Computational Verb Neural Networks

Tao Yang

***Abstract*— When any attribute value in a conventional neural network is verbified, the result is a computational verb neural network(VNN). We can verbify inputs, outputs, biases, weights and many other attributes of conventional neural networks. In this paper, we reported two types of VNNs. The first one consists of computational verb inputs and numerical output. The second one consists of computational verb inputs and outputs. The learning algorithms for both types of VNNs were provided. The existence of solutions and continuities of single-layer feedforward type-I VNN were studied.** *Copyright © 2007 Yang's Scientific Research Institute, LLC. All rights reserved.*

***Index Terms*— Computational verb neural network, VNN, computational verb, single-layer feedforward type-I VNN.**

I. INTRODUCTION

ONE of the main drawbacks of neural networks is its lack of high-level representing power of the knowledge learnt and represented in their weights. One way to overcome this problem is to implement the representing power of natural languages into the structures of neural networks. The physical linguistics is the very framework to make natural language measurable and therefore, the best tool to implement the representing power of natural languages into the structures of neural networks. The author will present a neural network structure where computational verbs will be workhorses to bring the high-level knowledge representing power into the domain of measurements.

In [38], the author presented the learning algorithms for sets of computational verb rules. These learning algorithms can be readily transformed to fit into the configurations of computational verb neural networks. A computational verb neural network is the result of verbifying a conventional neural network. There are many ways to verbify the attributes of a conventional neural network. For example, one can verbify the inputs, outputs, weights, and biases of a conventional neural network. The author will study the following two types of computational verb neural networks.

1) Type-I computational verb neural network consists of verbified inputs and real outputs. All other attributes of are numerical.
2) Type-II computational verb neural network consists of verbified inputs and outputs. All other attributes of are numerical.

Manuscript received July 4, 2007; revised August 2, 2007; May 27, 2008.
Tao Yang, Department of Electronic Engineering, Xiamen University, Xiamen 361005, P.R. China. Department of Cognitive Economics, Department of Electrical Engineering and Computer Sciences, Yang's Scientific Research Institute, 1303 East University Blvd., #20882, Tucson, Arizona 85719-0521, USA. Email: taoyang@xmu.edu.cn,taoyang@yangsky.com,taoyang@yangsky.us.
Publisher Item Identifier S 1542-5908(07)10304-3/$20.00

The organization of this paper is as follows. In Section II, the brief history of computational verb theory will be given. In Section III, the structure of neuron of type-I computational verb neural networks and its learning algorithm will be presented. In Section IV, the structure of neuron of type-II computational verb neural networks and its learning algorithm will be given. In Section V, the existence of solutions and the continuities of single-layer feedforward type-I VNN will be studied. In Section VI, some concluding remarks will be presented.

II. A BRIEF HISTORY OF COMPUTATIONAL VERB THEORY

As the first paradigm shift for solving engineering problems by using verbs, the computational verb theory and physical linguistics have undergone a rapid growth since the birth of computational verb in the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley in 1997[8], [9]. The paradigm of implementing verbs in machines were coined as *computational verb theory*[22]. The building blocks of computational theory are *computational verbs*[17], [12], [10], [18], [23]. The relation between verbs and adverbs was mathematically defined in [11]. The logic operations between verb statements were studied in [13]. The applications of verb logic to verb reasoning were addressed in [14] and further studied in [22]. A logic paradox was solved based on verb logic[19]. The mathematical concept of set was generalized into verb set in[16]. Similarly, for measurable attributes, the number systems can be generalized into verb numbers[20]. The applications of computational verbs to predictions were studied in [15]. In [24] fuzzy dynamic systems were used to model a special kind of computational verb that evolves in a fuzzy space. The relation between computational verb theory and traditional linguistics was studied in [22], [25]. The theoretical basis of developing computational cognition from a unified theory of fuzzy and computational verb theories is the theory of the Unicogse that was studied in [25], [30]. The issues of simulating cognition using computational verbs were studied in [26]. A way to implementing feelings in machines was proposed based on grounded computational verbs and computational nouns in [32]. In [39] a new definition of the similarity between computational verbs was studied. The theory of computational verb has been taught in some university classrooms since 2005[1].

The latest active applications of computational verb theory are listed as follows.

