การเรียนรู้เพียงครั้งเดียวแบบเพิ่มเติมได้สำหรับปัญหาการแบ่งกลุ่มข้อมูลโดยใช้โครงข่ายประสาทเทียมและฟังก์ชันไฮเพอร์อัลลิปซอยด์

One-Pass Incremental Learning for Classification Problem Using Neural Network and Hyper-Ellipsoidal Function

สายชล ใจเย็น ชิดชนก เหลือสินทรัพย์* และศุภกานต์ พิมลธเรศ

ศูนย์วิจัย AVIC ภาควิชาคณิตศาสตร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

Saichon Jaiyen, Chidchanok Lursinsap* and Suphakant Phimoltares

Research Center of AVIC, Department of Mathematics, Faculty of Science, Chulalongkorn University

บทคัดย่อ

บทความนี้นำเสนอโครงข่ายประสาทเทียมชนิดใหม่แบบเรียนรู้ข้อมูลใหม่เพียงครั้งเดียวแล้วทิ้งข้อมูลนั้นโดยไม่ต้องใช้ข้อมูลเก่าที่เรียนรู้ไปแล้วด้วยการใช้ฟังก์ชันไฮเพอร์อัลลิปซอยด์ ฟังก์ชันชนิดนี้มีรูปทรงเป็นทรงรีและสามารถเรียนรู้ข้อมูลในหลายมิติได้โดยการล้อมข้อมูลที่เข้ามา ข้อมูลที่ใช้ในการปรับค่าตัวแปรต่างๆ ของฟังก์ชันชนิดนี้จะใช้เพียงข้อมูลที่เข้ามาใหม่เพียงข้อมูลเดียว โครงข่ายประสาทเทียมชนิดนี้สามารถเรียนรู้ข้อมูลที่ใช้สำหรับเรียนรู้โดยใช้เวลา $O(n)$ เมื่อ n คือจำนวนข้อมูลสำหรับเรียนรู้

คำสำคัญ : การแบ่งกลุ่มข้อมูล การเรียนรู้อย่างเร็ว การวิเคราะห์ส่วนประกอบ ฟังก์ชันไฮเพอร์อัลลิปซอยด์

Abstract

This paper introduces a new neural network which can learn each new data only in one epoch and, then, throw it away without involving the previously learned data by using a hyper-ellipsoidal function. This function has an elliptical shape and can learn any multi-dimensional data by surrounding them. Only one new incoming data is used to adjust the learning parameters. This neural network spends $O(n)$ time complexity to learn a set of n data.

