Efficient Algebra for Extreme Learning Machines

A PROJECT REPORT

Submitted in partial fulfillment of the

requirements for the award of the degrees

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIECE AND ENGINEERING

Submitted by:

Dhruva Ahuja & Pravin Kumar

Guided by:

Dr. Kapil Ahuja

INDIAN INSTITUTE OF TECHNOLOGY INDORE

Nov 28, 2016

CANDIDATE’S DECLARATION

We hereby declare that the project entitled **“Efficient Algebra for Extreme Learning Machines”** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science and Engineering, completed under the supervision of **Dr. Kapil Ahuja (Assistant Professor, Computer Science and Engineering), Mr. Chandan Gautam (Ph.D. Scholar in Computer Science and Engineering)** IIT Indore is an authentic work.

Further, we declare that we have not submitted this work for the award of any other degree elsewhere.

Dhruva Ahuja
130001010
Discipline of Computer Science and Engineering
Indian Institute of Technology, Indore

Pravin Parmar
130002026
Discipline of Electrical Engineering
Indian Institute of Technology, Indore

CERTIFICATE by BTP Guide

It is certified that the above statement made by the students is correct to the best of my knowledge.

Dr. Kapil Ahuja,
Assistant Professor,
Discipline of Computer Science & Engineering,
Indian Institute of Technology Indore
(Project Guide)

Preface

This report on Efficient Algebra for Extreme Learning machines is prepared under the guidance of Dr. Kapil Ahuja and Mr. Chandan Gautam.

Through this report, we have tried to give a detailed formulation for doing Multi-Task One-class classification using Extreme Learning machines and shown the evidences of its importance over one class classification. We have tried to the best of our abilities and knowledge to explain the content in a lucid manner. We have also added graphs and figures to make it more illustrative.

Dhruva Ahuja

B.Tech. IV Year

Discipline of Computer Science and Engineering

IIT Indore

Pravin Kumar

B.Tech. IV Year

Discipline of Electrical Engineering

IIT Indore

Acknowledgements

We wish to thank Dr. Kapil Ahuja for his kind support and valuable guidance and a special thanks to Mr. Chandan Gautam for being a very helpful and always available mentor.

It is their help and support, due to which we became able to complete the design and technical report.

Without their support this report would not have been possible.

Dhruva Ahuja

B.Tech. IV Year

Discipline of Computer Science and Engineering

IIT Indore

Pravin Kumar

B.Tech. IV Year

Discipline of Electrical Engineering

IIT Indore

Abstract

The machines learning is a type of artificial intelligence which provides computer an ability to learn without being explicitly programmed. Classification, the important aspect of machine learning can be of different types- One-class Classification and Multi-class classification. In our project, we are concentrating upon multi-task one-class classification problems. Taking in account that for some one-class classification problems the training data are very limited so we take advantage of Multi-Task Learning (MTL) in which training of related tasks is done together and the One-Class MTL gives better generalizing capability. One-class classification can be done with the help of different methods e.g. SVM (Support Vector Machines), Extreme Learning machines etc. Multi-task learning is an effective way to improve the generalization performance by training multiple data.

Extreme learning machine (ELM) is efficient tool for solving classification problem. This has been proved to be a very efficient and fast method for doing one class classification. However, while classifying multiple related tasks in which only few data per task are available ELM hardly gives spectacular result. To overcome this problem, we extended ELM to Multi-task learning (MTL). First, based on assumption that Multiple related tasks share some similarity within the patterns of the samples, a new MTL algorithm for ELM is proposed to learn related tasks jointly via matrix inversion (pseudo matrix inversion) and coming up with a single decision function which is able to classify the test samples from all related tasks. So, finally an efficient algebra has been proposed to tackle multiple task learning.

Keywords: Support Vector Machines, Extreme learning machine, multitask learning.

Table of Contents

Chapter 1

Introduction

Multi-task learning (MTL, also known as *inductive transfer* or *learning to learn*) has become a research topic of renewed interest in machine learning. One main insight of multi-task learning techniques is that related tasks share similar structure and information which may be useful for improving the performance of these tasks.

