REFERENCES

1. Zikopoulos, P.C., et al., Understanding big data – Analytics for enterprise class Hadoop and streaming data. 2012: McGraw-Hill.
2. Suthaharan, S., Big data classification: problems and challenges in network intrusion prediction with machine learning, in Proceedings of the flagship conference of the ACM special interest group for the computer systems performance evaluation community ACM SIGMETRICS. 2013.
3. Feng, W., et al., Online classification algorithm for data streams based on fast iterative kernel principal component analysis, in Proceedings of Fifth International Conference on Natural Computation. 2009. p. 232 - 236.
4. Jaiyen, S., C. Lursinsap, and S. Phimoltares, A very fast neural learning for classification using only new incoming datum. IEEE Transactions on Neural Networks and Learning Systems, 2010. **21**(3): p. 381-392.
5. Chengjun, L. and H. Wechsler, A shape- and texture-based enhanced Fisher classifier for face recognition. IEEE Transactions on Image Processing, 2001. **10**(4): p. 598-608.
6. Haijing, W., L. Peihua, and Z. Tainwen, Histogram feature-based Fisher linear discriminant for face detection. Neural Computing and Applications, 2008. **17**(1): p. 49-58.
7. Meng, J.E., et al., Face recognition with radial basis function (RBF) neural networks. IEEE Transactions on Neural Networks and Learning Systems 2002. **13**(3): p. 697-710.
8. Yin, N.C., et al., Face recognition using nearest feature space embedding. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011. **33**(6): p. 1073-1086.
9. Michael, W.S., Learning Image components for object recognition. Journal of Machine Learning Research, 2006. **7**: p. 793-815.
10. Sahambi, H.S. and K. Khorasani, A neural-network appearance-based 3-d object recognition using independent component analysis. IEEE Transactions on Neural Networks and Learning Systems, 2003. **14**(1): p. 138- 149.
11. Tatiana, B., et al., Neural classifier for larvae recognition, in Proceedings of International Joint Conference on Neural Networks. 2010. p. 1-4.
12. Wenjing, L., G. Bebis, and N.G. Bourbakis, 3-d object recognition using 2-d views. IEEE Transactions on Image Processing, 2008. **17**(11): p. 2236-2255.
13. Khunarsal, P., C. Lursinsap, and T. Raicharoen, Singing voice recognition based on matching of spectrogram pattern, in Proceedings of IJCNN International Joint Conference on Neural Networks. 2009. p. 1595 - 1599.

14.Leung, K., et al., Application of a modified neural fuzzy network and an improved genetic algorithm to speech recognition. Neural Computing and Applications, 2007. **16**(4): p. 419-431.

15.Maglogiannis, I., et al., Radial basis function neural networks classification for the recognition of idiopathic pulmonary fibrosis in microscopic images. IEEE Transactions on Information Technology in Biomedicine, 2008. **12**(1): p. 42-54.

16.George, M., Radial basis function neural networks and principal component analysis for pattern classification, in Proceedings of 2007 International Conference on Computational Intelligence and Multimedia Applications. 2007. p. 200-206.

17.Jadhav, S.M., S.L. Nalbalwar, and A.A. Ghatol, Arrhythmia disease classification using artificial neural network model, in Proceedings of IEEE International Conference on Computational Intelligence and Computing Research. 2010. p. 1-4.

18.Salankar, S.S. and B.M. Patre, RBF neural network based model as an optimal classifier for the classification of radar returns from the ionosphere, in Proceedings of IEEE International Conference on Industrial Technology. 2006. p. 2043 - 2048.

19.Srinivasan, D., W.S. Ng, and A.C. Liew, Neural-network-based signature recognition for harmonic source identification. IEEE Transactions on Power Delivery, 2006. **21**(1): p. 398-405.

20.Duan, H., et al., An incremental learning algorithm for Lagrangian support vector machine. Pattern Recognition Letters, 2009. **30**(15): p. 1384-1391.

21.Polikar, R., et al., Learn++: an incremental learning algorithm for supervised neural networks. IEEE Transactions on Systems, Man Cybernetics, Part C: Applications and Reviews, 2001. **31**(4): p. 497-508.

22.Wilson, D.R. and T.R. Martinez, The general inefficiency of batch training for gradient descent learning. Neural Networks, 2003. **16**: p. 1429-1451.

23.Constantinopoulos, C. and A. Likas, An incremental training method for the probabilistic RBF network. IEEE Transactions on Neural Networks and Learning Systems, 2006. **17**(4): p. 966 - 974.

24.Furao, S. and O. Hasegawa, A fast nearest neighbor classifier on self-organizing incremental neural network. Neural Networks, 2008. **21**: p. 1537-1547.