1) Computational Verb Controllers. The applications of computational verbs to different kinds of control prob-

[1]Dr. G. Chen, EE 64152 - Introduction to Fuzzy Informatics and Intelligent Systems, Department of Electronic Engineering, City University of Hong Kong. Dr. Mahir Sabra, EELE 6306: Intelligent Control, Electrical and Computer Engineering Department, The Islamic University of Gaza.

lems were studied on different occassions[21], [22]. For the advanced applications of computational verbs to control problems, two papers reporting the latest advances had been published[28], [27]. The design of computational verb controller was also presented in a textbook in 2006[1].

2) Computational Verb Image Processing and Image Understanding. The recent results of image processing by using computational verbs can be found in[29]. The applications of computational verbs to image understanding can be found in [31].
3) Stock Market Modeling and Prediction based on computational verbs. The product of Cognitive Stock Charts[4] was based on the advanced modeling and computing reported in [33]. Applications of computational verbs was used to study the trends of stock markets known as Russell reconstruction patterns [34].

Computational verb theory has been successfully applied to many industrial and commercial products. Some of these products are listed as follows.

1) Visual Card Counters. The *YangSky-MAGIC* card counter[6], developed by Yang's Scientific Research Institute and Wuxi Xingcard Technology Co. Ltd., was the first visual card counter to use computational verb image processing technology to achieve high accuracy of card and paper board counting based on cheap webcams.
2) CCTV Automatic Driver Qualify Test System. The *DriveQfy* CCTV automatic driver qualify test system[7] was the first vehicle trajectory reconstruction and stop time measuring system using computational verb image processing technology.
3) Visual Flame Detecting System. The *FireEye* visual flame detecting system[2] was the first CCTV or webcam based flame detecting system, that works under color and black & white conditions, for surveillance and security monitoring system[36], [37].
4) Smart Pornographic Image and Video Detection Systems. The *PornSeer*[5] pornographic image and video detection systems are the first cognitive feature based smart porno detection and removal software.
5) Webcam Barcode Scanner. The *BarSeer*[3] webcam barcode scanner took advantage of the computational verb image processing to make the scan of barcode by using cheap webcam possible.
6) Cognitive Stock Charts. By applying computational verbs to the modeling of trends and cognitive behaviors of stock trading activities, cognitive stock charts can provide the traders with the "feelings" of stock markets by using simple and intuitive indexes.

III. TYPE-I COMPUTATIONAL VERB NEURAL NETWORKS

A type-I computational verb neural network(VNN)[2] takes implementations of computational verbs as inputs, compares the input verbs with some template verbs and output real numbers. Therefore, a type-I VNN has verbs as inputs and real numbers as outputs. In a type-I computational verb neural network, for input waveforms $x_i(t) \in \mathbb{R}, t \in [0,T], i = 1,\ldots,n$, there are template computational verbs V_i such that the computational verb similarity between $x_i(t)$ and the evolving function of V_i, $\mathcal{E}_i(t)$, will be weighted and results in the output $y \in \mathbb{R}$ of the neuron as follow.

$$y = f\left(\sum_{i=1}^{n} w_i S(x_i(t), \mathcal{E}_i(t)) - \theta\right), t \in [0,T] \tag{1}$$

where $S(\cdot,\cdot)$ is a computational verb similarity[35], $\theta \in \mathbb{R}$ is the bias of the neuron and the function $f(\cdot)$ is the transfer function for the neuron. Given m canonical computational verbs, $\{\widetilde{\mathcal{E}}_j\}_{j=1}^{m}$, we can represent a computational verb as

$$\mathcal{E}_i(t) = \sum_{j=1}^{m} \alpha_{ij} \widetilde{\mathcal{E}}_j(t) \tag{2}$$

where $\alpha_{ij} \in \mathbb{R}$ are constants and can be viewed as adverbs.