In past decade, extreme learning machine (ELM), introduced by Huang [2], has played an important role in one class classification. Much research has been done in one class classification using ELM and SVM or simple neural network like backpropagation based classification. Unlike the slow gradient-based learning algorithms used to train the neural networks in which all the parameters of the networks are tuned iteratively, ELM randomly chooses the weights between input nodes hidden nodes and analytically determines the output weights. That’s why this algorithm tends to give good generalization performance at extremely fast learning speed. In most cases and can learn hundred times faster than conventional popular learning algorithms for feedforward neural networks.

The one class classification problems are the ones in which the training samples related to just one class (usually novelty) are available. This has many real-life applications; take the case of online shopping service, in order to recommend the goods user wants you need to track the history of the items which the user has bought earlier (positive training samples), While the collection of the negative test sample is very challenging. Multi-task one class classification problems are approached by variants of Support Vector Machine (ν-SVM). It is very common to tackle multiple related learning tasks in many real-life applications. In this case, traditional ELM is incapable of handling these tasks in effective way. Specially, if training data per task are deficient while the dimension is low, ELM tends to give an unsatisfactory result due insufficient domain knowledge. Many empirical works have shown that the learning performance can be improved with the help of other related tasks, which is called multi-task learning (MTL). And some supervised learning approaches like support vector machine and Backpropagation neural network have been extended [2][3] to solve MTL problems. Considering the higher speed and simpler architecture of ELM than the traditional learning approaches above, it is natural to extend the classical ELM to multi-task learning case for better generalization performance. In multi-task learning as specified above, multiple related tasks, which have close enough, that is, their model or their model parameters are close to certain mean function. For multi-task ELM, there is a single decision function which is able to classify the test samples from multiple related tasks.

Chapter 2

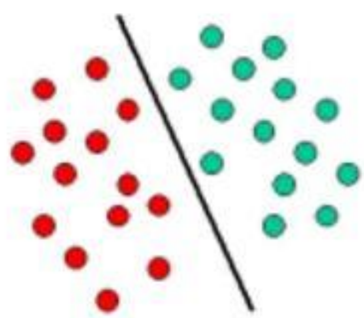
Literature Survey

In the real-life scenario while doing one class classification, sometimes number of training samples are very limited, in those cases the independent learning methods yield very poor generalization. So in that case we utilize the advantages of the several related tasks for learning and providing a better generalization capability for the machines.

To do Multi-Task Learning in one class classification, several frameworks have been proposed using the variant of Support Vector Machines (SVM) which is v-SVM. We will go briefly through the SVM, it's idea and will see approaches used to make it work for multiple tasks.

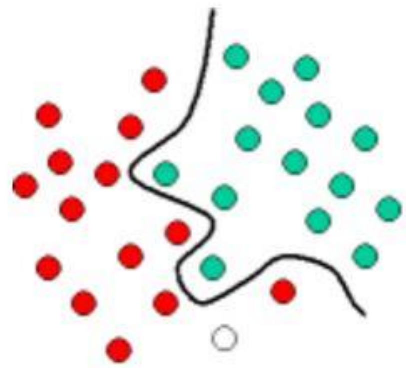
Support Vector Machines – is a supervised machine learning algorithm which can be used for both classification or regression problems. In this we search for a decision boundary which classifies the given data into separate classes efficiently. In this algorithm, we plot each data item as a point in n-dimensional space (n is feature space) with value of each feature being value of particular coordinate and classification is done finding that effective hyperplane (decision boundary).

This is the example of simple linearly separable two class classification with the effective boundary –



Red and green color dots represent data from two different classes and are separated black colored boundary.

