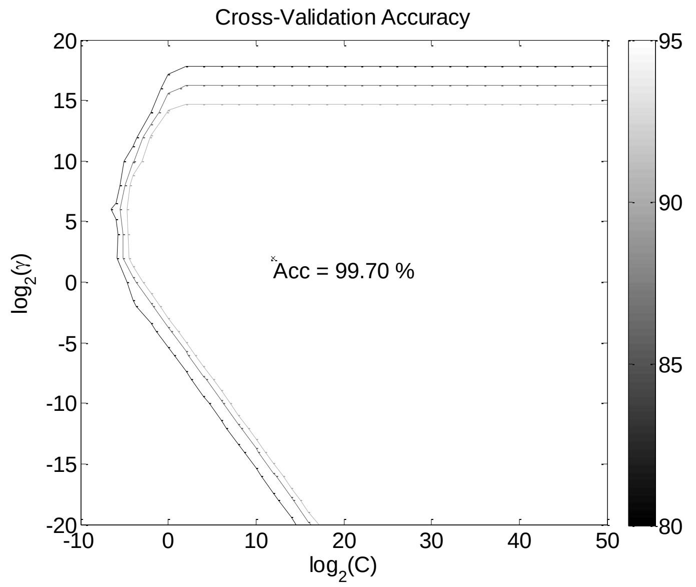


Figure 4.20 Contour of cross-validation accuracy via grid-search for multiclass classification with SVM.

The best cross-validation accuracy is achieved at C = 2^6 and $\gamma = 2^4$ for binary classification while C = 2^{12} and $\gamma = 2^2$ for multiclass classification. Then, the pair of parameter (C, γ) will be used to train a model and test it again with test set.

Table 4.12 Simulation results for SVM with RBF kernel

	Training Set	**Cross-validation Set**	**Test Set**
Accuracy (%) (Binary)	99.17	99.80	99.50
Accuracy (%) (Multiclass: one-against-all)	99.83	99.70	100.00

SVM with RBF kernel is able to achieve high accuracy of 99% for both binary and multiclass classification, as shown in Table 4.12. The accuracy for test set in multiclass classification is 100% with all the sample links are correctly classified. This can be over-fit as well since the test set is small but *C* is set to be large. From the results obtained, SVM is proved to outperform LR in both binary (with slightly higher accuracy) and multiclass classification (with average of 20% higher accuracy).

4.4 Conclusion

This section concludes the simulation results and discussion with performance comparisons among the popular network prediction approaches. In conclusion, as indicated in Section 4.2, the proposed SSGD algorithm manages to improve the convergence and prediction performance by around 20% by initializing SGD using historical bandwidth information. SGD is affected by the setting of parameters as presented in earlier discussion for Figure 4.3 to Figure 4.6. There is no one optimum setting of the parameters which is suitable for all sizes of dataset. Hence, parameter tunings is required to perform on different set of data and cross validated.

In the second part of the thesis, binary and multiclass classification was implemented to classify the link quality between two network nodes. By taking available bandwidth and latency into consideration for link quality prediction, the video quality is adjusted based on the network condition. In this thesis, two major classification approaches, i.e. logistic regression and SVM is implemented to perform the classification task. Logistic regression is easier to be implemented with only one parameter, the regularization coefficient. SVM is able to give high accuracy even with small training dataset. Furthermore, it is also suitable for large classification problem since it is able to classify the data in higher dimensions with the use of kernel function. In the simulations carried out, SVM outperforms logistic regression in both binary and multiclass classification. The training parameters obtained in the simulations are not generalizable as it depends highly on the nature of the dataset, including the network environment and number of network nodes.

The next chapter will discuss on the future work and conclude the thesis.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusions

Network performance prediction poses various challenges such as network stability, metric diversity and costly intrusive measurement. This research was motivated by the growing need to enable best effort Quality of Service on interactive applications via predictable network metrics. The comparison and performance analytics of Euclidean embedding, prediction tree construction, last-mile observation, and matrix factorization approaches for latency and/or available bandwidth prediction is presented in this thesis.

With the recent advances in machine learning, the approach developed in this thesis extends the conventional matrix factorization framework by integrating initialization stage in prior to the stochastic updates of the prediction over time. The initialization stage uses advance network historical data to improve over random initialization which able to improve algorithm convergence. The predicted network metrics are further utilized for network link quality prediction via supervised learning algorithm, namely support vector machine.

An assumption of the SSGD algorithm is made whereby historical data is collected and the network nodes are correlated. The convergence of the algorithm has proved to be improved by 20% as presented in Chapter 4. In real-time implementation, each node will measures to random subset of neighbours before initialization of SSGD algorithm. Aside from the simulation results, real implementation with 4 network nodes is shown in Chapter 5 as well to enable adaptive streaming. Furthermore, it is also shown that SSGD algorithm is able to handle small scale network prediction with less traffic and increased efficiency.

Qualitative optimization enables binary and multiclass classification which is advantageous by reflecting user experience on network performance and also classifies respective link quality with more than one metric. Link quality classification proposed in this thesis is a unique feature which distinguishes it from binary classification presented by (Liao, et al., 2011). It can be easily extended to other network metrics such as packet loss and able to perform multiclass classification with more than one feature.

The objective of this research has been achieved by enabling adaptive streaming utilizing SSGD algorithm. SSGD algorithm is integrated into video conference system which is currently carried out by MIMOS for medical conferencing purpose. The quality of the content delivery is adjusted based on the dynamic network condition to ensure smooth streaming experience.

5.2 Future Work

One of the future directions for this research is the prediction of other network metrics, to study whether the proposed algorithm able to work for other metrics such as packet loss as well. The research can be further extended to perform multiclass classification by involving more than two metrics.

Apart from that, it would be interesting to further study how to adapt the tuneable parameters such as regularization parameter and learning rate in machine learning algorithms to the respective learning environment. This is motivated by the fact that tuneable parameters always pose dilemma by having to pre-set them to fit to a certain learning environment.

Furthermore, it would be good to implement machine learning based classification in practical use for multimedia streaming and also online learning. This is useful for real-time video conference system where large number of nodes communicate frequently. Data can be collected from historical data and update the streaming quality periodically. Moreover, online learning is appropriate for dynamic route convergence for spontaneous connections between large numbers of new nodes.

REFERENCES

Baker, K., 2013. *Singular Value Decomposition tutorial.* [Online]
Available at: http://www.ling.ohio-state.edu/~kbaker/pubs/Singular_Value_Decomposition_Tutorial.pdf
[Accessed 20 August 2013].

Beaumont, O., Eyraud-Dubois, L. & Won, Y. J., 2011. Using the last-mile model as a distributed scheme for available bandwidth prediction. *Euro-Par 2011 Parallel Processing,* Volume 6852, pp. 103-116.

Bottou, L., 1998. *Online algorithms and stochastic approximations,* Cambridge: Cambridge University Press.

Brand, M., 2002. *Incremental singular value decomposition of uncertain data with missing values.* [Online]
Available at:
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.4.361&rep=rep1&type=pdf
[Accessed 20 August 2013].

Buneman, P., 1974. A note on the metric properties of trees. *Journal of Combinatorial Theory,* pp. 48-50.

Chae, M., Shim, S., Cho, H. & Lee, B., 2007. An empirical analysis of fraud detection in online auctions: Credit card phantom transaction. *IEEE 40th Annual Hawaii International Conference on System Sciences,* p. 155a.

Chang, C. C. & Lin, C. J., 2001. LIBSVM: A library for support vector machines. *ACM Transaction on Intelligent Systems and Technology.*

Chang, M. W., Yih, W. T. & Meek, C., 2008. Partitioned logistic regression for spam filtering. *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* pp. 97-105.

Cortes, C. & Vapnik, V., 1995. Support-vector networks. *Machine Learning,* 20(3), pp. 273-297.

Dabek, F., Cox, R., Kasshoek, F. & Morris, R., 2004. Vivaldi: a decentralized network coordinate system. *Proceedings of the 2004 conference on applications, technologies, architectures, and protocols for computer communications,* pp. 15-26.

Dong, C., Wang, G., Yang, C. & Deng, B., 2010. Handling triangle inequality violations in Euclidean distance based network coordinate systems. *18th IEEE International Workshop on Quality of Service (IWQoS),* pp. 1-2.

Eugene Ng, T. S. & Zhang, H., 2002. Towards Global Network Positioning. *ACM SIGCOMM Computer Communication Review,* 32(1), p. 61.

Eyraud-Dubois, L. & Uznanski, P., 2012. Bedibe: Datasets and software tools for distributed bandwidth prediction. *14th Algorithmic Aspects of Telecommunications.*

Hsu, C. W., Chang, C. C. & Lin, C. J., 2010. *A practical guide to support vector classification.* [Online]
Available at:
https://www.cs.sfu.ca/people/Faculty/teaching/726/spring11/svmguide.pdf
[Accessed 12 September 2013].