Keywords : classification, fast learning, principal component

***Corresponding author.** E-mail: lchidcha@gmail.com

บทนำ

โครงข่ายประสาทเทียมชนิด Radial Basis function (RBF) เป็นโครงข่ายประสาทเทียมที่นิยมใช้กันมากในการประมาณฟังก์ชันเนื่องจากสามารถประมาณฟังก์ชันได้โดยตรงจากอินพุต (input) และเอาต์พุต (output) โดยใช้โครงสร้างของโครงข่ายประสาทเทียมแบบง่ายและไม่ซับซ้อน แต่โครงข่ายประสาทเทียมชนิดนี้มีข้อเสียคือ จำนวนของเซลล์ประสาทเทียมจะถูกกำหนดแน่นอนก่อนที่จะเรียนรู้ข้อมูลที่ใช้สำหรับให้โครงข่ายประสาทเทียมเรียนรู้ นอกจากนี้เรายังไม่สามารถรู้ล่วงหน้าได้ว่าจะต้องใช้จำนวนของเซลล์ประสาทเทียมเท่าไรถึงจะเหมาะสมกับข้อมูลที่โครงข่ายประสาทเทียมจะต้องเรียนรู้ จึงทำให้โครงข่ายประสาทเทียมชนิดนี้ไม่ยืดหยุ่นต่อการใช้งานจริง Platt (1991) เสนออัลกอริทึมที่เรียนรู้แบบลำดับ (sequential learning) สำหรับโครงข่ายประสาทเทียมชนิด Radial Basis function ที่สามารถเรียนรู้ข้อมูลที่ละตัวโดยที่ในตอนเริ่มต้นจะยังไม่มีเซลล์ประสาทเทียมอยู่ในโครงข่ายเลยแต่จำนวนของเซลล์ประสาทเทียมจะค่อยๆ เพิ่มขึ้นระหว่างการเรียนรู้โครงข่ายประสาทเทียมนี้รู้จักในชื่อ resource allocation network (RAN) แต่เนื่องจากโครงข่ายประสาทเทียมชนิดนี้ใช้อัลกอริทึม least-mean square (LMS) ในการปรับค่าน้ำหนัก (weight) ของโครงข่ายประสาทเทียมทำให้ใช้เวลาในการเรียนรู้มาก ต่อมา Kadirkamanathan และ Niranjan (1993) ได้ปรับปรุงวิธีการของ RAN โดยใช้วิธี extended Kalman filter (EKF) ในการปรับค่าน้ำหนักของโครงข่ายประสาทเทียมแทนวิธี LMS ซึ่งรู้จักในชื่อ RAN extended Kalman filter (RANEKF) อย่างไรก็ตามโครงข่ายประสาทเทียมทั้งสองชนิดนี้ก็มีข้อเสียคือจำนวนของเซลล์ประสาทเทียมจะเพิ่มขึ้นได้อย่างเดียว แต่ไม่มีวิธีการที่จะลดจำนวนของเซลล์ประสาทเทียมที่ไม่จำเป็นออกจากโครงข่ายประสาทเทียมต่อมา Yingwei *et al.* (1997) ได้ปรับปรุงวิธีการของ RANEKF โดยการเพิ่มวิธีการที่จะลบเซลล์ประสาทเทียมที่ไม่จำเป็นออกไปซึ่งรู้จักในชื่อ minimal resource allocating network (MRAN) ต่อมา Li Yan *et al.* (2000) ได้ปรับปรุงวิธีการของ (MRAN) ซึ่งรู้จักกันในชื่อ Extended-MRAN (EMRAN) แต่อย่างไรก็ตามเมื่อสิ้นสุดกระบวนการเรียนรู้และโครงข่ายประสาทเทียมได้ถูกนำไปใช้งานแล้ว แต่ต่อมาในภายหลังเรามีข้อมูลชุดใหม่ที่ต้องการให้โครงข่ายประสาทเทียมเหล่านี้เรียนรู้ โครงข่ายประสาทเทียมเหล่านี้จะไม่สามารถที่จะเรียนรู้ข้อมูลชุดใหม่นี้ได้โดยปราศจากข้อมูลชุดเดิม ทำให้โครงข่ายประสาทเทียมเหล่านี้ไม่มีความสามารถเรียนรู้ความรู้ใหม่ได้

โครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์

การเรียนรู้ความรู้ใหม่โดยปราศจากความรู้เก่าเป็นปัญหาที่สำคัญมากในงานวิจัยทางด้านโครงข่ายประสาทเทียมซึ่งจะทำให้การเรียนรู้ของโครงข่ายประสาทเทียมใกล้เคียงกับความสามารถของมนุษย์มากขึ้น ในบทความนี้จึงนำเสนอโครงข่ายประสาทเทียมชนิดใหม่ที่ใช้ฟังก์ชันไฮเพอร์อัลลิปซอยด์ที่สามารถปรับเปลี่ยนตำแหน่ง ขนาด และทิศทางตามการกระจายตัวของข้อมูล โครงสร้างของเครือข่ายชนิดนี้ประกอบด้วยชั้นของเซลล์ประสาทเทียมจำนวนสามเลเยอร์ด้วยกันคือ อินพุตเลเยอร์ (input layer) ฮิดเดนเลเยอร์ (hidden layer) และเอาต์พุตเลเยอร์ (output layer) ในฮิดเดนเลเยอร์จะถูกแบ่งออกเป็นส่วนย่อยๆ ตามจำนวนของกลุ่มข้อมูลซึ่งจะเรียกว่า ฮิดเดนเลเยอร์ย่อย (subhidden layer) ในตอนเริ่มต้นของการเรียนรู้โครงข่ายประสาทเทียมชนิดนี้จะยังไม่มีเซลล์ประสาทเทียมอยู่ในโครงข่ายแต่จำนวนของเซลล์ประสาทเทียมจะค่อยๆ เพิ่มขึ้นเรื่อยๆ ในระหว่างการเรียนรู้ ถ้ากำหนดให้เวกเตอร์ $\mathrm{x} = [x_1, x_2, \ldots, x_n]^T$ เป็นเวกเตอร์ใน $\Re^n$ เอาต์พุตของเซลล์ประสาทเทียมที่ k ในฮิดเดนเลเยอร์ คือ $\varphi_k(\mathrm{x})$ สามารถคำนวณได้ดังสมการต่อไปนี้

$$\varphi_k(\mathbf{x}) = \sum_{i=1}^{n} \frac{((\mathbf{x}\text{-}\mathrm{c}_k)^T \mathrm{u}_i)^2}{a_i^2} - 1,$$