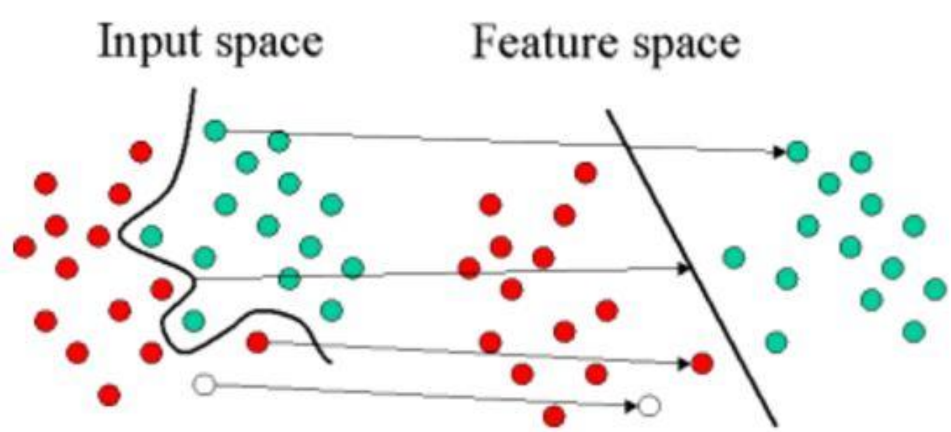
However, this was a very simple basic example but for most of the real-life examples decision boundary is not very simple. Like, take the example below-



This is an example of nonlinear classification, so we have to use some trick for such kind of classification which is inspired from Cover's Theorem.

<u>Cover's Theorem</u> – States that given a set of training data that is not linearly separable, one can with high probability transform it into a training set that is linearly separable by projecting it into a higher-dimensional space via some non-linear transformation. [Wikipedia]

So, to make the problem linearly separable, idea is to map the original objects into a feature space of higher dimensions using some mathematical functions, Kernels. After making them linearly separable, the decision boundary can be easily found.



(Picture taken from http://www.statsoft.com/)

<u>SVM for one-class classification –</u> There are two variants of SVM which are used for one class classification named SVDD, ν-SVM.

- Support Vector Data Description (SVDD): Since the data corresponding to just one class will be available, so the idea is to put all the training samples into a hypersphere of a minimum radius R. So, we have to search for a spherical boundary which efficiently encloses all the novelty samples. If the upcoming test samples fits in that sphere, then it is classified as positive sample otherwise negative. SVDD can be formulated as a minimization problem-

$$\begin{aligned} \min \quad & L_{P_{\text{SVDD}}} = R^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & \|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2 + \xi_i, \quad i = 1, \ldots, N \\ & \xi_i \geq 0, \quad \forall i, \end{aligned}$$

R is the radius of the hypersphere; a is the center of the sphere & C is trade off parameter.

After solving this optimization problem R and a can be found and for the upcoming sample it can be easily determined whether the sample is outlier or novelty.

- v-Support Vector Machine (v-SVM): This is similar to SVDD but it has the advantage of using v for controlling the number of support vectors. The value of v is bounded by 0 and 1. This additional bound is beneficial in implementation.

v-SVM can be formulated as -

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\rho} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu N}\sum_{i=1}^{N}\xi_i - \rho \\ \text{s.t.} \quad & \mathbf{w}^\top\phi(\mathbf{x}_i) \geq \rho - \xi_i, \quad i = 1, \ldots, N, \\ & \mathbf{w} \in \mathbb{R}^f, \quad \boldsymbol{\xi} \in \mathbb{R}_+^N, \quad \rho \in \mathbb{R} \end{aligned}$$

N is the number of training samples

And the final decision boundary is

$$f(\mathbf{x}) = \text{sign} \left(\sum_{i=1}^{N} \alpha_i \mathbf{K}(\mathbf{x}_i, \mathbf{x}) - \rho \right)$$

Multi-Task Learning (MTL) – Until now we have discussed about one-task one class classification. But in many real life scenarios we have a very few number of samples corresponding to a single task and we have other related tasks which share similar basic structures, so in those cases the choice of multi-task becomes more relevant.

Some examples of multiple tasks- Cancer cell detection, Surveillance Systems for fault detection in different machines.