Hu, N. & Steenkiste, P., 2005. Exploiting internet route sharing for large scale available bandwidth estimation. *Proceedings of the 5th ACM SIGCOMM conference on Internet,* pp. 16-16.

Jain, M. & Dovrolis, C., 2002. pathload: A measurement tool for end-to-end available bandwidth. *Proceedings of 3rd Passive and Active Measurements Workshop.*

Kaafar, M. A., Cantin, F., Gueye, B. & Leduc, G., 2009. Detecting triangle inequality violations for Internet coordinate systems. *IEEE International Conference on Communications*, pp. 1-6.

Karlson, G., 1996. Asynchronous transfer of video. *IEEE COmmunications Magazine,* pp. 118-126.

Klema, V. & Laub, A. J., 1980. The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control,* 25(2), pp. 164-176.

Komarek, P., 2004. *Logistic regression for data mining and high-dimensional classification,* Pennsylvania: Camegie Mellon University.

Koren, Y., Bell, R. & Volinsky, C., 2009. Matrix factorization techniques for recommender systems. *IEEE Computer Society,* pp. 30-37.

Lee, D. D. & Seung, H. S., 2001. Algorithms for non-negative matrix factorization. *NIPS, MIT Press,* pp. 556-562.

Liao, Y., Du, W., Geurts, P. & Leduc, G., 2012. DMFSGD: A decentralized matrix factorization algorithm for network distance prediction. *IEEE/ACM Transactions on Networking.*

Liao, Y., Geurts, P. & Leduc, G., 2010. Network distance prediction based on decentralized matrix factorization. *Proceedings of IFIP Networking 2010*, pp. 15-26.

Mack, Y., 2010. *K-Nearest neighbor estimation,* California: Google Books.

Menard, S., 2010. *Logistic regression: Introductory to advanced concepts and applications,* New York: SAGE Publications.

Pathak, A. et al., 2008. A measurement study of Internet delay asymmetry. *Proceedings of the Passive and Active Measurement*, pp. 182-191.

Ramasubramanian, V. et al., 2009. On the treeness of internet latency and bandwidth. *ACM SIGMETRICS/Performance '09.*

Ribeiro, V. J. et al., 2003. pathChirp: Efficient available bandwidth estimation for network paths. *Passive and Active Measurement Workshop.*

Schober, S., Brenner, S., Kapitza, R. & Hauck, F. J., 2013. Bandwidth prediction in the face of asymmetry. *IFIP International Federation for Information Processing*, pp. 99-112.

Timmerman, D. et al., 2005. Logistic regression model to distinguish between benign and malignant adnexal mass before surgery: a multicenter study by the international ovarian tumor analysis group. *Journal of Clinical Oncology,* 23(34), pp. 8794-8801.

Xing, C., Chen, M. & Yang, L., 2009. Predicting available bandwidth of internet path with ultra metric space-based approaches. *Proceedings of the 28th IEEE Conference on Global telecommunications*, pp. 584-589.

Yun, M., Saul, L. & Smith, J. M., 2006. IDES: An internet distance estimation service for large networks. *IEEE Journal on Selected Areas in Communications,* pp. 2273-2284.

Doval, D. & O'Mahony, D., 2003. Overlay networks: A scalable alternative for P2P. *IEEE Internet Computing*, pp. 79-82.

Carter, R.L, & Crovella, M.E., 1999. On the network impact of dynamic server selection. *Computer Networks*, pp. 2529-2558.

APPENDIX A

PUBLICATION BY THE AUTHOR

The work presented in this thesis has been published in the following papers.

- Lim, S.J., et. al., 2013. Implementing stochastic gradient descent based on historical network distance for available bandwidth prediction. T*he 2nd International Conference on e-Technologies and Networks for Development*, pp. 202 - 207.
- Lim, S.J., Lee, S.W., Simon, L., Karuppiah, E., 2013. Link quality prediction for multimedia streaming based on available bandwidth and latency. The 7th IEEE Workshop on Network Measurements, pp. 1025 - 2013.

IMPLEMENTING STOCHASTIC GRADIENT DESCENT BASED ON HISTORICAL NETWORK DISTANCE FOR AVAILABLE BANDWIDTH PREDICTION

Lim Su Jin
Faculty of Engineering and Science
University Tunku Abdul Rahman, Malaysia
limsujin@hotmail.my

Lim Boon Ping
MIMOS Berhad
Technology Park Malaysia,
57000 Kuala Lumpur, Malaysia
bp.lim@mimos.my

Lee Sze Wei
Faculty of Engineering and Science
University Tunku Abdul Rahman, Malaysia
leeszewei@utar.edu.my

Simon Lau
Faculty of Engineering and Science
University Tunku Abdul Rahman, Malaysia
simonlau@utar.edu.my

Ettikan Karuppiah
MIMOS Berhad
Technology Park Malaysia,
57000 Kuala Lumpur, Malaysia
ettikan.karuppiah@mimos.my

Shahirina Mohd Tahir
MIMOS Berhad
Technology Park Malaysia,
57000 Kuala Lumpur, Malaysia
shahirina.mtahir@mimos.my

ABSTRACT

Predicting network bandwidth of large systems based on a few pairs of network nodes is essential to overcome large measurement overhead over full-mesh active measurements. Recently, prediction using low-rank matrix factorization has gained attention. The algorithm is fully decentralized where no explicit matrix constructions or special nodes such as landmarks and central server is needed. Prediction error and convergence to global minimum are two major concerns of this type of algorithm. In this paper, we propose to enhance low-rank matrix factorization by Stochastic Gradient Descent (SGD) initialized with Singular Value Decomposition (SVD). Experimental results show enhanced prediction error and convergence performance is achieved through our approach.

KEYWORDS

Available bandwidth prediction, matrix factorization, historical network distance, Singular Value Decomposition, Stochastic gradient descent

1. INTRODUCTION

Available bandwidth between two nodes is the maximum throughput that a flow between two hosts can achieve in the presence of cross-traffic. Data-intensive applications such as multimedia streaming are directly impacted by the available bandwidth. Heterogeneous network environment may cause streaming quality to vary significantly, and hence, affect the overall quality. This will lead to fluctuation of the clarity of the multimedia content. The knowledge of available bandwidth information can be used as optimization parameter in the changing network condition to deliver predictable results in order to achieve good Quality-of-Service (QoS). However, performing active probing between all network nodes is expensive and incurs large measurement overhead. Hence, it is imperative to measure only a subset of the network nodes for QoS support without having to probe every network node.

This paper aims to present an enhanced low-rank matrix factorization algorithm for available bandwidth prediction through Stochastic Gradient Descent (SGD) initialized with Singular Value Decomposition (SVD). SGD is a fully decentralized consisting of only vector operation. With its simplicity, each node will equally probe the same number of nodes known as neighbour nodes iteratively to update distance measurement one at a time. In real network environment, nodes join and leave a network frequently and the measurements vary over time. So, SGD is highly adaptive and simple to implement in actual Internet applications.

In this work, the low-rank matrix factorization algorithm is initialized with historical available bandwidth data to achieve better convergence to global minimum and prediction error. Since SVD cannot handle missing element in the available bandwidth information, matrix row and column mean normalization were applied on the missing elements. The predicted distance matrix is approximated by two smaller matrices. SVD provides these two smaller matrices by decomposing the matrix based on historical bandwidth information.

In the following section, we summarize some of related work in this research area.

2. RELATED WORK

This section discusses the existing network distance prediction approaches by focusing on matrix factorization based approaches. These approaches adopt either landmark-based or decentralized-based methods. For landmark-based methods, the network distances between the landmark nodes are measured and ordinary nodes only probe landmark nodes. In the decentralized-based methods each network node equally probes a number of other nodes known as neighbour nodes. Matrix factorization was first introduced in Internet Distance Estimation Service (IDES) [1] for latency prediction. IDES is a landmark-based matrix factorization method where the algorithm predicts large number of network distances from limited samples of Internet measurements. Landmark-based system often suffers from several drawbacks including landmark node overload and failures. The landmark selection is crucial as it can also affect the accuracy of the prediction.

SGD is a method often used for online machine learning [2]. Instead of collecting all training samples, each epoch of SGD chooses one random sample and updates the estimated network distance along the negative gradients. Decentralized-based matrix factorization approach is adopted in Decentralized matrix factorization by Stochastic Gradient Descent (DMFSGD) [3] to perform network distance prediction. The algorithm is fully decentralized where each node exchanges messages with other nodes and processes local measurement without explicit matrix construction or landmark node. In DMFSGD, the algorithm operates at each node with measurement carried out one-by-one. DMFSGD algorithm is efficient and simple to implement as measurements can be acquired on demand and processed locally at each node. Furthermore, the update rules only involve vector operations and are able to deal with large-scale dynamic measurements. As far as we know, DMFSGD algorithms had been implemented and tested based on latency information but is yet to be implemented and tested on available bandwidth. Hence, this work aims to enhance the algorithm and implement it for the prediction of available bandwidth information.