โดยที่ u_i เป็นเวกเตอร์หนึ่งหน่วยตามแนวแกนในแต่ละแกนของฟังก์ชันไฮเพอร์อัลลิปซอยด์ a_i เป็นความกว้างในแต่ละแกนของฟังก์ชันไฮเพอร์อัลลิปซอยด์ และ c_k เป็นจุดศูนย์กลางของฟังก์ชันไฮเพอร์อัลลิปซอยด์ เอาต์พุต (output) ของเซลล์ประสาทเทียมที่ p ในเอาต์พุตเลเยอร์คือ $y_p(\mathrm{x})$ สามารถคำนวณได้โดยใช้สมการดังต่อไปนี้

$$y_p(\mathbf{x}) = \min_j(\varphi_j(\mathbf{x}))$$

โดยที่ $\varphi_j(\mathrm{x})$ เป็นเอาต์พุตของเซลล์ประสาทเทียมที่ j ในฮิดเดนเลเยอร์ เนื่องจากจำนวนของเอาต์พุตในเอาต์พุตเลเยอร์มีจำนวนเท่ากับจำนวนของฮิดเดนเลเยอร์ย่อย เพราะฉะนั้นเราจำเป็นต้องมีฟังก์ชันที่ใช้ในการตัดสินใจว่าจะให้อินพุตที่เข้ามาอยู่ในกลุ่มไหน ซึ่งเราจะนิยามฟังก์ชันการตัดสินใจ $D(\mathrm{x})$ ดังต่อไปนี้

$$D(\mathrm{x}) = \begin{cases} k, \text{ if } k = \arg\min(y_i(\mathrm{x})) \text{ and } y_k(\mathrm{x}) \le 0 \\ \mathit{unknown}, \text{ otherwise} \end{cases}$$

ถ้าเรามีข้อมูลชุดใหม่เข้ามาสมมติว่าเป็นเวกเตอร์ x เวกเตอร์นี้จะถูกจัดอยู่ในกลุ่ม k หรือไม่ก็อยู่ในกลุ่ม *unknown*

การคำนวณเวกเตอร์ค่าเฉลี่ย และโคแวเรียนเมทริกซ์ แบบรีเคอร์ซีฟ

ในการหาจุดศูนย์กลางของฟังก์ชันไฮเพอร์อัลลิปซอยด์นั้น เราจะใช้ค่าเฉลี่ยในการคำนวณหาจุดศูนย์กลางของฟังก์ชันไฮเพอร์อัลลิปซอยด์ แต่เนื่องจากการคำนวณค่าเฉลี่ยโดยใช้สูตรคำนวณแบบเดิมเราต้องใช้ข้อมูลทั้งหมดในการคำนวณ ซึ่งไม่เหมาะกับการเรียนรู้แบบเพิ่มเติมได้ ดังนั้นเราจึงนำสูตรคำนวณหาค่าเฉลี่ยแบบเดิมมาเขียนใหม่ในรูปของรีเคอร์ซีฟระหว่างค่าเฉลี่ยใหม่และค่าเฉลี่ย เก่าดังทฤษฎีบทต่อไปนี้

ทฤษฎีบทที่ 1

กำหนดให้ $X=\{\mathrm{x}_1,\mathrm{x}_2,...,\mathrm{x}_N\}$ เป็นเซตของข้อมูล โดยที่ x_i เป็นเวกเตอร์ใน $\Re^n$ และ กำหนดให้ μ_{old} เป็นเวกเตอร์ค่าเฉลี่ยของเซตนี้ ถ้า x_{N+1} เป็นข้อมูลตัวใหม่ที่ถูกเพิ่มเข้าไปในเซตนี้ จะได้ว่า

$$\mu_{new}=\alpha\mu_{old}+\beta$$

โดยที่ μ_{new} เป็น เวกเตอร์ค่าเฉลี่ยใหม่ $\alpha=\dfrac{N}{N+1}$ และ $\beta=\dfrac{\mathrm{x}_{N+1}}{N+1}$ ในการคำนวณหาค่าของเวกเตอร์หนึ่งหน่วยตามแกนของฟังก์ชันไฮเพอร์อัลลิปซอยด์เราจะคำนวณจากโคแวเรียนเมทริกซ์ แต่วิธีการหาค่าโคแวเรียนเมทริกซ์แบบเดิมจะต้องใช้ข้อมูลทั้งหมดในการคำนวณซึ่งจะไม่เหมาะสมกับการเรียนรู้แบบเพิ่มเติมได้ ดังนั้นเราจึงควรเขียนเมทริกซ์ใหม่ในรูปแบบของรีเคอร์ซีฟดังแสดงในทฤษฎีบทที่ 2