For the testing purposes, Multi-task samples can be simulated by adding noise in the simple data like:

(Taken from- **Multi-Task Learning for One-class Classification** - Haiqin Yang, and Irwin King, *Senior Member, IEEE*, and Michael R. Lyu, *Fellow, IEEE*)

Chapter 3

Proposed Work

3.1 Past Work:

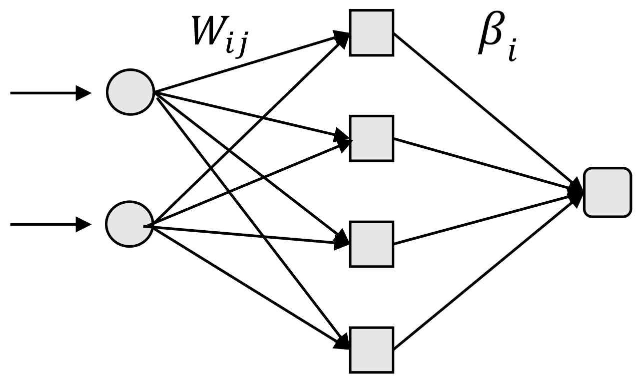
Multitask version of ν-SVM- In this case there are more than one related tasks, ν-SVM can be extended from one-task to multitask under some assumptions.

There are T tasks and N number of total training samples and samples of the tasks are considered to be mutually exclusive. So, formulation of the problem is (as given in Multi-task learning with one-class SVM Xiyan He, Gilles Mourot, Didier Maquin, José Ragot, Pierre Beauseroy, André Smolarz, EdithGrall-Maës)

$$\begin{aligned}
\min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\rho},\eta} \quad & \frac{1}{2T}\sum_{t=1}^{T}\|\mathbf{w}_t\|^2 + \frac{1}{N}\sum_{t=1}^{T}\frac{1}{\nu_t}\sum_{i\in\mathcal{T}_t}\xi_i - \sum_{t=1}^{T}\rho_t + C_\eta\eta\,, \\
\text{s.t.} \quad & \mathbf{w}_t^\top\phi(\mathbf{x}_i) \geq \rho_t - \xi_i, \quad \forall\ i\in\mathcal{T}_t, \quad t=1,\ldots,T, \\
& \frac{1}{2}\|\mathbf{w}_{i_m} - \mathbf{w}_{j_m}\|^2 \leq \eta, \quad \forall\ \ (i_m, j_m)\in\mathcal{E}, \\
& \mathbf{w}\in\mathbb{R}^{f\times T}, \ \ \boldsymbol{\xi}\in\mathbb{R}_+^N, \ \ \boldsymbol{\rho}\in\mathbb{R}^T, \ \ \eta\in\mathbb{R}_+
\end{aligned}$$

3.2 Preliminaries:

Extreme Learning Machine is feed-forward neural network for classification or regression with single layer of hidden nodes (Wikipedia). Extreme learning machine is a learning algorithm which randomly chooses the hidden node and analytically determines the output weight of Single layer feed forward networks.



In this architecture weights W_{ij} are selected semi-randomly (with some information) and input sample is mapped into a higher dimensional feature space. βs are calculated analytically while encountering the test samples.

Lagrange multiplier method - also called Lagrangian multipliers, can be used to find the extrema of a multivariate function $f(x_1, x_2 .....x_n)$ subject to the constraint $g_1(x_1, x_2 .....x_n)$, $g_2(x_1, x_2 .....x_n)$, … where f and $g_1, g_2, ..$. are functions with continuous first partial derivatives on the open set containing the curve $g_1(x_1, x_2 .....x_n) = 0,\ g_2(x_1, x_2 .....x_n) = 0$ … and $\nabla g_i \neq 0$ at any point on the curve (where ∇ is the gradient). where f and $g_1, g_2, ..$. are functions with continuous first partial derivatives on the open set containing the curve , and $g_1(x_1, x_2 .....x_n) = 0,\ g_2(x_1, x_2 .....x_n) = 0$, … and $\nabla \boldsymbol{g} \neq \mathbf{0}$ at any point on the curve (where ∇ is the gradient). For $f(x_1, x_2 .....x_n)$ to be maximized

$$\nabla f(x_1, x_2 .....x_n) = \nabla g_1(x_1, x_2 ....x_n) + \nabla g_2(x_1, x_2 ......x_n) + ...$$

Extreme Learning Machines- As stated above in the introduction Extreme Learning Machine is a feed forward Neural Network, so unlike gradient based neural networks in which error is back propagated, this is very fast. Here, parameters are not tuned iteratively but are initialized randomly and final weights are calculated using pseudo-inverse.