Assume that we have K set of training examples $\{u_{k1}(t),\ldots,u_{kn}(t),d_k\}_{k=1}^{K}$, where $u_{ki}(t), i = 1,\ldots,n$, are n input waveforms and d_k is the corresponding output. Our goal is to learn all computational verbs $\mathcal{E}_i(t)$ and bias θ from the training examples. For the kth training sample, we construct an output y_k as,

$$y_k = f\left(\sum_{i=1}^{n} w_i S(u_{ki}(t), \mathcal{E}_i(t)) - \theta\right), \tag{3}$$

based on which we construct the following error function.

$$E = \sum_{k=1}^{K} (y_k - d_k)^2. \tag{4}$$

The learning rules are given by

$$\begin{aligned} w_i(l+1) &= w_i(l) + \gamma_i^w(l)\Delta w_i(l), \\ \theta(l+1) &= \theta(l) + \gamma^\theta(l)\Delta\theta(l), \\ \alpha_{ij}(l+1) &= \alpha_{ij}(l) + \gamma_{ij}^\alpha(l)\Delta\alpha_{ij}(l), \quad l = 0,\ldots, \end{aligned} \tag{5}$$

where $\gamma_i^w(l) \in \mathbb{R}^+$, $\gamma^\theta(l) \in \mathbb{R}^+$ and $\gamma_{ij}^\alpha(l) \in \mathbb{R}^+, i = 1,\ldots,n; j = 1,\ldots,m$, are learning rates for training iteration l, and

$$\begin{aligned} \Delta w_i &= -\frac{\partial E}{\partial w_i} \\ &= -\frac{\partial \sum_{k=1}^{K} (y_k - d_k)^2}{\partial w_i} \\ &= -2\sum_{k=1}^{K} (y_k - d_k)\frac{\partial y_k}{\partial w_i} \\ &= -2\sum_{k=1}^{K} (y_k - d_k)\dot{f}\left(\sum_{h=1}^{n} w_h S(u_{kh}(t), \mathcal{E}_h(t)) - \theta\right) \\ &\quad \times S(u_{ki}(t), \mathcal{E}_i(t)), \end{aligned} \tag{6}$$

[2]Instead of using CVNN, we use VNN to stand for computational verb neural network to avoid clutter.

$$\begin{aligned}\Delta\theta &= -\frac{\partial E}{\partial \theta} \\ &= -\frac{\partial \sum_{k=1}^{K}(y_k-d_k)^2}{\partial \theta} \\ &= -2\sum_{k=1}^{K}(y_k-d_k)\frac{\partial y_k}{\partial \theta} \\ &= 2\sum_{k=1}^{K}(y_k-d_k)\dot{f}\left(\sum_{h=1}^{n} w_h S(u_{kh}(t),\mathcal{E}_h(t))-\theta\right),\end{aligned} \tag{7}$$

$$\begin{aligned}\Delta\alpha_{ij} &= -\frac{\partial E}{\partial \alpha_{ij}} \\ &= -\frac{\partial \sum_{k=1}^{K}(y_k-d_k)^2}{\partial \alpha_{ij}} \\ &= -2\sum_{k=1}^{K}(y_k-d_k)\frac{\partial y_k}{\partial \alpha_{ij}} \\ &= 2\sum_{k=1}^{K}(y_k-d_k)\dot{f}\left(\sum_{h=1}^{n} w_h S(u_{kh}(t),\mathcal{E}_h(t))-\theta\right) \\ &\quad \times w_i\frac{\partial S(u_{ki}(t),\mathcal{E}_i(t))}{\partial \alpha_{ij}}\end{aligned} \tag{8}$$

where $\frac{\partial S(u_{ki}(t),\mathcal{E}_i(t))}{\partial \alpha_{ij}}$ is given by Eq. (10) where $s_t(\cdot)$ is a *saturate function*[35] given by

$$s_t(x)=\frac{1}{1+e^{-x}},\ \text{and}\ \dot{s}_t(x)=\frac{e^{-x}}{(1+e^{-x})^2}. \tag{9}$$

IV. Type-II Computational Verb Neural Networks

A type-II VNN has verbs as inputs and verbs as outputs. A neuron in a type-II computational verb neural network is represented as

$$\mathcal{E}_y(t)=\sum_{i=1}^{n} w_i S(x_i(t),\mathcal{E}_i(t))\mathcal{E}_i(t)-\mathcal{E}_\theta(t),\quad t\in[0,T] \tag{11}$$

where $\mathcal{E}_y(t)$ is the evolving function of the output computational verb, $\mathcal{E}_\theta(t)\in\mathbb{R}$ is the verb bias of the neuron. Given m canonical computational verbs, $\{\widetilde{\mathcal{E}}_j\}_{j=1}^m$, computational verbs $\mathcal{E}_i(t), i=1,\dots,n$, are constructed as in Eq. (2) and $\mathcal{E}_\theta(t)$ is constructed as

$$\mathcal{E}_\theta(t)=\sum_{j=1}^{m}\alpha_j^\theta\widetilde{\mathcal{E}}_j(t) \tag{12}$$

where $\alpha_j^\theta\in\mathbb{R}$ are constants and can be viewed as adverbs.