ทฤษฎีบทที่ 2

กำหนดให้ $X=\{\mathrm{x}_1,\mathrm{x}_2,...,\mathrm{x}_N\}$ เป็นเซตของข้อมูล โดยที่ x_i เป็นเวกเตอร์ใน $\Re^n$ และ กำหนดให้ S_{old} เป็นโคแวเรียนเมทริกซ์ ของเซตนี้ ถ้า x_{N+1} เป็นข้อมูลตัวใหม่ที่ถูกเพิ่มเข้าไปในเซตนี้ จะได้ว่า

$$S_{new}=\alpha S_{old}+\kappa$$

โดยที่ $\alpha=\dfrac{N}{N+1}$,

$$\kappa=\kappa_1+\kappa_2,$$

$$\kappa_1=\frac{\mathrm{x}_{N+1}\mathrm{x}_{N+1}^T}{N+1}-\mu_{new}\mu_{new}^T+\mu_{old}\mu_{old}^T,$$

$$\kappa_2=-\frac{\mu_{old}\mu_{old}^T}{N+1},$$

μ_{new} เป็นเวกเตอร์ค่าเฉลี่ยใหม่

และ S_{new} เป็นโคแวเรียนเมทริกซ์ ใหม่

อัลกอริทึมการเรียนรู้สำหรับโครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์

การเรียนรู้สำหรับโครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์นี้จะเกี่ยวข้องกับการปรับค่าพารามิเตอร์ต่างๆ ของฟังก์ชันไฮเพอร์อัลลิปซอยด์ซึ่งได้แก่ จุดศูนย์กลางของฟังก์ชันไฮเพอร์อัลลิปซอยด์ เวกเตอร์หนึ่งหน่วยตามแกนของ ฟังก์ชันไฮเพอร์อัลลิปซอยด์ ความกว้างในแต่ละแกนของฟังก์ชันไฮเพอร์อัลลิปซอยด์ กำหนดให้ $\Omega=\{\Omega_k \mid 1\le k\le K\}$ เป็นเซตของเซลล์ประสาทเทียม K ตัว กำหนดให้ $\Omega_k=(C^k,S^k,N_k,A_k,d_k)$ เป็นเซลล์ประสาทเทียมที่ k โดยที่ C^k เป็นจุดศูนย์กลางของเซลล์ประสาทเทียมที่ k, S^k เป็นโคแวเรียนเมทริกซ์ ของเซลล์ประสาทเทียมที่ k, N_k เป็นจำนวนของข้อมูลของเซลล์ประสาทเทียมที่ k, $A_k=[a_1^k,a_2^k,...,a_n^k]^T$ เป็นความกว้างของเซลล์ประสาทเทียมที่ k, และ D_k เป็นชื่อคลาสของเซลล์ประสาทเทียมที่ k เรากำหนดให้ N_0 เป็นค่าคงที่ที่ใช้สำหรับปรับค่าความกว้าง A_k

1 การสร้างเซลล์ประสาทเทียมใหม่

ในตอนเริ่มต้นของการเรียนรู้ จะไม่มีเซลล์ประสาทเทียมอยู่ในโครงข่ายประสาทเทียมเลย แต่จำนวนเซลล์ประสาทเทียมจะเพิ่มขึ้นในระหว่างการเรียนรู้ข้อมูลใหม่ๆ สมมติให้ x_j เป็นอินพุตของโครงข่ายประสาทเทียม และ t_j เป็นชื่อคลาสของ x_j ในขณะที่โครงข่ายประสาทเทียมเรียนรู้อินพุต x_j ก็จะค้นหาเซลล์ประสาทเทียมที่อยู่ใกล้กับอินพุต x_j มากที่สุด ถ้าหาเซลล์ประสาทเทียมดังกล่าวได้ ค่าพารามิเตอร์ของฟังก์ชันไฮเพอร์อัลลิปซอยด์ของเซลล์ที่ได้จะถูกปรับแบบชั่วคราวดังนี้