3. CONTRIBUTION

In this paper, we propose an algorithm named SVD-SGD (SSGD) based on matrix factorization by stochastic gradient descent (SGD) which makes use of the network historical distances as initialization to allow better convergence to global minimum and improve prediction error.

4. SSGD ALGORITHM

The SSGD algorithm is presented in this section. Each network node is assumed to keep a set of bandwidth information. In a changing network condition, a random node will be selected at each epoch to update the bandwidth information. The missing elements in historical bandwidth information can be estimated by mean normalization as in function (1). The *n* nodes in the network will form an *n* × *n* distance matrix with some distances between nodes measured and others unmeasured. The distance matrix *D* is treated as a historical network distances. The missing elements are replaced by mean normalization using the following function

$$\widehat{M}_{i,j} = \min\left(\overline{M}_{j^{th}\,row}, \overline{M}_{j^{th}\,col}\right) \qquad (1)$$

where $\widehat{M}_{i,j}$ is the missing data from node i to node j, $\overline{M}_{j^{th}\,row}$ is the mean of row i^{th} and $\overline{M}_{j^{th}\,col}$ is the mean of j^{th} column.

Details of the algorithm are listed as follows:

SSGDAlgorithm

```
1:  Input: $D$ distance matrix
2:  Predict the missing element in $D$ using
    mean normalization function in (1)
```

```
        i)   search zero elements in distance matrix
        ii)  replace zero with the minimum of average of row and column vector
3:      Execute SVD to obtain the two smaller matrices $X$ and $Y$
        i)   decompose historical distance matrix using SVD function in (2)
        ii)  initialize $X$ and $Y$ based on i)
4:      Node $i$ retrieves incoming and outgoing available bandwidth actively or passively
5:      Perform SGD:
        i)   select a random node at each iteration
        ii)  update $x_i$ and $y_i$ to minimize loss function in (3)
```

SVD is implemented as the initialization step to guarantee better convergence of the algorithm. Generally, SVD decomposes a given matrix D into three smaller matrices of the form [4]

$$D = USV^T \tag{2}$$

where U and V are unitary matrices, S is a diagonal matrix with nonnegative real numbers on the diagonal, which are called singular values. This number is equal to the rank of D. To obtain a low-rank factorization, only the r large singular value in S are kept and the rest are replaced by zero. These three smaller matrices are formed into two smaller initialization matrices X and Y in SGD. Let S_r be the new S, $X = US_r^{1/2}$ and $Y^t = S_r^{1/2}V^T$, the predicted distance matrix $\widehat{D} = XY^T$ is then the optimal low-rank approximation to D.

The goal of the above mentioned steps is to approximate the network distance with two smaller matrices by minimizing the regularized loss function [3] given by

$$L(D,X,Y,W,\lambda) = \sum_{i,j=1}^{n} \omega_{ij}\, l(d_{ij}, x_i y_j^T) + \lambda \sum_{i=1}^{n} x_i x_i^T + \lambda \sum_{i=1}^{n} y_i y_i^T \tag{3}$$

where λ is the regularization coefficient to avoid over-fitting, ω_{ij} is the weight representing 1 if d_{ij} is measured and 0 otherwise, each row of X and of Y is denoted by x_i and y_i and called x and y coordinates. The loss function, l is the most commonly used square loss function,

$$l(d,\hat{d}) = (d-\hat{d})^2 \tag{4}$$

When SGD is used, each node measures with respect to one neighbour at a time and retrieves the node's coordinates. Each node i then updates its coordinates along the negative gradient directions given by

$$x_i = (1-\eta\lambda)x_i + \eta(d_{ij} - x_i y_j^T)y_j \tag{5}$$

$$y_i = (1-\eta\lambda)y_i + \eta(d_{ij} - x_j y_i^T)x_j \tag{6}$$

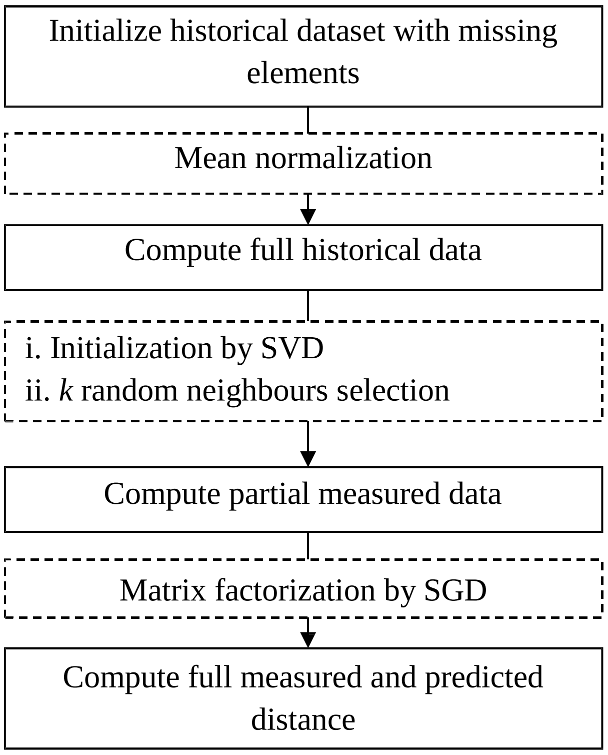
where η is the learning rate which controls the speed of the updates.

With the decomposition of the two smaller matrices from the historical bandwidth information by SVD, the algorithm is able to enhance the convergence of SGD to global minimum and improve prediction error.

5. EVALUATION RESULTS AND DISCUSSIONS

Evaluation setup

The algorithm is illustrated in Fig. 1.



Initialize historical dataset with missing elements
Mean normalization
Compute full historical data
i. Initialization by SVD
ii. k random neighbours selection
Compute partial measured data
Matrix factorization by SGD
Compute full measured and predicted distance

Figure 1: Overview of the algorithm flow

Input datasets are adopted from the S-cube project [5], a project implemented with scalable sensing service for real-time and configurable monitoring and management system for large networked systems. The available bandwidth dataset consists of asymmetric measurement of 99 nodes with 11.23% of missing element. Mean normalization is performed over the missing elements to construct a full historical data. With this full historical distance data, we are able to initialize the two smaller matrices *X* and *Y* by SVD which yields better initial position to converge to global minimum. This results in getting better starting point for convergence rather than random initialization which requires more iterations of the algorithm.

A random node is selected at each iteration of the algorithm update the coordinates along negative gradient. Finally, a full distance matrix can be obtained with measured and predicted bandwidth information. The relative error of measured and predicted bandwidth information are calculated based on the input matrix and the full measured and predicted distance matrix.

Evaluation metrics

Quality of the prediction algorithm is measured by the cumulative distribution function (CDF) of relative error for all pairs of hosts. The relative error is defined as

$$RE_{x,y} = \frac{|M_{x,y} - P_{x,y}|}{M_{x,y}} \qquad (6)$$

where $M_{x,y}$ is the measured value and $P_{x,y}$ is the predicted value. The CDF plots of relative errors indicate better estimation performance the higher the corresponding plot is.

The overall fitness of the embedding is measured by stress, which is used to illustrate the convergence of the algorithm, defined as follows

$$stress = \sqrt{\frac{\sum_{x,y=1}^{n}\left(M_{x,y} - P_{x,y}\right)^2}{\sum_{x,y=1}^{n} M_{x,y}^2}} \qquad (7)$$

High variance in the available bandwidth dataset is undesirable and will lead to low convergence of the prediction algorithm. High prediction errors are normally due to over-estimated or under-estimated of the bandwidth information. This should be avoided since it affects the quality of AV delivery with poor bandwidth responsiveness. Performance results and discussions of the proposed SSGD algorithm against conventional SGD are presented in the following section.

Results and discussions

The stress plots are shown below with comparison of different ranks of approximation, *r* regularization coefficients, *λ* and learning rates, *η* for the available bandwidth dataset. The number of neighbours selected, *k* is set to 32 in this experiment and run on 100 iterations for different parameter settings. Larger values of *k* would improve the prediction accuracy.

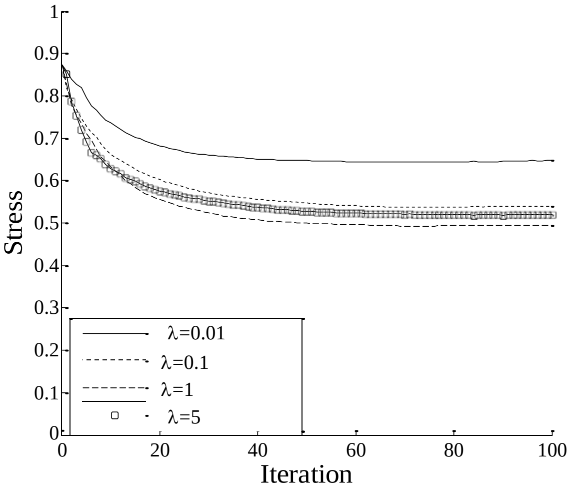


Figure 2: Stress of the algorithm embeddings with different λ. η = 3x10^{-5}, *r* = 10.