25.Domingos, P. and G. Hulten, Mining High-Speed Data Streams, in Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. 2000: Boston, MA, USA. p. 71-80.

26.Pang, S., S. Ozawa, and N. Kasabov, Incremental learning discriminant analysis classification of data streams. IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics., 2005. **35**(5): p. 905-914.

27.Wan, S. and L.E. Banta, Parameter Incremental Learning Algorithm for Neural Networks. IEEE Transactions on Neural Networks and Learning Systems 2006. **17**(6): p. 1424-1438.

28.Ozawa, S., S. Pang, and N. Kasabov, Incremental learning of chunk data for online pattern classification systems. IEEE Transactions on Neural Networks and Learning Systems, 2008. **19**(6): p. 1061-1074.

29.Ye, X., S. Furao, and Z. Jinxi, An incremental learning vector quantization algorithm for pattern classification. Neural Computing and Applications, 2012. **21**(6): p. 1205-1215.

30.Bache, K. and M. Lichman, UCI Machine Learning Repository. 2013, Irvine, CA: University of California, School of Information and Computer Science.

31.Haykin, S., Neural networks: A comprehensive foundation. 2nd ed. 1999: Prentice-Hall International, Inc.

32.Martinez-Rego, D., et al., A robust incremental learning method for non-stationary environments. Neurocomputing, 2011. **74**(11): p. 1800-1808.

APPENDIX

Theorem 1. Given a data set $\mathbf{Y}$ used for parameter adjustment of a VEBF, assume that a data points $\mathbf{y}_i \in \mathbf{Y}$ is uncovered by the closest j^{th} VEBF, $\psi(\mathbf{y}_i : \bar{\mathbf{x}}^j, \mathbf{w}^j, \mathbf{U}^j)$, If the width vector $\mathbf{w}^j = [w_1^j \; w_2^j \; \cdots \; w_n^j]^T$ is updated by $\mathbf{w}_l^j = \varepsilon \mathbf{w}_l^j$, where $\varepsilon = \sqrt{1 + \eta * max_{val}}$ and $\eta \geq 1$, then the data set $\mathbf{Y}$ will eventually be covered by the VEBF $\psi(\mathbf{y}_i : \bar{\mathbf{x}}^j, \mathbf{w}^j, \mathbf{U}^j)$.

Proof

Let $\eta \geq 1$ *and* $max_{val} = \max_{\mathbf{y}_i \in \mathbf{Y}} (\psi(\mathbf{y}_i, \bar{x}^j, \mathbf{w}^j, \mathbf{U}^j))$

So, $\forall \mathbf{y}_i \in \mathbf{Y}$, $\psi(\mathbf{y}_i, \bar{\mathbf{x}}^j, \mathbf{w}^j, \mathbf{U}^j) \leq max_{val}$

$$\sum_{l=1}^{n} \frac{\left((\mathbf{y}_i - \bar{\mathbf{x}}^j)^T \mathbf{u}_l^j\right)^2}{\left(w_l^j\right)^2} - 1 \leq max_{val}$$

Since $\eta \geq 1$,

$$\sum_{l=1}^{n} \frac{\left((\mathbf{y}_i - \bar{\mathbf{x}}^j)^T \mathbf{u}_l^j\right)^2}{\left(w_l^j\right)^2} - 1 \leq \eta * max_{val}$$

$$\sum_{l=1}^{n} \frac{\left((\mathbf{y}_i - \bar{\mathbf{x}}^j)^T \mathbf{u}_l^j\right)^2}{\left(w_l^j \sqrt{1 + \eta * max_{val}}\right)^2} \leq 1$$

$$\sum_{l=1}^{n} \frac{\left((\mathbf{y}_i - \bar{\mathbf{x}}^j)^T \mathbf{u}_l^j\right)^2}{\left(w_l^j \sqrt{1 + \eta * max_{val}}\right)^2} - 1 \leq 0$$

$$\psi(\mathbf{y}_i, \bar{x}^j, \varepsilon \mathbf{w}^j, \mathbf{U}^j) \leq 0, \text{ where } \varepsilon = \sqrt{1 + \eta * max_{val}}.$$

By Definition 1, *the data set* $\mathbf{Y}$ *is covered if the width vector is updated by* $\mathbf{w}^j = \varepsilon \mathbf{w}^j$ *where* $\varepsilon = \sqrt{1 + \eta * max_{val}}$.

Theorem 2. Given a data chunk $\wp$ with *n* samples, the time complexity T_{alg} of the Data-throwaway Learning Streaming Chunk (DLSC) algorithm is $O(K+n^2)$, where K the number of hidden neurons before presenting a data chunk $\wp$.