(1) คำนวณค่า C_{new}^{cs} จากทฤษฎีบทที่ 1

(2) คำนวณค่า S_{new}^{cs} จากทฤษฎีบทที่ 2

(3) คำนวณค่า eigenvalue ของ S_{new}^{cs}

$$\lambda_1>\lambda_2>,..,>\lambda_n$$

(4) คำนวณค่า eigenvector ของ S_{new}^{cs}

$$\mathrm{u}_1^{cs},\mathrm{u}_2^{cs},..,\mathrm{u}_n^{cs}$$

กำหนดให้ u_i^{cs} เป็นเวกเตอร์หนึ่งหน่วยตามแนวแกน i ของฟังก์ชันไฮเพอร์อัลลิปซอยด์ การที่จะทดสอบว่าเซลล์ประสาทเทียมนี้สามารถคลุมข้อมูลใหม่ได้หรือเปล่า เราจะทดสอบโดยใช้สมการดังต่อไปนี้

$$\varphi_{cs}(\mathrm{x}_j)=\sum_{i=1}^{n}\frac{((\mathrm{x}_j-C_{new}^{cs})^T\mathrm{u}_i^{cs})^2}{(a_i^{cs})^2}-1$$

เมื่อ cs คืออินเด็กซ์ของเซลล์ประสาทเทียมที่อยู่ใกล้กับอินพุตมากที่สุดและ n คือจำนวนมิติของอินพุต ถ้าค่าของ $\varphi_{cs}(\mathrm{x}_j) > 0$ แสดงว่าเซลล์ประสาทเทียมนี้ไม่สามารถคลุมอินพุตนี้ได้ ดังนั้นเซลล์ประสาทตัวใหม่จะถูกสร้างขึ้นและถูกเพิ่มเข้าไปในโครงข่ายประสาทเทียมและค่าพารามิเตอร์ของเซลล์ประสาทเทียมใหม่นี้จะถูกกำหนดค่าเริ่มต้นใหม่ดังนี้

$$C^{K+1} = \mathrm{x}_i$$
$$S^{K+1} = 0 \ \text{(zero matrix)}$$
$$N_{K+1} = 1$$
$$d_{K+1} = t_j$$
$$A_{K+1} = A_0$$

เมื่อ A_0 คือพารามิเตอร์ความกว้างเริ่มต้นของเซลล์ประสาทเทียม

2 และการปรับค่าพารามิเตอร์ของเซลล์ประสาทเทียม

ถ้าค่าของ $\varphi_{cs}(\mathrm{x}_i) \le 0$ แสดงว่าเซลล์ประสาทเทียมนี้สามารถคลุมข้อมูลตัวนี้ได้ ดังนั้นค่าพารามิเตอร์ของเซลล์ประสาทเทียมนี้จะถูกปรับตามค่าพารามิเตอร์ชั่วคราวของฟังก์ชันไฮเพอร์อัลลิปซอยด์ดังต่อไปนี้

$$C^k = C^{cs}_{new}$$
$$S^k = S^{cs}_{new}$$
$$N_k = N_k + 1$$

ถ้า $N_k > N_0$ แล้ว $a_i^k = a_i^k + \left|(C^k_{new} - C^k_{old})^T \mathrm{u}_\mathrm{i}\right|$

3 วิธีการลบเซลล์ประสาทเทียมที่ไม่จำเป็นออกจากโครงข่ายประสาทเทียม

เนื่องจากจำนวนเซลล์ประสาทเทียมจะเพิ่มขึ้นในระหว่างการเรียนรู้ ดังนั้นอาจมีเซลล์ประสาทเทียมบางตัวที่ไม่มีความจำเป็นต้องใช้ เราจึงควรมีวิธีการที่จะกำจัดเซลล์ที่ไม่มีความจำเป็นเหล่านี้ในโครงข่ายประสาทเทียม กำหนดให้

$\Omega_x = (C^x, S^x, N_x, A_x, d_x)$ และ

$\Omega_y = (C^y, S^y, N_y, A_y, d_y)$

เป็นเซลล์ประสาทเทียมสองตัวใดๆ ในโครงข่ายประสาทเทียม ถ้าเรารวมสองเซลล์ประสาทเทียมนี้เข้าด้วยกันกลายเป็นเซลล์ประสาทเทียมตัวใหม่คือ

$$\Omega_{new} = (C^{new}, S^{new}, N_{new}, A_{new}, d_{new})$$

ค่าพารามิเตอร์ของเซลล์ประสาทเทียมตัวใหม่สามารถคำนวณได้ดังสมการต่อไปนี้

$$C^{new} = \frac{1}{N_x + N_y}\left(N_x C^x + N_y C^y\right)$$
$$S^{new} = \frac{N_x}{N_x + N_y} S^x + \frac{N_y}{N_x + N_y} S^y$$
$$\frac{N_x N_y}{(N_x + N_y)^2}\left(C^x - C^y\right)\left(C^x - C^y\right)^T$$
$$N_{new} = N_x + N_y$$
$$a_j^{new} = \sqrt{2\pi\left|\lambda_j\right|} + \delta,\ j = 1, 2, ..., n$$
$$d_{new} = d_x$$