Extreme Learning machines for One-Class Classification can be formulated as –

$$\min \quad L_{P_{\mathrm{ELM}}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\frac{1}{2}\sum_{i=1}^{N}\|\xi_i\|^2$$

$$\text{s.t.} \quad \mathbf{h}(\mathbf{x}_i)^T\boldsymbol{\beta} = t_i - \xi_i, \quad i = 1, \ldots, N,$$

Where

$\boldsymbol{\beta}$ is the vector of the weights between hidden and output layer; t_i is the output and ξ_i is the error.

After Solving Lagrangian dual problem and enforcing KKT decision function is:

$$f(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T\boldsymbol{\beta} = \mathbf{h}(\mathbf{x})^T\mathbf{H}^T\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}.$$

The kernel version of the decision function is written as -

$$f(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T\boldsymbol{\beta} = \mathbf{h}(\mathbf{x})^T\mathbf{H}^T\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{T}$$
$$= \begin{bmatrix} K(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ K(\mathbf{x}, \mathbf{x}_N) \end{bmatrix}^T \left(\frac{\mathbf{I}}{C} + \boldsymbol{\Omega}_{\mathrm{ELM}}\right)^{-1}\mathbf{T}$$

And the kernel matrix is defined as

$$\boldsymbol{\Omega}_{\mathrm{ELM}} = \mathbf{H}\mathbf{H}^T : \Omega_{\mathrm{ELM}i,j} = \mathbf{h}(\mathbf{x}_i) \cdot \mathbf{h}(\mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j)$$

Final decision function is:

$$d_{\text{NORM-ELM}}(\mathbf{z} \mid X, \lambda) = \left| \mathbf{h}(\mathbf{z})^T \mathbf{H}^T \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^T \right)^{-1} \mathbf{e} - 1 \right|$$

3.3 Proposed Formulation:

This classification problem can be formulated as a minimization problem in which for better generalization we have to minimize the final weights (Covers theorem), the errors vectors and the sigma of the norm of the difference of the final weight with individual task's weights (much like variance).

Suppose there are T related tasks.

So, final weight β is defined as- $\beta = \frac{1}{T}\sum_{t=1}^{T}\beta_t$ where β_t is the weight vector of the t^{th} task.

So, our objective function is-

$$\min \frac{1}{2}\|\beta\|^2 + \frac{1}{2}\sum_{t=1}^{T}\|\beta - \beta_t\|^2 + \frac{C}{T}\sum_{t=1}^{T}\|\xi_t\|^2$$

Subjected to $H_t\beta_t - t_t + \xi_t = 0$

Where H is the output at the hidden layer and $\xi_t = \left[\xi_{t_1}, \xi_{t_2} \ldots\ldots \xi_{t_{N_t}}\right]^T$ is error vector.

Since this is a nonlinear constraint optimization problem. To solve this problem, we use Lagrange multipliers method.

Formulating above problem as Lagrange's dual problem-

$$L = \frac{1}{2}\|\beta\|^2 + \frac{1}{2}\sum_{t=1}^{T}\|\beta - \beta_t\|^2 + \frac{C}{2T}\sum_{t=1}^{T}\|\xi_t\|^2 - \sum_{t=1}^{T}\alpha_t^T\left(H_t\beta_t - t_t + \xi_t\right)$$

Where $\alpha_t = \left[\alpha_{t_1}, \alpha_{t_2} ....\alpha_{t_{N_t}}\right]^T$ are Lagrange's multipliers.