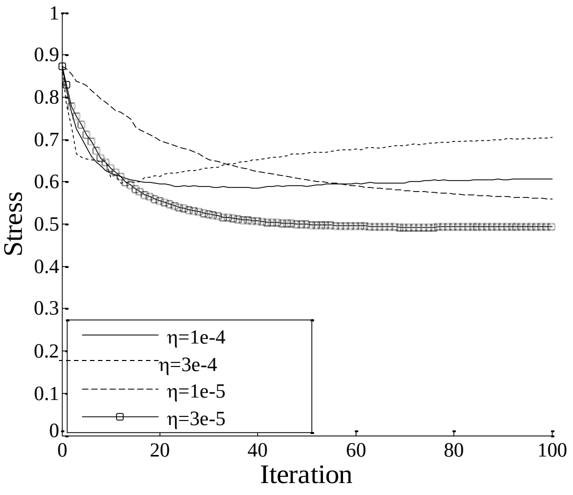


Figure 3: Stress of the algorithm embeddings with different η. λ = 3x10^{-5}, *r* = 10.

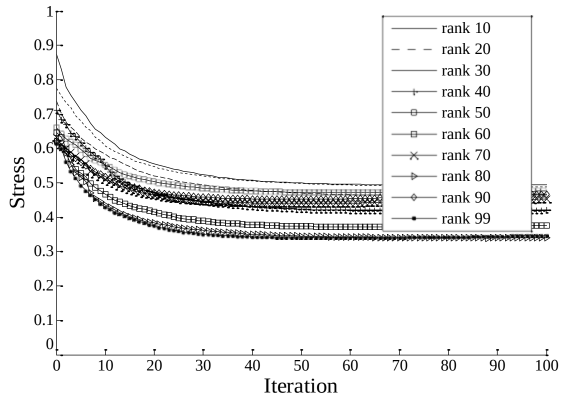


Figure 4: Stress of the algorithm with different *r*. $\lambda = 1$, $\eta = 3x10^{-5}$.

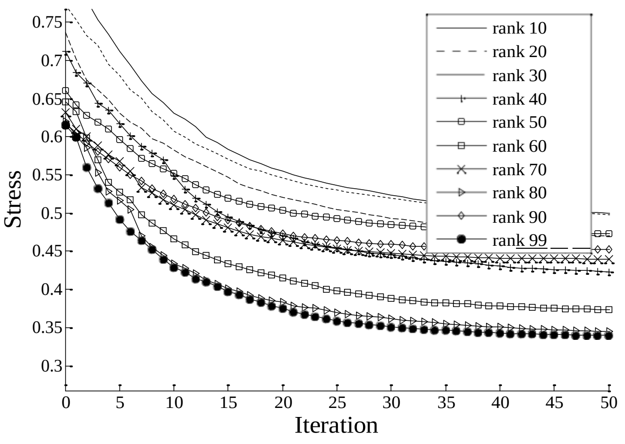


Figure 5: Detail view of stress of the algorithm with different *r*. $\lambda = 1$, $\eta = 3x10^{-5}$.

The regularization coefficient, λ reduces over-fitting in the algorithm and drift of the coordinates. Besides, too large λ may lead to over-fitting which introduced higher bias in the estimate for dataset with high variance. We experimented with $\lambda = \{0.01, 0.1, 1, 5\}$, $\eta = 3x10^{-5}$ and $r = 10$. It is clear from Fig. 2 that $\lambda = 1$ shows better convergence. With λ being too small (<1) poor convergence is experienced. Thus, λ is set to 1 for the rest of the experiments.

The learning rate or often referred to as step size affects the speed of convergence of the algorithm. Too large value for η may also lead to divergence in the algorithm. As shows in Fig. 3, with $\eta = 1x10^{-4}$ and $\eta = 3x10^{-4}$, the algorithm diverges after some iterations. There is no optimum setting of this parameter but $\eta = 3x10^{-5}$ is set in our tests since it shows best performance for this dataset.

The rank-constraint in SVD affects the performance of the algorithm by changing the number of singular values, which represents the approximation rank. The approximation rank, *r* is set to 10 for the ease of comparison although higher *r* will render better result as shown in Fig. 4. Lower values of *r* require less matrix operation as rows and columns indexed higher than *r* are filled with zeros and this enable faster computation of the algorithm.

It is evident from Fig. 5 and Fig. 6 that using SVD as initialization over random initialization improves the convergence and prediction error. The parameters were set as $\lambda = 1$, $\eta = 3x10^{-5}$, $r = 10$ and $k = 32$ based on earlier tests to optimize the algorithm.

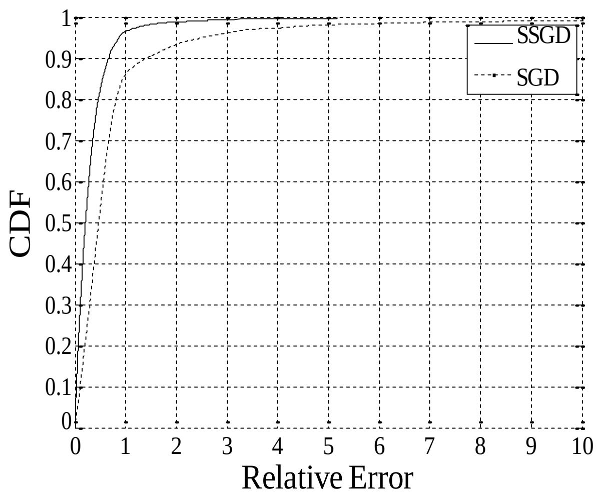


Figure 6: CDF of relative error for SGD and SSGD algorithm.

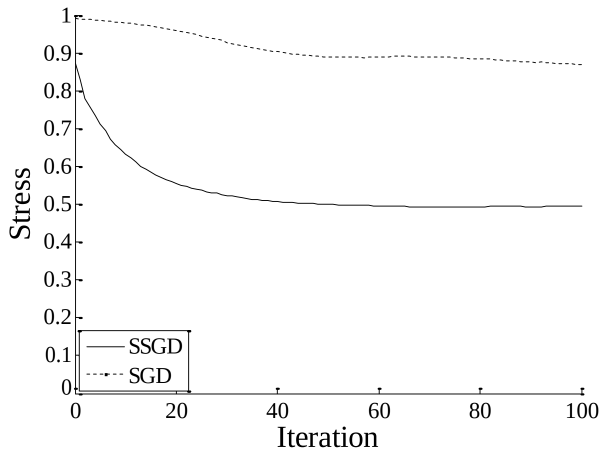


Figure 7: Stress of the SSGD algorithm embeddings compared to SGD after each iteration.

It is shown in Fig.5 the CDF of relative error is improved when SVD initialization is performed through stochastic gradient descent (SGD) method using historical network distances with approximately 95% of the predicted bandwidth information having relative error of 0 to 1. Furthermore, convergence also shows improvement via SSGD as shown in Fig. 6.

To sum up, as indicated in Fig. 5 and Fig. 6, the proposed SSGD algorithm manages to improve the convergence and prediction performance by around 10% by initializing SGD using historical bandwidth information. The convergence of the algorithm also shows improvement with approximately 50% compared to SGD algorithm. SGD is affected by the setting of parameters as presented in earlier discussion for Fig. 2 to Fig. 4. There is no one optimum setting of the parameters which is suitable for all sizes of dataset.

6. CONCLUSION

Predicting available bandwidth between a transmitting and a receiving node in a large scale network is crucial in delivering better QoS for data intensive application such as multimedia streaming. In this regard, mesh measurement is computationally expensive and hence it is an ongoing challenge for the research community to predict the bandwidth information from a subset of nodes. However, ensuring good convergence of predicted values as well as achieving high accuracy is challenging. In this paper, we have proposed to enhance conventional matrix factorization by SGD by the use of SVD where each node maintains a set of historical data and the matrix is decomposed into two smaller matrices as pre-processing step. Convergence and prediction accuracy of the proposed SSGD algorithm are significantly improved with the proposed enhancements. In the future, we are going to investigate on the influence of variance in dataset and the neighbour selection and run the tests with larger dataset.

7. REFERENCES

1. Y. Mao, L. Saul, and J. M. Smith: IDES: An Internet distance service for large networks. In: *IEEE Journal on Selected Areas in Communications*, vol.24, no. 12, pp. 2273-2284 (Dec. 2006).
2. L. Bottou: Online algorithms and stochastic approximations. In: *Online Learning and Neural Networks*, D. Saad, Ed. Cambridge University Press (1998).
3. Y. Liao, W. Du, P. Geurts, and G. Leduc: DMFSGD: A decentralized matrix factorization algorithm for network distance prediction. In: *IEEE/ACM Trans. on Networking* (2012).
4. G. W. Stewart: On the early history of Singular Value Decomposition. In:*SIAM Review*, pp 551-566 (1992).
5. P. Yalagandula, P. Sharma, S. Banerjee, S. Basu, and S. J. Lee: S3: A scalable sensing service for monitoring large networked systems. In:*ACM SIGCOMM workshop on Internet network management*, pp 71-76, Pisa, Italy (Sept. 2006).