ผลการทดลอง

เราจะทดสอบประสิทธิภาพของโครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์ (VEBF) เทียบกับโครงข่ายประสาทเทียมที่นิยมใช้กันในปัจจุบันซึ่งได้แก่ โครงข่ายประสาทเทียม Radial Basis function (RBF) และ โครงข่ายประสาทเทียม Multilayer Perceptron (MLP) เราจะใช้ข้อมูลมาตรฐานที่นิยมใช้กันมากในการทดสอบโมเดลของโครงข่ายประสาทเทียม โดยชุดข้อมูลดังกล่าวเราเลือกมาจากฐานข้อมูล UCI [6] เป็นชุดข้อมูลที่ใช้ในการเรียนรู้ของโครงข่ายประสาทเทียมและใช้ทดสอบโครงข่ายประสาทเทียม และเราใช้วิธี 5-fold cross-validation สำหรับทุกโมเดล โดยวิธีการนี้จะแบ่งข้อมูลออกเป็น 5 ส่วนโดยข้อมูลในแต่ละส่วนจะไม่ซ้ำกัน ซึ่งเราจะใช้ 4 ส่วนสำหรับให้โครงข่ายประสาทเทียมเรียนรู้ และอีก 1 ส่วนสำหรับทดสอบโมเดลและทำซ้ำจนครบ 5 ครั้ง แล้วผลลัพธ์จะถูกนำมาเฉลี่ย สำหรับค่าความกว้างเริ่มต้นของ RBF และ VEBF เราจะคำนวณโดยใช้สมการดังนี้

$$a_k = \delta * d_{av}$$

โดยที่ δ เป็นค่าคงที่ และ $d_{av} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}$ และ d_{ij} เป็น Euclidean distance ระหว่างข้อมูลตัวที่ i และตัวที่ j

Iris Data Set

Iris Data Set มีข้อมูลทั้งหมด 150 ตัว โดยที่ข้อมูลแต่ละตัวประกอบด้วยแอตทริบิวทั้งหมด 4 แอตทริบิวและมีทั้งหมด 3 คลาสด้วยกัน ในการทดลองนี้เรากำหนดค่า δ = 1/3 สำหรับ VEBF และ δ = 10 สำหรับ RBF สำหรับ MLP เราจะรันโมเดลนี้จำนวน 10 ครั้งโดยเราจะเลือกครั้งที่ดีที่สุดของโมเดลนี้เป็นผลการทดลอง ในการทดลองนี้ VEBF เรียนรู้ข้อมูลเพียงรอบเดียวในขณะที่ RBF และ MLP เรียนรู้ข้อมูลหลายรอบ

ตารางที่ 1 ผลการทดลองจากการทดสอบด้วย Iris Data Set

Testing fold	**VEBF**			**RBF**			**MLP**		
	Time (second)	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**
1	0.08	3	96.67	1.18	3	96.67	1.11	3	96.67
2	0.03	3	100.00	0.43	3	100.00	0.13	3	96.67
3	0.03	3	96.67	0.43	3	100.00	0.10	3	100.00
4	0.03	3	100.00	0.44	3	96.67	0.10	3	96.67
5	0.03	3	96.67	0.43	3	96.67	0.10	3	96.67
Average	0.04	3	**98.00**	0.58	3	98.00	0.31	3	97.33

ผลการเปรียบเทียบดังแสดงในตารางที่ 1 จากตารางนี้เราจะเห็นว่าค่าเฉลี่ยของความถูกต้อง (accuracy) ของ VEBF จะเท่ากับ MLP แต่สูงกว่า RBF แต่เวลาที่ใช้ในการเรียนรู้ของ VEBF จะใช้เวลาน้อยกว่า RBF และ MLP