Based on the Karush-Kuhn-Tucker (KKT) theorem, to get the optimal solutions of, we should have

$$\nabla_{\beta_t} L = \frac{\beta}{T} + \left(\frac{T\beta_t - \beta}{T}\right) - H_t^T \alpha_t = 0$$

$$\Rightarrow \ \beta_t - H_t^T \alpha_t = 0 \qquad (1)$$

$$\nabla_{\xi_t} L = \frac{C}{T} \xi_t - \alpha_t = 0$$

$$\Rightarrow \frac{C}{T} \xi_t = \alpha_t \qquad (2)$$

$$\nabla_t L = H_t \beta_t - t_t + \xi_t = 0 \qquad (3)$$

Putting (1), (2) in (3)

We get: $\alpha_t = \left(H_t H_t^T + \frac{T}{C} I\right)^{-1} t_t$ & (4)

$$\xi_t = \frac{T}{C} \alpha_t$$

Now $\xi = \left[\xi_1, \xi_2 ....\xi_T\right]^T$ which keeps the distances of all the tasks' samples. As we know that more value of $\left|\xi_{tj}\right|$ represents the sample is more deviant from the target class. Hence, We derive the threshold θ based upon the quantile function to reject the most deviant training samples. So, $d = \left[d_1, d_2 ....d_T\right]^T$ represents the sorted sequences of the distances where all d_t are in turn sorted in non-decreasing order.

Now, θ can be determined as

$$\theta = \ d_{floor(\mu.N)},$$ where N is the total number of samples.

Coming back,

$$\beta_t = H_t^T \left(H_t H_t^T + \frac{T}{C} I \right)^{-1} t_t$$ from

(1), (4)

So, $$\beta = \left(\frac{1}{T} \sum_{t=1}^{T} h(x) H_t^T \left(H_t H_t^T + \frac{T}{C} I \right)^{-1} \right) t_t$$

For any upcoming test sample x, the decision function will be-

$$\delta(x) = sign\left(\theta - \left(\left(\frac{1}{T} \sum_{t=1}^{T} h(x) H_t^T \left(H_t H_t^T + \frac{T}{C} I \right)^{-1} \right) t_t - Y \right) \right)$$

$$\begin{cases} 1 & x \text{ is classified as a target} \\ -1 & x \text{ is classified as a outlier} \end{cases}$$

<u>The kernel version-</u>

$$\delta(x) = sign\left(\theta - \left(\left(\frac{1}{T} \sum_{t=1}^{T} \Omega_{x,t} \left(\Omega_t + \frac{T}{C} I \right)^{-1} \right) t_t - Y \right) \right)$$

The kernel matrix is defined as-

$$\Omega_t = H_t H_t^T : \Omega_{i,j} = h_t(x_i) . h_t(x_j) = K(x_{ti}, x_{tj}),$$

$$\Omega_{x,t} = \begin{bmatrix} K\left(x, x_{t_1}\right) \\ K\left(x, x_{t_2}\right) \\ . \\ . \\ K\left(x, x_{N_t}\right) \end{bmatrix}^T$$

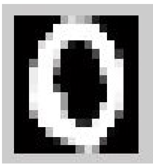
Chapter 4

Results

4.1 Dataset

USPS dataset for handwritten digit (zero) recognition data is taken, the machine is going to be trained with handwritten digit zero. For simulating the data for multiple tasks, we have added some noise in the picture which will keep the basic structure of the tasks same.

Positive training/test sample.

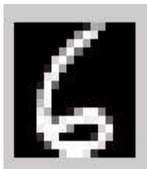


Pixel Value 1	Pixel Value 2	..	..	… …. ………	Pixel Value 256
-0.762	-1	..	0.341		-1

Positive training/test sample.

Pixel Value 1	Pixel Value 2	..	..	… …. ………	Pixel Value 256
-0.676	-0.990	..	0.294		-1

Negative training/test sample.