Link Quality Prediction for Multimedia Streaming based on Available Bandwidth and Latency

Lim Su Jin
Univ. Tunku Abdul Rahman
Kuala Lumpur, Malaysia
limsujin@hotmail.my

Lee Sze Wei
Univ. Tunku Abdul Rahman
Kuala Lumpur, Malaysia
leeszewei@utar.edu.my

Simon Lau
Univ. Tunku Abdul Rahman
Kuala Lumpur, Malaysia
simonlau@utar.edu.my

Ettikan Karuppiah
MIMOS Berhad Kuala
Lumpur, Malaysia
ettikan.karuppiah@mimos.my

Abstract—**Network performance metrics such as available bandwidth and latency are essential to achieve good Quality of Service (QoS) in multimedia streaming. There are unique requirements in network performance metrics for media applications, such as audio conferencing, video streaming, video conferencing, and high-definition (HD) video conferencing. In this paper, we focus on conference call type suggestion based on link quality prediction. The link's quality is classified based on the available bandwidth and latency between two network nodes. We have implemented and compared two of the most popular supervised learning based classification methods, i.e. logistic regression and support vector machine (SVM). We have compared the performance of both methods and their suitability to apply in link quality prediction. The experimental results show that SVM outperforms logistic regression for binary and multiclass classification in terms of accuracy.**

Keywords – multimedia streaming, classification, logistic regression, support vector machine.

I. Introduction

The on-growing usability in video conferencing system has led to high demand for better quality Audio/Video (AV) transmission. Network constraints are the major problem in multimedia streaming. Network performance metrics serve as optimization parameters for adaptive streaming. Available bandwidth is the maximum throughput between two hosts that can be achieved in the presence of cross traffic while latency is often referred to as time delay between two network nodes. Available bandwidth is important for video conferencing over the Internet to ensure a smooth image quality received. Meanwhile, latency is an essential metric in synchronizing between video and audio received. High latency would cause deteriorated call quality on the receiver and out-of-sync with the video images.

Due to these these considerations and requirements in video conferencing, we propose to predict the link quality between network nodes based on classification of network link based on available bandwidth and latency. Most of the existing approaches only consider either bandwidth or latency solely in their algorithm. Through our proposed supervised learning classification approach, we are able to consider both network performance metrics at once. We suggest to combine both metrics in evaluation based on the requirement of the application. For example, in video conferencing system, available bandwidth and latency plays equally important role in ensuring AV quality in synchronized pattern.

We apply logistic regression and SVM in this paper. These supervised-learning algorithms acquire knowledge from the previous training samples and adapt the system with a new model which is used for prediction on a new input data. Logistic regression (LR) analysis [1] with sigmoid function produces results between 0 to 1. These results can be classified into binomial or multinomial by setting the classification threshold. LR is frequently used to estimate qualitative response models in which the dependent variable is a dichotomy, such as email spam filtering [2], fraudulent detection for online transactions [3], and tumor malignancy classification [4]. The advanced optimization algorithms, such as gradient descent, are often applied in LR analysis to minimize the cost function iteratively.

Support Vector Machine (SVM) [5] is a statistical machine learning technique which learns from a training dataset and attempts to generalize and make correct predictions on new input data. The kernel-based SVMs are able to handle many types of data within the same model which encourage the flexibility of the learning algorithm. SVM has been successfully applied in many applications, such as TCP traffic classification [6], text classification [7] and more commonly in bioinformatics [8].

The key research contributions of this paper are listed as follows:

- Novel link quality prediction via LR and SVM classification are performed.
- Network performance metrics (available bandwidth and latency) are associated in pairs in classification which is useful for interactive AV applications.

The rest of this paper is organized as follows. In Section II, the related works of link quality prediction are discussed. It is followed by the implementation detail of logistic regression and support vector machine in Sections III and IV. The evaluation methodology used in this paper is explained in Section V. The experimental results and discussion are presented and discussed in Section VI and finally concluded in Section VII.

II. RELATED WORK

Network performance prediction is very relevant to the reduction of the overhead associated with continuous measurements. Current approaches for carrying out network performance prediction are based on network latency and available bandwidth normally measured and analyzed separately. Related works for latency prediction include Vivaldi [9], Global Network Positioning (GNP) [10], and Internet Distance Estimation Service (IDES) [11]. Vivaldi faces challenge in violation of triangle inequality while the accuracy of GNP and IDES are influenced by the landmark node selection. The prediction of available bandwidth as proposed by Last-Mile Model [12] is only limited to small scale of data where the accuracy is highly dependent on the number of neighbours selected to iteratively minimize the discrepancies between measured and predicted values. The tree embedding approach, introduced in Sequoia [13] faces constraints on triangle inequality violation, inability to predict asymmetric information, and the influence of the node selection process. Direction-aware embedding is proposed in [14] separating upstream from downstream properties of the hosts to tackle the limitations faced in Sequoia.

Work presented in [15] is the only related work that predicts end-to-end performance classes through decentralized approach based on stochastic gradient descent. The algorithm performs matrix completion with binary performance measures between network nodes with known and unknown nodes to be filled. This approach can be applied on both available bandwidth and latency, but the implementation for latency and available bandwidth is separated. The prediction is also performed in quantitative way, by filling the matrix with measured distance metric and predict the unmeasured through matrix completion (DMFSGD) [16]. The authors have recently extended their work to perform ordinal rating of network performance by network inference based on performance ratings [17].

We have performed linear regression analysis, by optimizing matrix completion problem with stochastic gradient descent in our previous work [18]. The unmeasured distance metrics are first predicted through interpolation, and initialized with singular value decomposition (SVD) before the optimization step. It has proven to be effective in improving the convergence of algorithm to global minimum. This paper is an extension to our previous work, to further enhance the prediction and adapt the algorithm to AV streaming applications through binomial and multinomial classification. The available bandwidth and latency are associated in pairs from our prediction algorithm previously.

LR and SVM classification approaches are described in the following section.

III. LOGISTIC REGRESSION (LR)

Logistic regression [19] is a regression technique suitable for data with binary outcomes {0, 1}. It builds a model from training samples and predicts the probability of the network link to be 0 (Good) or 1 (Bad). The input features are the available bandwidth and latency. In this section, we discuss how the class-based link quality is predicted through LR, whereby a set of training samples will be trained to obtain learning parameter, θ and regularization coefficient, λ for regularized logistic regression through gradient descent.

A. LR Model

Let X be a dataset with dichotomous outcome, $y = \{0, 1\}$. For each training sample x_i in X, the outcome is either $y_i = 1$ or $y_i = 0$. The experiments outcome with $y_i = 1$ are said to have 'Good' link quality, while for $y_i = 0$ for 'Bad' quality.

In supervised-learning, to make sure the input dataset is learnable, a differentiable function is needed to do the fitting instead of using two line segments. The probability that $y = 1$, given x, parameterized by θ or often referred to as logistic regression hypothesis is defined as:

$$p(y=1 \mid x;\theta) = h_\theta(x) = g(\theta^T x) \tag{1}$$

where function g is the sigmoid function and $h_\theta(x)$ is interpreted as the estimated probability that $y = 1$ on input x. Before executing the actual cost function, a sigmoid function is employed. The sigmoid function is defined as:

$$g(z) = \frac{1}{1+e^{-z}} \tag{2}$$

The training set $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$ with m samples where $x \in [x_1, x_2]^T$ is the set of input features (available bandwidth and latency) used to obtain the fitting parameter, θ to minimize the cost function $J(\theta)$ as follows [19]:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \tag{3}$$

In order to optimize the algorithm, gradient descent is applied iteratively as follows:

<u>Optimization algorithm:</u>

```
i.     Compute cost function $J(\theta)$.
ii.    To $min_\theta J(\theta)$, perform gradient descent:
iii.   Repeat {
iv.        $\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta)$   (for $j$ = 0, 1, .... n)
v.     }
```

The gradient of the cost function is calculated iteratively to achieve convergence to global minimum. The gradient of the cost is defined as follows:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\textstyle\sum_{i=1}^{m} h_\theta((x^{(i)}) - y^{(i)})\, x_j^{(i)} \tag{4}$$

The trained dataset will output a model with decision boundary and trained parameter for new input samples. For the non-linearly separable dataset, features are mapped into higher dimension with higher polynomial terms of x_1 and x_2 to fit it, and regularization term is use to do parameter tuning, θ. The polynomial is expanded up to the sixth power which is best fit in this case. The learning problem can be difficult if the dimension is too high.