Balance Scale Data Set

Balance Scale Data Set มีข้อมูลทั้งหมด 625 ตัว โดยที่ข้อมูลแต่ละตัวประกอบด้วยแอตทริบิวทั้งหมด 4 แอตทริบิวและมีทั้งหมด 3 คลาสด้วยกัน ในการทดลองนี้เรากำหนดค่า δ = 0.05 สำหรับ VEBF และ δ = 1 สำหรับ RBF สำหรับ MLP เราจะรันโมเดลนี้จำนวน 10 ครั้งโดยเราจะเลือกครั้งที่ดีที่สุดของโมเดลนี้เป็นผลการทดลอง ในการทดลองนี้ VEBF เรียนรู้ข้อมูลเพียงรอบเดียวในขณะที่ RBF และ MLP เรียนรู้ข้อมูลหลายรอบ ผลการเปรียบเทียบดังแสดงในตารางที่ 2 จากตารางนี้เราจะเห็นว่าค่าเฉลี่ยของความถูกต้อง (accuracy) ของ VEBF สูงกว่า RBF และ MLP นอกจากนี้เวลาที่ใช้ในการเรียนรู้ของ VEBF จะใช้เวลาน้อยกว่า RBF และ MLP

Parkinson Data Set

Parkinson Data Set มีข้อมูลทั้งหมด 197 ตัว โดยที่ข้อมูลแต่ละตัวประกอบด้วยแอตทริบิวทั้งหมด 23 แอตทริบิว และมีทั้งหมด 2 คลาสด้วยกัน ในการทดลองนี้เรากำหนดค่า δ = 0.05 สำหรับ VEBF และ δ = 1 สำหรับ RBF สำหรับ MLP เราจะรันโมเดลนี้จำนวน 10 ครั้งโดยเราจะเลือกครั้งที่ดีที่สุดของโมเดลนี้เป็นผลการทดลอง ในการทดลองนี้ VEBF เรียนรู้ข้อมูลเพียงรอบเดียวในขณะที่ RBF และ MLP เรียนรู้ข้อมูลหลายรอบ ผลการเปรียบเทียบดังแสดงในตารางที่ 3 จากตารางนี้เราจะเห็นว่า ค่าเฉลี่ยของความถูกต้อง (accuracy) ของ VEBF สูงกว่า RBF และ MLP นอกจากนี้เวลาที่ใช้ในการเรียนรู้ของ VEBF จะใช้เวลาน้อย MLP แต่มากกว่า RBF เพียงเล็กน้อย

ตารางที่ 2 ผลการทดลองจากการทดสอบด้วย Balance Scale Data Set

Testing fold	**VEBF**			**RBF**			**MLP**		
	Time (second)	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**
1	0.24	4	91.27	1.16	4	84.13	1.58	4	84.92
2	0.15	3	91.94	0.38	3	78.23	0.66	3	88.71
3	0.16	5	86.51	0.41	5	82.54	0.65	5	84.92
4	0.15	3	95.12	0.38	3	79.67	0.83	3	91.06
5	0.16	5	89.68	0.41	5	84.92	0.88	5	89.68
Average	0.17	4	**90.90**	0.55	4	81.90	0.92	4	87.86

ตารางที่ 3 ผลการทดลองจากการทดสอบด้วย Parkinson Data Set

Testing fold	VEBF			RBF			MLP		
	Time (second)	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**
1	2.12	82	92.11	1.60	82	76.32	52.30	82	94.74
2	1.74	82	87.18	0.79	82	66.67	48.24	82	76.92
3	1.76	92	85.00	0.94	92	75.00	65.43	92	82.50
4	1.74	90	92.50	0.90	90	77.50	65.41	90	77.50
5	1.74	87	84.21	0.86	87	68.42	76.54	87	81.58
Average	1.82	86.6	**88.20**	1.02	86.6	72.78	61.58	86.6	82.65

ตารางที่ 4 ผลการทดลองจากการทดสอบด้วย Zoo Data Set

Testing fold	VEBF			RBF			MLP		
	Time (second)	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**	**Time (second)**	**No. of neurons**	**Accuracy (%)**
1	0.10	11	100.00	0.82	11	78.95	1.29	11	94.74
2	0.05	11	100.00	0.13	11	75.00	0.48	11	60.00
3	0.04	11	95.24	0.09	11	80.95	0.51	11	85.71
4	0.04	10	100.00	0.19	10	75.00	0.41	10	80.00
5	0.05	11	95.24	0.21	11	90.48	0.46	11	95.24
Average	0.05	10.8	**98.10**	0.29	10.8	80.08	0.63	10.8	83.14