Pixel Value 1	Pixel Value 2	..	..	… …. ………	Pixel Value 256
-1	0.872	..	-1		-1

4.2 Observation

4.2.1 Thin Noise

Number of training samples used: 100 for each tasks

Number of novelty in test samples: 1077

Number of outliers in test samples:1200

Sample No.	**Multi-Task ELM**			**Single Task ELM**		
1	77.25 %	72.33%	**74.66%**	85.61%	56.16%	**70.09%**
2	77.71%	68.92%	**73.08%**	84.03%	60.33%	**71.54%**
3	78.74%	70.00%	**74.13%**	86.72%	54.41%	**69.69%**

4.2.2 Thick Noise

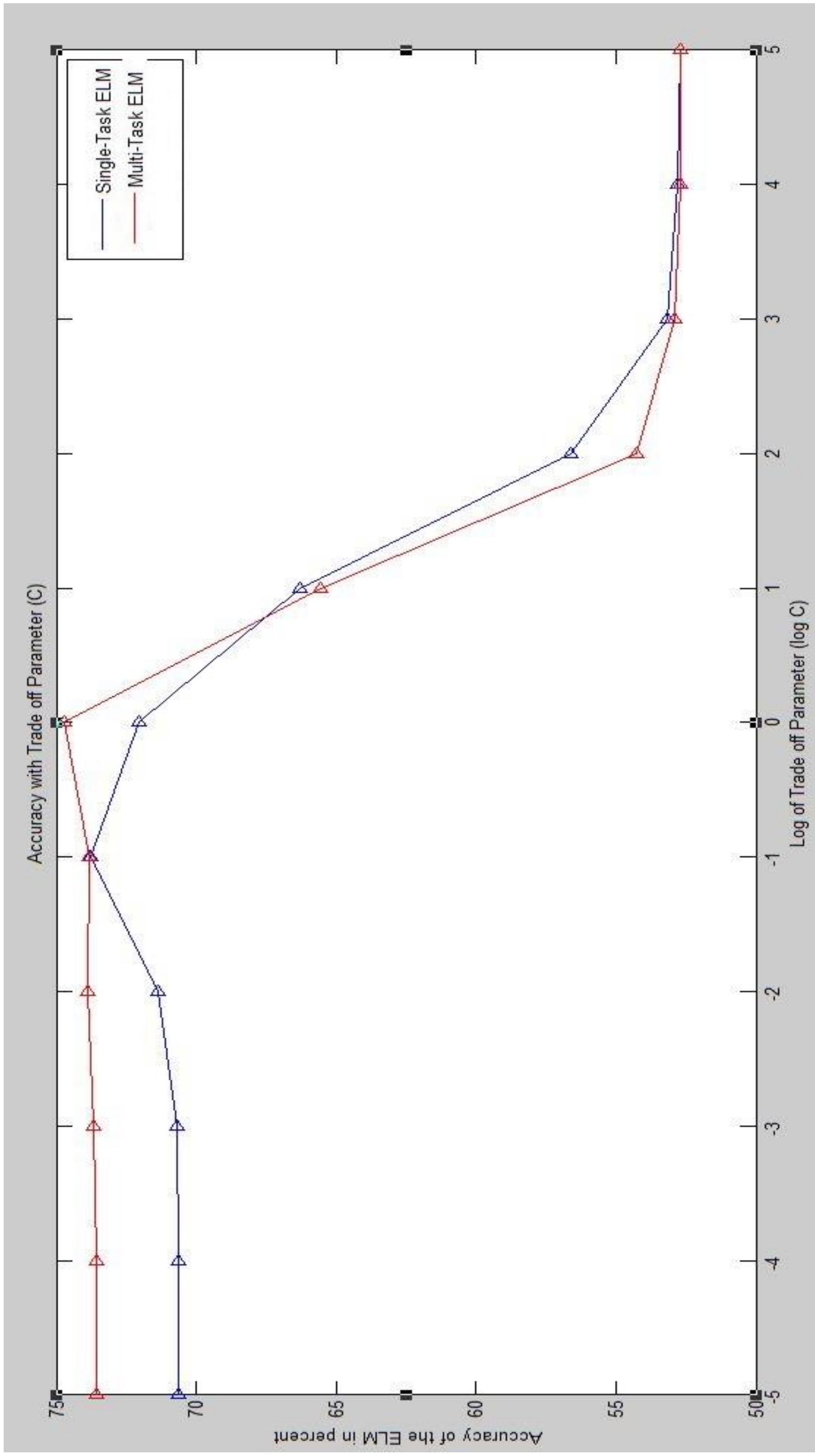
Number of training samples used: 100 for each task