Assume that we have K set of training examples $\{u_{k1}(t),\dots,u_{kn}(t),d_k(t)\}_{k=1}^K$, where $u_{ki}(t), i=1,\dots,n$, are n input waveforms and $d_k(t)$ is the corresponding output verb. Our goal is to learn all computational verbs $\mathcal{E}_i(t)$ and bias verb $\mathcal{E}_\theta(t)$ from the training examples. For the kth training sample, we construct an output $y_k(t)$ as,

$$y_k(t)=\sum_{i=1}^{n} w_i S(u_{ki}(t),\mathcal{E}_i(t))\mathcal{E}_i(t)-\mathcal{E}_\theta(t), \tag{13}$$

based on which we construct the following error function.

$$E=\sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]^2 dt. \tag{14}$$

The learning rules are given by

$$\begin{aligned} w_i(l+1) &= w_i(l)+\gamma_i^w(l)\Delta w_i(l), \\ \alpha_j^\theta(l+1) &= \alpha_j^\theta(l)+\gamma_j^\theta(l)\Delta\alpha_j^\theta(l), \\ \alpha_{ij}(l+1) &= \alpha_{ij}(l)+\gamma_{ij}^\alpha(l)\Delta\alpha_{ij}(l),\quad l=0,\dots,\end{aligned} \tag{15}$$

where $\gamma_i^w(l)\in\mathbb{R}^+$, $\gamma_j^\theta(l)\in\mathbb{R}^+$ and $\gamma_{ij}^\alpha(l)\in\mathbb{R}^+, i=1,\dots,n; j=1,\dots,m$, are learning rates for training iteration l, and

$$\begin{aligned}\Delta w_i &= -\frac{\partial E}{\partial w_i} \\ &= -\frac{\partial \sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]^2 dt}{\partial w_i} \\ &= -2\sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]\frac{\partial y_k(t)}{\partial w_i}dt \\ &= -2\sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]S(u_{ki}(t),\mathcal{E}_i(t))\mathcal{E}_i(t),\end{aligned} \tag{16}$$

$$\begin{aligned}\Delta\alpha_j^\theta &= -\frac{\partial E}{\partial \alpha_j^\theta} \\ &= -\frac{\partial \sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]^2 dt}{\partial \alpha_j^\theta} \\ &= -2\sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]\frac{\partial y_k(t)}{\partial \alpha_j^\theta}dt \\ &= 2\sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]\widetilde{\mathcal{E}}_j(t),\end{aligned} \tag{17}$$

$$\begin{aligned}\Delta\alpha_{ij} &= -\frac{\partial E}{\partial \alpha_{ij}} \\ &= -\frac{\partial \sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]^2 dt}{\partial \alpha_{ij}} \\ &= -2\sum_{k=1}^{K}\int_0^T [y_k(t)-d_k(t)]\frac{\partial y_k(t)}{\partial \alpha_{ij}}dt\end{aligned} \tag{18}$$

$$\begin{aligned}\frac{\partial S(u_{ki}(t),\mathcal{E}_i(t))}{\partial \alpha_{ij}} &= \frac{\partial\left\{1-\sqrt{\frac{1}{T}\int_0^T\left[s_t\left(u_{ki}(t)\right)-s_t\left(\sum_{j=1}^m \alpha_{ij}\widetilde{\mathcal{E}}_j(t)\right)\right]^2 dt}\right\}}{\partial \alpha_{ij}} \\ &= \frac{\int_0^T\left[s_t\left(u_{ki}(t)\right)-s_t\left(\sum_{j=1}^m \alpha_{ij}\widetilde{\mathcal{E}}_j(t)\right)\right]\dot{s}_t\left(\sum_{j=1}^m \alpha_{ij}\widetilde{\mathcal{E}}_j(t)\right)\widetilde{\mathcal{E}}_j(t)dt}{T\sqrt{\frac{1}{T}\int_0^T\left[s_t\left(u_{ki}(t)\right)-s_t\left(\sum_{j=1}^m \alpha_{ij}\widetilde{\mathcal{E}}_j(t)\right)\right]^2 dt}}. \end{aligned} \tag{10}$$