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \ldots) \tag{5}$$

Features x_1 and x_2 correspond to the available bandwidth and latency on certain link in the network. The cost function with regularization term is as follows [19]:

$$J(\theta) = -[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{m}\sum_{j=1}^{n}\theta_j^2 \tag{6}$$

This allows us to build a more expressive classifier which is not susceptible to under-fitting or over-fitting. Gradient descent is applied for optimization as well. The gradient of the cost is defined as follows:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j \tag{7}$$

for $j = 1, 2, 3, \ldots, n$.

B. Multiclass Logistic Regression (Multiclass LR)

To enable better classification, we implemented four classes by applying "one-against-all" multiclass LR. The hypothesis of multiclass LR is different from that of binary LR while the cost function is similar in both cases. The hypothesis of *N*-class multiclass logistic regression is:

$$h_\theta^i(x) = p(y = i \mid x; \theta) \quad (\text{for } i = 1, 2, \ldots, N) \tag{8}$$

where a logistic regression classifier $h_\theta^i(x)$ is trained for each class *i* to predict the probability that $y = i$. In "one-against-all" multiclass LR, for a new input *x*, prediction is made by picking the class *i* that maximizes $max_i\ h_\theta^i(x)$.

IV. SUPPORT VECTOR MACHINE

SVM is suitable for classification problems with high dimensional feature space and small training set size [20]. There are four common kernels in SVM: linear, polynomial, radial basis function (RBF), and sigmoid. In this paper, we focus on C-support vector classification with linear and RBF kernel to study if the distribution is linearly separable or non-linearly separable.

A. *C-Support Vector Classification (C-SVC)*

The main parameter in C-SVC is *C*, the balance parameter, which plays the role similar to $1/\lambda$, regularization parameter in RLR. The training features with *l* samples are interpreted as vectors in C-SVC, such that $x_i \in R^n, i = 1,\ldots,l$, and the label vector $y \in R^l$ such that $y_i \in \{0,1\}$, as introduced in [21] to solve the following primal optimization problem.

$$\begin{aligned} &\min_{\omega,b,\xi} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l}\xi_i \\ \text{subject to}\quad & y_i(\omega^T\phi(x_i) + b) \ge 1 - \xi_i, \\ & \xi i \ge 0, i = 1,\ldots,l, \end{aligned} \tag{9}$$

where the kernel function, $\phi(x_i)$, maps x_i into a higher-dimensional space, $C > 0$ is the regularization parameter, *b* is a bias, ω is the feature vector and $\sum_{i=1}^{l}\zeta_i$ is the sum of errors in addition to $\omega^T\omega$. Due to the possible high dimensionality of vector variable ω, usually the dual problem is solved as follows:

$$\begin{aligned} &\min_{\alpha} \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \\ \text{subject to}\quad & y^T\alpha = 0, \\ & 0 \le \alpha i \le C, i = 1,\ldots,l, \end{aligned} \tag{10}$$

where $e = [1,\ldots,l]^T$ is the vector of all ones, *Q* is an 1 by *l* positive semi-definite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is the kernel function.

After solving the equation in (10), by using the primal-dual relationship, the optimal ω satisfies

$$\omega = \sum_{i=1}^{l} y_i\alpha_i\phi(x_i) \tag{11}$$

and the decision function is

$$\text{sgn}(\omega T\phi(x) + b) = \text{sgn}(\sum_{i=1}^{l} y_i\alpha_i K(x_i, x) + b). \tag{12}$$

Then, we store $y_i, \alpha_i \forall_i$, *b*, class label names, support vectors, and other information such as kernel parameters in the model for prediction. In this paper, we compare linear and RBF kernel to fit our problem.

Linear kernel: $K(x_i, x_j) = x_i^T x_j$ (13)

RBF kernel: $K(x_{i,} x_j) = \exp(-\gamma \| x_i - x_j \|^2, \gamma > 0$ (14)

The balance parameter *C* and kernel parameters are then chosen according to the selected kernel. RBF kernel is able to map samples into higher dimensional space with non-linear attributes, while linear kernel is able to handle linear relation between class labels and attributes only.

TABLE I. BANDWIDTH REQUIREMENT FOR SKYPE[22] AND TRUECONF[23]

Application	Call type	Minimum download speed (kbps)	Minimum upload speed (kbps)
Skype	Calling	30	30
	Video calling/scree sharing	128	128
	Video calling (high-quality)	400	400
	Video calling (HD)	1200	1200
	Group video (3 people)	512	128
TrueConf	Video Call HD	128	128
	Video lecture (speaker)	1920	128
	Video lecture (listener)	128	128
	Virtual meeting (speaker)	384	128
	Virtual meeting (listener)	512	-

TABLE II. THRESHOLD SETTINGS FOR BINARY AND MULTICLASS CLASSIFICATION

Classification	Link Quality	Available bandwidth	Latency	Suggested call type
Binary	Good	$\geq$ 1Mbps	$\leq$ 0.05s	Video calling (high-quality), conferencing
	Bad	< 1Mbps	> 0.05s	Video calling (2 people)
Multiclass	Very Good	$\geq$ 3Mbps	$\leq$ 0.03s	Video calling (HD), conferencing
	Good	$\geq$ 1Mbps	$\leq$ 0.05s	Video calling (high-quality), conferencing
	Moderate	$\geq$ 100kbps	$\leq$ 0.5s	Video calling (2 people)
	Bad	< 100kbps	> 0.5s	Calling (audio)

V. EVALUATION METHODOLOGY

The aim of a learning algorithm is to train a set of training samples to obtain the best fitting parameter. For both classification methods, we evaluate the accuracy as:

$$Accuracy = \frac{P_{correct}}{m} \times 100\% \qquad (15)$$

where $P_{correct}$ is the number of correctly predicted data, m is the total number of data. If a trained model has high accuracy on the cross-validated training dataset, then it is assumed to fit to all other new samples, and can be used to classify new samples. The datasets used are from the *bedibe* project [23] and is divided in the ratio of 600:200:200 for training set, cross-validation set and test set.

A. Cross-validation and Grid Search

There is no tuning parameter in LR, while the only parameter λ, regularization parameter in regularized LR, is trained with different values to obtain the best fitting parameter for the algorithm. The learning model is generated from the training set, and the model is validated with cross-validation dataset.

The regularization parameter, C in C-SVC with linear kernel is experimented in the same way, by generating learning model with different parameter. Using linear kernel, large value of C will tend to minimize misclassification which smaller C values would maximize the margin between boundaries.

For C-SVC with RBF kernel, there are two parameters: C and γ. The selection of parameters for C-SVC with RBF kernel is performed through grid search. The different pairs of (C, γ) values are tested and the one with the best cross-validation accuracy is picked. We used 5-fold cross-validation in the experiments to prevent over-fitting problem. We have deployed LIBSVM [22] to run our experiment C-SVC.

B. Classification Threshold

The classification threshold settings are based on the applications requirement, such as bandwidth requirement for Skype [24] and TrueConf [25] as shown in Table I. The incoming bandwidth requirement for video lecture (speaker) in TrueConf is higher to support the continuous media streaming. Interactive communications have stringent requirement in delay. The acceptable values for one-way delay are within 150 to 400ms [26]. Therefore, link with latency larger than 500ms with 5Mbps available bandwidth is predicted as bad. Though with high available bandwidth, the video image received is good but the high latency is causing the audio to be out-of-sync.

In order to secure maximum performance, we have added 50% safety margin to the network performance metrics. With video conferencing being the key interest in our research, the threshold is set accordingly as in Table II. For example, with the 'Very Good' quality, we are able to start a group video with three parties while if the quality is 'Bad', only audio calling is preferable.

VI. RESULTS AND DISCUSSIONS

This section shows experimental results in terms of accuracy for both LR and SVM respectively.

C. Experimental Results for LR

The experimental results for binary classification through LR are shown in Table III.

TABLE III. BINARY CLASSIFICATION BY LOGISTIC REGRESSION

	Training Set	**Cross Validation Set**	**Test Set**
Accuracy (%)	98.17	99.00	99.50

The binary classification obtained through LR presents high accuracy as shown in Table III, with 98.17% of training data are correctly classified. The training set data is plotted as shown in Fig. 1 to illustrate the binary link quality classification. The two axes in Fig. 1 are the corresponding performance metrics (available bandwidth and latency), which act as the input features to the learning algorithm.

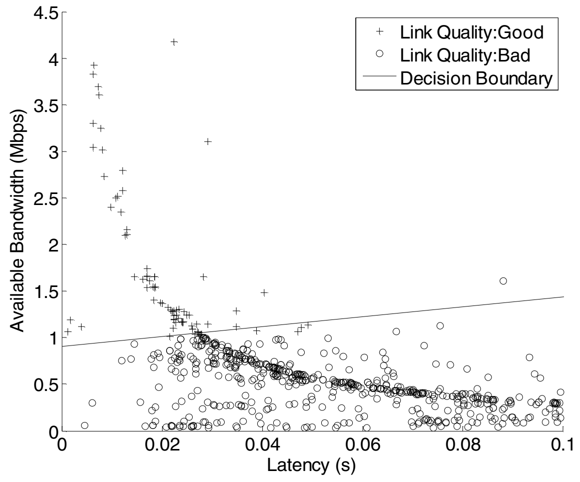


Fig. 1. Binary classification by LR

For regularized logistic regression, we have tested it with different value of regularization coefficient, λ and obtain the respective accuracy as in Table IV.