Number of novelty in test samples: 1077

Number of outliers in test samples:1200

Sample No.	**Multi-Task ELM**			**Single Task ELM**		
1	71.68 %	72.83%	**72.29%**	86.25%	53.41%	**68.95%**
2	70.75%	73.00%	**71.94%**	87.74%	45.41%	**65.43%**
3	73.82%	73.83 %	**73.83%**	82.45%	61.41%	**71.36%**

Variation with Trade of Parameter (C) –



4.3 Conclusion and Future Work

- The proposed formulation for multi-task is feasible and easy to implement.
- Multi-task gives better results and generalization when compared to Single-Task ELM.
- Comparison with Multi-Task One-Class Support Vector Machine.
- For Multi-Task One Class Classification ν-SVM has also been used.
- The following are the results of **Multi-task Learning for One-class Classification using ν-SVM** by Haiqin Yang, and Irwin King, *Senior Member, IEEE,* and Michael R. Lyu, *Fellow, IEEE*

TABLE I
THE PERFORMANCE (AUC) OF EACH METHOD FOR THE USPS DATASET (%).

#	IL–SVM			One–SVM	MTL–OC		
	thin	thick	average	average	thin	thick	average
5	82.2±5.1	57.6±3.1	69.9±3.9	81.7±5.1	83.9±5.1	81.2±5.2	**82.6**±5.2
10	86.2±2.7	56.3±2.3	71.2±2.3	82.8±2.9	86.2±2.9	82.8±2.4	**84.5**±2.6
20	87.2±1.9	55.7±2.1	71.4±1.8	83.3±1.2	87.2±1.3	83.1±1.6	**85.1**±1.4
40	87.3±1.3	53.8±1.5	70.5±1.2	84.1±1.1	87.2±1.1	83.1±1.5	**85.2**±1.3

Chapter 5

References

1. Book followed: Neural-Networks-Learning-Machines-Haykin

2. G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, Extreme learning machine: A new learning scheme of feedforward neural networks, in *Proc. 2004 Int. Joint Conf. on Neural Networks* (*IJCNN'2004*), Budapest, Hungary, July 25–29, 2004, pp. 985–990.

3. Haiqin Yang, Irwin King, Michael R. Lyu – Multi-task Learning for One class Classificaiton

   http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5596881 [Last accessed by Oct. 4, 16]

4. Research Progress Seminar by Ph.D Scholar Mr. Chandan Gautam.

5. Introduction to One-class Support Vector Machines

   http://rvlasveld.github.io/blog/2013/07/12/introduction-to-one-class-support-vector-machines/ [Last accessed by Oct. 4, 16]

6. Haiqin Yang, Irwin King, Michael R. Lyu – Multi-task Learning for One class Classificaiton

   http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5596881 [Last accessed by Oct. 4, 16]

7. Xiyan He, Gilles Mourot, Didier Maquin – One Class SVM in multi-task learning

   https://www.researchgate.net/publication/216603630_One-class_SVM_in_multi-task_learning [Last accessed by Oct. 4, 16]

8. Guang-Bin Huang Qin-Yu Zhu, Chee-Kheong Siew – Extreme Learning Theory and Applications

   http://www.sciencedirect.com/science/article/pii/S0925231206000385 [Last accessed by Oct. 4, 16]

9. Qian Leng, Honggang Qi, Jun Miao, Wentao Zhu, and Guiping Su One class Classification with extreme learning machines

https://www.hindawi.com/journals/mpe/2015/412957/ [Last accessed by Oct. 4, 16]

10. Wentao Mao, Jiucheng Xu, Shengjie Zhao – Research of multitask learning based on extreme learning machines

http://www.worldscientific.com/doi/abs/10.1142/S0218488513400175 [Last accessed by Oct. 4, 16]