where $\frac{\partial y_k(t)}{\partial \alpha_{ij}}$ is given by

$$\begin{aligned}\frac{\partial y_k(t)}{\partial \alpha_{ij}} &= w_i\frac{\partial S(u_{ki}(t),\mathcal{E}_i(t))}{\partial \alpha_{ij}}\mathcal{E}_i(t) \\ &\quad +w_i S(u_{ki}(t),\mathcal{E}_i(t))\frac{\partial \mathcal{E}_i(t)}{\partial \alpha_{ij}} \\ &= w_i\mathcal{E}_i(t)\frac{\partial S(u_{ki}(t),\mathcal{E}_i(t))}{\partial \alpha_{ij}} \\ &\quad +w_i S(u_{ki}(t),\mathcal{E}_i(t))\widetilde{\mathcal{E}}_j(t) \end{aligned} \tag{19}$$

where $\frac{\partial S(u_{ki}(t),\mathcal{E}_i(t))}{\partial \alpha_{ij}}$ is given by Eq. (10).

V. Single-Layer Feedforward Type-I VNN

The structure of single-layer feedforward type-I VNN is shown in Fig. 1. We assume that there are n input nodes, m hidden nodes and 1 output node. The weight $w_{ij}, i = 1,\dots,n; j = 1,\dots,m$, connects between the ith input node and the jth hidden node. The weight v_j connect the jth hidden node to the only output node.

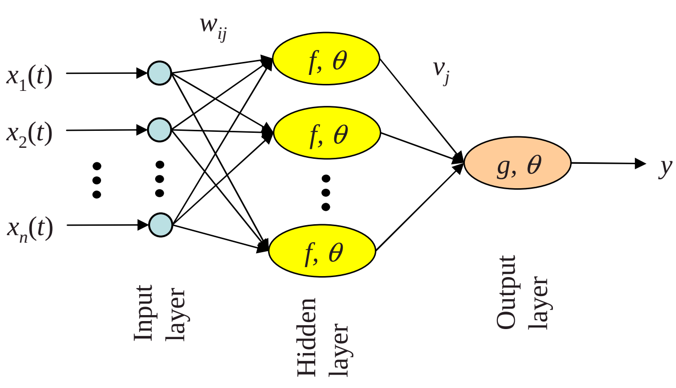

Fig. 1. The structure of single-layer feedforward type-I VNN.

The input-output relation of this VNN is given by

$$y = g\left(\sum_{j=1}^m v_j f\left(\sum_{i=1}^n w_{ij}S(x_i(t),\mathcal{E}_i(t))-\theta_j\right)-\theta\right) \tag{20}$$

where each computational verb is constructed based on p canonical computational verbs

$$\mathcal{E}_i(t) = \sum_{h=1}^p \alpha_{ih}\widetilde{\mathcal{E}}_h(t). \tag{21}$$

Remark. If in Eq. (20) we choose a computational verb similarity and a set of computational verbs such that $S(x_i(t),\mathcal{E}_i(t)) = x_i(t)$ and only consider one sample point of the input waveforms, then VCNN (20) degenerates into the following conventional feedforward neural network with one hidden layer.

$$y = g\left(\sum_{j=1}^m v_j f\left(\sum_{i=1}^n w_{ij}x_i-\theta_j\right)-\theta\right). \tag{22}$$

The general conclusion is that *all* conventional neural networks can be verbified into computational verb neural networks.

Theorem 1. Assume that $\{\boldsymbol{u}_k(t), d_k\}, k = 1,\dots,K$, are K training samples that are independent and have the same distribution. $\boldsymbol{u}_k(t) \in \underbrace{C[0,T]\times C[0,T]\dots,\times C[0,T]}_{n}$ and $d_k \in \mathbb{R}$. $g(\cdot)$ is monotonic increasing. Then for any a given approximation error $\varepsilon \in \mathbb{R}^+$, the VNN (20) can approximate the input-output relation defined by the training set if the number of hidden nodes, m, is big enough.