Proof

Given VEBFNN with K hidden neurons and a data chunk $\wp$ with *n* samples, $\wp=\left\{\{\mathbf{X}^j,t^j\}_{j=1}^{p}\right\}$ where $\{\mathbf{X}^j,t^j\}$ is the set samples $\mathbf{X}^j$ with the class label t^j and *p* is a small number. Let $\left|\mathbf{X}^j\right|$ denote the number of samples belonging to class label t^j and $\sum_{j=1}^{p}|\mathbf{X}^j|=n$. At lines 1-4, since the since the computational time does not depend on the number of data, the time taken in these lines is constant and then, given by $T_1=O(1)$.

For each category $\{\mathbf{X}^j,t^j\}$, if t^j is the new class label, then Lines 6-13 are performed. The time complexity is $T_2=O(1)$ at lines 6. In Do While loop at line 7, the worst case is considered. Because the learning time of *CreateNewNeuron* at line 8 does not depend on the number of data, the time complexity is $O(1)$. Thus, the time complexity of *CreateNewNeuron* for all data with *p* class label is $T_3=pO(1)=O(1)$. Then, the ***UpdateParameter*** at line 10 is executed until $\mathbf{X}^j$ is empty. The worst case is considered. The worst case appears when the size of each category $\mathbf{X}^j$ is reduced by one at a time for parameter update in ***UpdateParameter***. The function ***UpdateParameter*** contains a subfunction called *FormCoveredData*. There are four operations including three computations and one comparison within this subfunction. The computational time of each operation is constant. Since the size of the $\mathbf{X}^j$ is reduced by one at a time for worst case, the computational time for each operation is given by

$(|\mathbf{X}^j|)+(|\mathbf{X}^j|-1)+(|\mathbf{X}^j|-2)+\ldots+2+1=\frac{1}{2}(|\mathbf{X}^j|+1)|\mathbf{X}^j|\in O(|\mathbf{X}^j|^2)$. Then, the time for $\mathbf{X}^j$ is $T_4=4\times O(|\mathbf{X}^j|^2)$. Within ***UpdateParameter***, the worst case appears when only one datum is used for parameter update given by $|\mathbf{Y}|=1$. Then, the time does not depend on the number of data. The time is $O(|\mathbf{X}^j|)$. The time of

UpdateParameter in Do-While loop is $O(|\mathbf{X}^j|^2)+O(|\mathbf{X}^j|^2)=O(|\mathbf{X}^j|^2)$. So, the time for new class label is $T_2+T_3+T_4=O(1)+O(1)+O(|\mathbf{X}^j|^2)=O(|\mathbf{X}^j|^2)$.

If t^j is the old class label, then Lines 15-31 are performed. As the old class label, the worst case is considered. In Do-While loop at lines 15-23, the worst case appears when the data $\mathbf{X}^j$ reduced by one for the data $\mathbf{X}^j$. The computational time at line 16 is $T_5=O(|\mathbf{X}^j|)$. At line 17, the time is $T_6=O(K)$. For merging process, the time of worst case is equal to $T_7=K+|\mathbf{X}^j|=O(K+|\mathbf{X}^j|)$. In Do-While loop at lines 24-29, the time of worst case is $T_8=O(|\mathbf{X}^j|^2)$ as the old class label case.

Therefore, the time complexity of the DLSC algorithm for p classes is T_1 + time for each new class+ time for each old class $\leq T_1+p(T_2+T_3+T_4)+p(T_5+T_6+T_7+T_8)$

$$\leq O(1)+\left(pO(1)+pO(1)+\sum_{j}^{p}O(|\mathbf{X}^j|^2)\right)+(\sum_{j=1}^{p}O(|\mathbf{X}^j|)+$$

$$pO(K)+O(K+\sum_{j}^{p}|\mathbf{X}^j|)+O(\sum_{j}^{p}|\mathbf{X}^j|^2)\in O(K+\sum_{j}^{p}|\mathbf{X}^j|^2)=O(K+n^2).$$

VITA

Name: Mr. Prem Junsawang

Date of Birth: 28th Febuary, 1982

Educations:

M.Sc. degree in Computational Science, Department of Mathematics, Faculty of Science, Chulalongkorn University, Bangkok, Thailand.

B.Sc. degree (honors) in Mathematics, Department of Mathematics, Faculty of Science, Khon Kaen University, Khon Kaen, Thailand.

Scholarship: Development and Promotion of Science and Technology Talents Project (DPST), the Institute for Promotion and Teaching Science and Technology (IPST), Ministry of Science and Technology.