TABLE IV. BINARY CLASSIFICATION BY RLR EXPERIMENTED WITH DIFFERENT λ.

Regularization coefficient, λ	**Accuracy (%)**
	Training Set
1	97.33
10	95.33
100	92.67

The experiment where $\lambda = 1$ yields the highest accuracy as shown in Table IV. The regularization parameter is included in training set to prevent over-fitting of the algorithm. Hence, the experimental results show training accuracy for RLR is slightly lower than normal LR for binary classification.

TABLE V. BINARY CLASSIFICATION BY RLR

	Training Set	**Cross Validation Set**	**Test Set**
Accuracy (%)	97.33	96.50	98.00

We further cross validated and tested the accuracy with the trained parameter from training set where $\lambda = 1$ and the results are shown in Table V. The decision boundary generated for RLR in Fig. 2 produces a straight line from the training model.

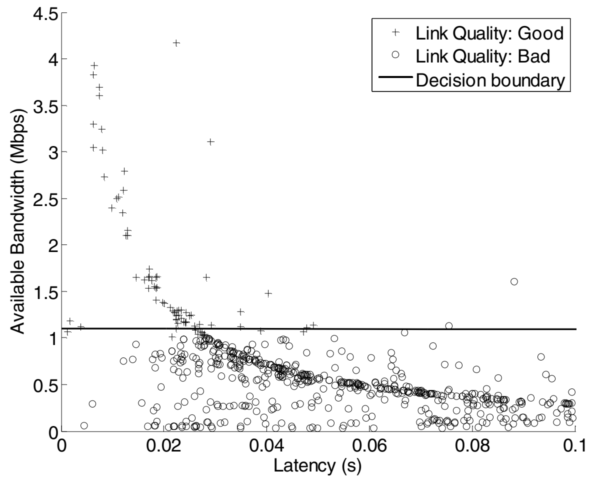


Fig. 2. Classification by RLR with λ=1.

We have implemented the multiclass regularized logistic regression through "one-against-all" method. The experimental results for multiclass classification by RLR as presented in Table VI. The illustration of multiclass classification by RLR is plotted in Fig. 3.

TABLE VI. MULTICLASS CLASSIFICATION BY ONE-AGAINST-ALL RLR EXPERIMENTED WITH DIFFERENT λ.

Regularization coefficient, λ	**Accuracy (%)**
	Training Set
1	78.33
10	78.16
100	76.50

The results in Table VI show highest accuracy is achieved when $\lambda = 1$ for training data set. The training model is cross validated and tested as shown in Table VII.

TABLE VII. MULTICLASS CLASSICIATION BY RLR

	Training Set	**Cross Validation Set**	**Test Set**
Accuracy (%)	78.33	78.00	73.50

The accuracy for multiclass RLR as shown in Table VII is lower, with around 78% of correctly predicted samples.

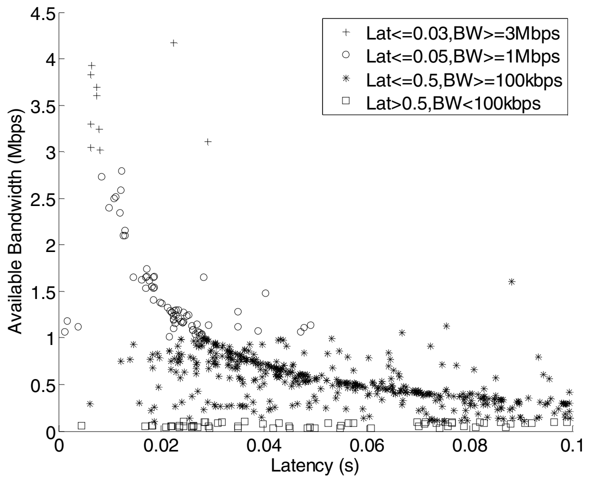


Fig. 3. Multiclass classification by one-against-all RLR with λ=1.

D. Experimental Results (C-SVC with Linear Kernel)

We have experimented with different values of the balance parameter, *C* of C-SVC as shown in Table VIII.

Table VIII. C-SVC WITH LINEAR KERNEL EXPERIEMENTED WITH DIFFERENT VALUES OF *C*

log2c	Training Set Accuracy (%)
-10	86.67
0	97.50
5	99.00
10	99.50
15	99.67
20	99.67

The experimental results show that $C=2^{20}$ is the best fit for the classification. It is known that large value of *C* will lead to over-fitting of the algorithm for smaller data size. To prevent this, the parameter *C* is set to 2^5 and applied in cross-validation set and test set.

Table VIIII. EXPERIMENTAL RESULTS FOR C-SVC WITH LINEAR KERNEL

	Training Set	Cross-validation Set	Test Set
Accuracy (%) (Binary)	99.00	99.50	98.50
Accuracy (%) (Multiclass)	98.83	98.00	99.00

From Table VIIII, C-SVC is able to achieve high accuracy in binary classification compared to multiclass classification. We have further cross-validated the respective training model obtained, which achieved 98% of accuracy.

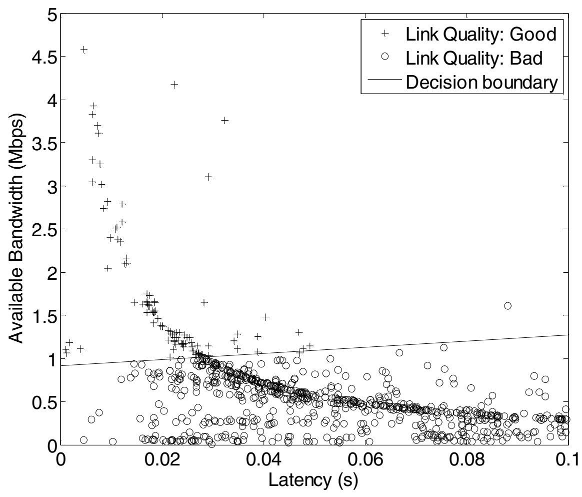


Fig. 4. Decision boundary plotted for C-SVC with linear kernel.

The decision boundary for C-SVC with linear kernel is plotted as in Fig. 4. It is shown that in Fig 4 the training data is not linearly separable, thus we applied kernel function (RBF kernel) to divide the data in a higher dimensional space in the following section.

E. Experimental Results (C-SVC with RBF kernel)

Since there are two important parameters (*C*, *γ*) in C-SVC with RBF kernel, we implemented grid search to find the best suited parameter and the result is shown in contour plot in Fig. 5 and Fig. 6.

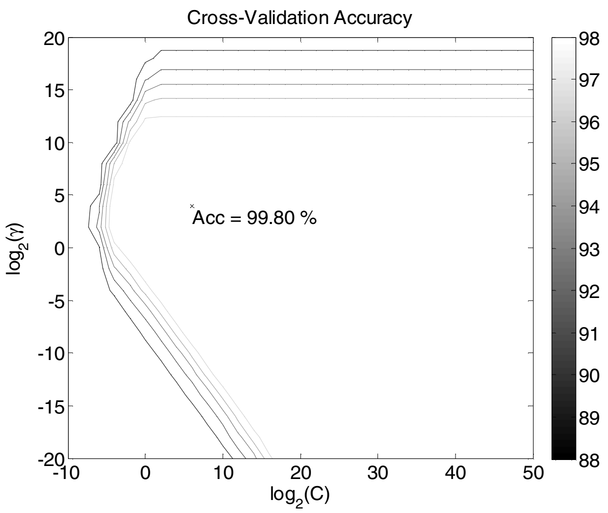


Fig. 5 Contour of cross-validation accuracy via grid-search for binary classification with SVM.

The best cross-validation accuracy is achieved at $C = 2^6$ and $\gamma = 2^4$ for binary classification while $C = 2^{12}$ and $\gamma = 2^2$ for multiclass classification. Then, the parameters (*C*, *γ*) will be used to train a model and test it again with test set.

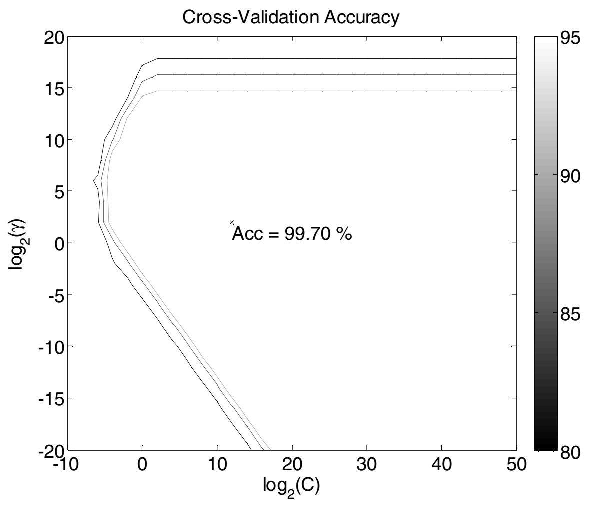


Fig. 6 Contour of cross-validation accuracy via grid-search for multiclass classification with SVM.