Zoo Data Set

Zoo Data Set มีข้อมูลทั้งหมด 101 ตัว โดยที่ข้อมูลแต่ละตัวประกอบด้วยแอตทริบิวทั้งหมด 17 แอตทริบิวและมีทั้งหมด 7 คลาสด้วยกัน ในการทดลองนี้เรากำหนดค่า δ = 0.5 สำหรับ VEBF และ δ = 10 สำหรับ RBF สำหรับ MLP เราจะรันโมเดลนี้จำนวน 10 ครั้งโดยเราจะเลือกครั้งที่ดีที่สุดของโมเดลนี้เป็นผลการทดลอง ในการทดลองนี้ VEBF เรียนรู้ข้อมูลเพียงรอบเดียวในขณะที่ RBF และ MLP เรียนรู้ข้อมูลหลายรอบ ผลการเปรียบเทียบดังแสดงในตารางที่ 4 จากตารางนี้เราจะเห็นว่า ค่าเฉลี่ยของความถูกต้อง (accuracy) ของ VEBF สูงกว่า RBF และ MLP นอกจากนี้เวลาที่ใช้ในการเรียนรู้ของ VEBF จะใช้เวลาน้อยกว่า RBF และ MLP

สรุป

บทความนี้นำเสนอโครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์ (VEBF) ซึ่งใช้ฟังก์ชันไฮเพอร์อัลลิปซอยด์ในการเรียนรู้ข้อมูลซึ่งฟังก์ชันนี้สามารถเรียนรู้ข้อมูลได้โดยการย้ายจุดศุนย์กลาง หมุน และขยายขนาดของฟังก์ชันเพื่อคลุมข้อมูลที่ต้องการเรียนรู้ โครงข่ายประสาทเทียมนี้สามารถที่จะเรียนรู้ข้อมูลเพียงแค่รอบเดียวในขณะที่โครงข่ายประสาทเทียมประเภทอื่นๆ จะต้องเรียนรู้ข้อมูลหลายรอบจึงจะสามารถเรียนรู้ข้อมูลได้ดี ซึ่งในขณะที่มีการเรียนรู้ค่าพารามิเตอร์ต่างๆ ของฟังก์ชันไฮเพอร์อัลลิปซอยด์จะถูกปรับจากข้อมูลที่เข้ามาใหม่เพียงข้อมูลเดียวจึงทำให้โครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์นี้สามารถที่จะเรียนรู้ข้อมูลใหม่ๆ ได้หลังจากที่โครงข่ายประสาทเทียมนี้ถูกนำไปใช้งานแล้วโดยไม่จำเป็นต้องเก็บข้อมูลเก่าที่ได้เรียนรู้ไปแล้วซึ่งโครงข่ายประสาทเทียมประเภทอื่นๆ ไม่สามารถทำได้ นอกจากนี้ประสิทธิภาพของโครงข่ายประสาทเทียมชนิดฟังก์ชันไฮเพอร์อัลลิปซอยด์ที่ได้จากการทดลองยังสูงกว่าโครงข่ายประสาทเทียม RBF และ MLP ซึ่ง RBF และ MLP เป็นโครงข่ายประสาทเทียมที่นิยมใช้กันในปัจจุบัน

กิตติกรรมประกาศ

งานวิจัยนี้ได้รับการสนับสนุนเงินทุนจากสำนักงานกองทุนสนับสนุนการวิจัย

เอกสารอ้างอิง

A. Asuncion & D. Newman. (2007) *UCI Machine Learning Repository*. School of Information and Computer Sciences. University. of California, Irvine.

Kadirkamanathan, V., & Niranjan, M.. (1993). A Function Estimation Approach to Sequential Learning with Neural Networks. *Neural Computation*, 954–975

Li, Y., Sundararajan, N., & Saratchandran, P. (2000). Analysis of Minimal Radial Basis Function Network Algorithm for Real-time Identification of Nonlinear Dynamic Systems. *IEEE Proceedings-Control Theory and Applications*.

Platt, J., A. (1991). Resource-allocating Network for Function Interpolation. *Neural Computation*, 213–225.

Saichon Jaiyen, Chidchanok Lursinsap & Suphakant Phimoltares. (2009. A Versatile Hyper-Ellipsoidal Basis Function for Function Approximation in High Dimensional Space. *6th International Symposium on Neural Networks*.

Yingwei, L., Sundararajan, N., & Saratchandran, P.. (1997). A Sequential Learning Scheme for Function Approximation Using Minimal Radial Basis Function (RBF) Neural Networks. *Neural Computatation*.