Proof. The kth training sample is

$$(\boldsymbol{u}_k(t), d_k) = (u_1^{(k)},\dots,u_n^{(k)}, d_k). \tag{23}$$

It follows from Eq. (20) that the kth training sample satisfies Eq. (24). Then for the training samples $\{\boldsymbol{u}_k(t), d_k\}, k = 1,\dots,K$ and a given error ε Eq. (25) is satisfied. In Eq. (25) w_{ij}, v_j, α_{ih} and θ are adjustable parameters to be determined. Therefore, Eq. (25) defines a region in the parameter space to include the solution to the input-output relation defined by the training samples. The existence of such a solution is guaranteed by principles of nonlinear programming for each fixed θ when we choose the total number of parameters no less than the number of training samples; namely, $nm+m+np \geq K$.

■

Remark. It follows from the conclusion in [38] that the set of nonlinear equations can be solved numerically. The details can be found from Sec. IV of [38].

Theorem 2. For VNN (20), assume that $\{\boldsymbol{x}_k(t), y_k\}, k = 1, 2$, are 2 input-output pairs where $\boldsymbol{x}_k(t) \in \underbrace{C[0,T]\times C[0,T]\dots,\times C[0,T]}_{n}$ and $y_k \in \mathbb{R}$, and assume that $f(\cdot)$ and $g(\cdot)$ are continuous, then for any $\varepsilon > 0$, there

$$
\begin{aligned}
d_k &= g\left(\sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n} w_{ij} S\left(u_i^{(k)}(t), \mathcal{E}_i(t)\right) - \theta_j\right) - \theta\right) \\
&= g\left(\sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n} w_{ij} S\left(u_i^{(k)}(t), \sum_{h=1}^{p} \alpha_{ih} \widetilde{\mathcal{E}}_h(t)\right) - \theta_j\right) - \theta\right).
\end{aligned} \tag{24}
$$

$$
\left\{
\begin{array}{c}
\sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n} w_{ij} S\left(u_i^{(1)}(t), \sum_{h=1}^{p} \alpha_{ih} \widetilde{\mathcal{E}}_h(t)\right) - \theta_j\right) = g^{-1}(d_1 + \theta), \\
\sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n} w_{ij} S\left(u_i^{(2)}(t), \sum_{h=1}^{p} \alpha_{ih} \widetilde{\mathcal{E}}_h(t)\right) - \theta_j\right) = g^{-1}(d_2 + \theta), \\
\vdots \\
\sum_{j=1}^{m} v_j f\left(\sum_{i=1}^{n} w_{ij} S\left(u_i^{(K)}(t), \sum_{h=1}^{p} \alpha_{ih} \widetilde{\mathcal{E}}_h(t)\right) - \theta_j\right) = g^{-1}(d_K + \theta).
\end{array}
\right. \tag{25}
$$

exists such a $\delta > 0$ that when $\|\boldsymbol{x}_1(t) - \boldsymbol{x}_2(t)\| < \delta$, we have $|y_1 - y_2| < \varepsilon$.

Proof. To avoid clutter, let us use the following notation.

$$
\upsilon_j^{(k)} \triangleq \sum_{i=1}^{n} w_{ij} S\left(x_i^{(k)}(t), \mathcal{E}_i(t)\right) - \theta_j, \quad k = 1, 2. \tag{26}
$$

Since $g(\cdot)$ is continuous, for any $\varepsilon > 0$, there exists such a $\delta_1 > 0$ that when

$$
\begin{aligned}
&\left|\left(\sum_{j=1}^{m} v_j f\left(\upsilon_j^{(1)}\right) - \theta\right) - \left(\sum_{j=1}^{m} v_j f\left(\upsilon_j^{(2)}\right) - \theta\right)\right| \\
= &\left|\sum_{j=1}^{m} v_j \left[f\left(\upsilon_j^{(1)}\right) - f\left(\upsilon_j^{(2)}\right)\right]\right| < \delta_1,
\end{aligned} \tag{27}
$$

we have $|y_1 - y_2| < \varepsilon$. The next step is to prove that for $\delta_1 > 0$, there exists such a $\delta > 0$ that when $\|\boldsymbol{x}_1(t) - \boldsymbol{x}_2(t)\| < \delta$, Eq. (27) satisfies. Since $f(\cdot)$ is continuous, for $\delta_1 > 0$, there exist such a $\delta_2 > 0$ that when $|\upsilon_j^{(1)} - \upsilon_j^{(2)}| < \delta_2$, we have

$$
|f(\upsilon_j^{(1)}) - f(\upsilon_j^{(2)})| < \frac{\delta_1}{m \max_{j=0}^{m} |v_j|}. \tag{28}
$$