Table X. EXPERIMENTAL RESULTS FOR C-SVC WITH RBF KERNEL

	Training Set	**Cross-validation Set**	**Test Set**
Accuracy (%) (Binary)	99.17	99.80	99.50
Accuracy (%) (Multiclass: one-against-all)	99.83	99.70	100.00

C-SVC with RBF kernel is able to achieve high accuracy for both binary and multiclass classification, as shown in Table X. The accuracy for test set in multiclass classification gives 100% accuracy with all the sample links are correctly classified. This can be over-fit as well since the test set is small but C is set to be very large.

F. Discussions

We have implemented both binary and multiclass classification for logistic regression with the results shown in Section VI. The training parameter obtained from the training samples produces high accuracy for both the test set and the cross-validation set in binary classification. To further verify that the prediction algorithm does not over-fit, regularization coefficient, λ is included. We have tested it with different values of λ. As shown in Table IV, the accuracy is best achievable at $\lambda = 1$. While for multiclass classification with RLR, the accuracy is not comparable with C-SVC, with highest achievable accuracy of 78.33%.

For binary classification with C-SVC linear kernel, the regularization parameter, C, is experimented with different values, and shown high accuracy when C=2^{20} in Table VIII. In order to prevent algorithm from over-fitting, C is set to 2^5. While for RBF kernel, we have performed the parameter selection through grid search, which provides high accuracy on training, cross-validation and test set as shown in Table X. It is observed that test set in multiclass classification achieved 100% accuracy. This is due to the large C trained from the training data over-fit the algorithm with smaller samples. C-SVC with RBF kernel gives higher accuracy compared to C-SVC with linear kernel. From the results obtained, we are able to prove that SVM outperforms logistic regression in both binary (with slightly higher accuracy) and multiclass classification (with average of 20% higher accuracy).

The training parameters obtained in the experiments are not generalizable as it depends highly on the nature of the dataset, including the network environment and number of network nodes.

G. LR vs. SVMs

The difference between LR and SVMs is that LR predicts a link's quality based on probabilistic function, while SVMs are statistical learning theory of finding a predictive function based on training data. LR is a regression analysis which maximizes the likelihood of data iteratively. SVM generates a model function and directly maximize the accuracy [5]. Kernelized SVM works better than linear SVM for non-linear separable data. LR does not require tuning parameter, except the regularization parameter in RLR to prevent over-fitting, but it is unable to achieve comparable accuracy in multiclass classification. While SVM is able to achieve high accuracy with small training dataset size and outperform LR especially in multiclass classification.

The metrics pair is constructed via our previous prediction algorithm [18], in which prediction is performed in a large scale network by measuring to a few and predicts the rest. This provide us with *O(nxn)* look-up table with *n* number of network links rather than depend solely on one single link. As network metrics varied over time, the future link condition can be predicted via the updated metrics value from the network resource prediction algorithm [18]. In real implementation, these classification approaches are integrated into the video conferencing system to suggest the supported call type to user based on the link quality.

VII. CONCLUSIONS

Knowledge of network performance metrics such as available bandwidth and latency are important for the scalability and Quality of Service in multimedia streaming. Existing prediction approaches involve only either available bandwidth or latency with real values. By taking available bandwidth and latency into consideration for link quality prediction, we are able to predict the quality of the link and suggest supported conference call type to the users. In this paper, we have implemented two major classification approaches, i.e. logistic regression and SVM to perform the classification task. Logistic regression is easier to be implemented with only one parameter, the regularization coefficient. SVM is able to give high accuracy even with small training dataset. Furthermore, it is also suitable for large classification problem since it is able to classify the data in higher dimensions with the use of kernel function. Through our experiments, SVM outperforms logistic regression in both binary and multiclass classification. In the future, we plan to experiment classification through SVM in practical use for multimedia streaming and also the online learning (real-time learning).

ACKNOWLEDGMENT

We would like to thank Boon Ping for her advice on the project and also the developers of LIBSVM for their useful toolkit and hard work.

REFERENCES

[1] B. Ratner, "Logistic Regression: The Workhorse of Response Modeling," in *Statistical and machine learning data mining: Techniques for better predictive modeling and analysis of big data*, 2012, ch. 8.

[2] M. W. Chang, W. T. Yih, C. Meek, "Partitioned logistic regression for spam filtering," in Proc. Of the 14th ACM SIGKDD Int. conf. on Knowledge discovery and data mining , pp. 97-105, 2008.

[3] M. Chae, S. Shim, H. Cho, B. Lee, "An empirical analysis of fraud detection in online auctions: credit card phantom transaction," in Proc. of the 40th Hawaii Int. Conf. on System Science, 2007.

[4] D. Timmerman et. al, "Logistic regression model to distinguish between benign and malignant adnexal mass before surgery: A multicenter study by the Int. ovarian tumor analysis group," in J. of Clinical Oncology, 2005.

[5] C. Campbell and Y. Ying, "Learning with support vector machines," in *Synthesis lectures on artificial intelligence and machine learning*, 2011, ch. *1*, pp. *1-14*.

[6] A. Este, F. Gringoli, L. Salgarelli, "Support vector machines for TCP traffic classification," in Int. J. of Comp. and Telecommun. Networking 2009.

[7] T. Simon, and D. Koller, "Support vector machine active learning with applications to text classification," in J. of Machine Learning Research, Stanford Univ, USA, 45-66, 2001.

[8] T. S. Furey, N. Cristianini, N. Duffy, et al., "Support vector machine classification and validation of cancer tissue samples using microarray expression data," in BIOINFORMATICS, USA, vol. 16, pp. 906-914, May 2000.

[9] F. Dabek, R. Cox, R. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in SIGCOMM'04, USA, August 2004.

[10] T. S. Eugene Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," in Proc. of IEEE Infocom'02, New York, June 2002.

[11] Y. Mao, L. Saul, and J. M. Smith, "IDES: An internet distance service for large networks," in IEEE J. on Selected Areas in Comm., pp. 2273-2284, December 2006.

[12] O. Beaumont, L. Eyraud-Dubois, and Y. J. Won, "Using the last-mile model as a distributed scheme for available bandwidth prediction," in EuroPar 2011, France, April 2011.

[13] V. Ramasubramanian, D. Malkhi, F. Kuhn, M. Balakrishnan, et al., "On the treeness of internet latency and bandwidth," in SIGMETRICS/Performance'09, USA, June 2009.

[14] S. Schober, S. Brenner, R. Kapitza, et al., "Bandwidth prediction in the face of asymmetry," in Proc. 13th Int. IFIP Conf. on Distrib. Appl. And Interop. Sys., January 2013.

[15] Y. J. Liao, P. Geurts and G. Leduc, "Network distance prediction based on decentralized matrix factorization," in Proc. of IFIP Networking 2010, India , May 2010.

[16] Y. Liao, W. Du, P. Geurts, and G. Leduc, "DMFSGD: A decentralized matrix factorization algorithm for network distance prediction," in IEEE/ACM Trans. on Netw., 2012.

[17] W. Du, Y. Liao, P. Geurts and G. Leduc, "Ordinal rating of network performance and inference by matrix completion," Available: http://arxiv.org/abs/1211.00447.

[18] S. J. Lim, B. P. Lim, S. Lau, et al., "Implementing stochastic gradient descent based on historical network distance for available bandwidth prediction," in 2nd Int. Conf. on e-Technologies and Network for Development, 2013.

[19] P. Komarek, "Logistic regression for data mining and high-dimensional classification," M.S. thesis, Dept. of Math Sciences, Camegie Mellon Univ.

[20] V. N. Vapnik, "An overview of statistical learning theory," in IEEE Transaction on Neural Networking, September 1999.

[21] B. E. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in Proc. of the 5th Annual Workshop on Computational Learning Theory, ACM Press, pp. 144-152, 1992.

[22] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines," in ACM Trans. on Intelligent Systems and Technology, 2001.

[23] L.Eyraud-Dubois and P. Uznanski, "Bedibe: Datasets and Software Tools for Distributed Bandwidth Prediction,", in *AlgoTel*, 2012.

[24] Skype homepage. [Online]. Available: https://support.skype.com/en/faq/FA1417/how-much-bandwidth-does-skype-need

[25] TrueConf homepage. [Online]. Available: http://trueconf.com/support/communication-channels.html

[26] G. Karlson, "Asynchronous transfer of video," in *IEEE Communications Magazine*, August 1996, pp. 118-126.