Therefore, if we choose a $\delta > 0$ satisfying

$$
\begin{aligned}
&\left|\sum_{j=1}^{m} v_j \left[f\left(\upsilon_j^{(1)}\right) - f\left(\upsilon_j^{(2)}\right)\right]\right| \\
\leq &\sum_{j=1}^{m} |v_j| \left|f\left(\upsilon_j^{(1)}\right) - f\left(\upsilon_j^{(2)}\right)\right| \\
\leq &\; m \max_{j=0}^{m} |v_j| \frac{\delta_1}{m \max_{j=0}^{m} |v_j|} = \delta_1.
\end{aligned} \tag{29}
$$

■

VI. Concluding Remarks

The results presented in this paper revealed the brand new opening to a new world of learning from dynamical data or time series. The potentials of applying VNN to financial world are tremendous if we view the records of intra-day stock prices as different computational verbs and transform the data-mining problem of intra-day stock prices into a learning problem of VNN. Needless to say, many similar cases exist in applications where dynamical observations are of regular time windows and the knowledge discovery from these data becomes impossible if we only view each observations isolated, without considering the embedded natural time-line in them. It is the author's vision that within ten years, computational neural network alone will become a comprehensive discipline in the family of computational verb theory.

References

[1] Guanrong Chen and Trung Tat Pham. *Introduction to Fuzzy Systems*. Chapman & Hall/CRC, November 2005. ISBN:1-58488-531-9.

[2] Yang's Scientific Research Institute LLC. **FireEye** *Visual Flame Detecting Systems*. http://www.yangsky.us/products/flamesky/index.htm, http://www.yangsky.com/products/flamesky/index.htm, 2005.

[3] Yang's Scientific Research Institute LLC. **BarSeer** *Webcam Barcode Scanner*. http://www.yangsky.us/demos/barseer/barseer.htm, http://www.yangsky.com/demos/barseer/barseer.htm, 2006.

[4] Yang's Scientific Research Institute LLC. *Cognitive Stock Charts*. http://www.yangsky.us/products/stock/, http://www.yangsky.com/products/stock/, 2006.

[5] Yang's Scientific Research Institute LLC. **PornSeer** *Pornographic Image and Video Detection Systems*. http://www.yangsky.us/products/dshowseer/porndetection/PornSeePro.htm, http://www.yangsky.com/products/dshowseer/porndetection/PornSeePro.htm, 2006.

[6] Yang's Scientific Research Institute LLC. and Wuxi Xingcard Technology Ltd. **YangSky-MAGIC** *Visual Card Counters*. http://www.yangsky.us/products/cardsky/cardsky.htm, http://www.yangsky.com/products/cardsky/cardsky.htm, 2004.

[7] Yang's Scientific Research Institute LLC. and Chinese Traffic Management Research Institute of the Ministry of Public Security(TMRI-China). **DriveQfy** *Automatic CCTV Driver Qualify Testing Systems*. http://www.yangsky.us/products/driveqfy/driveqfy.htm, http://www.yangsky.com/products/driveqfy/driveqfy.htm, 2005.

[8] T. Yang. Verbal paradigms—Part I: Modeling with verbs. Technical Report Memorandum No. UCB/ERL M97/64, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, CA 94720, 9 Sept. 1997. page 1-15.

[9] T. Yang. Verbal paradigms—Part II: Computing with verbs. Technical Report Memorandum No. UCB/ERL M97/66, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, CA 94720, 18 Sept. 1997. page 1-27.

[10] T. Yang. Computational verb systems: Computing with verbs and applications. *International Journal of General Systems*, 28(1):1–36, 1999.

[11] T. Yang. Computational verb systems: Adverbs and adverbials as modifiers of verbs. *Information Sciences*, 121(1-2):39–60, Dec. 1999.

[12] T. Yang. Computational verb systems: Modeling with verbs and applications. *Information Sciences*, 117(3-4):147–175, Aug. 1999.

[13] T. Yang. Computational verb systems: Verb logic. *International Journal of Intelligent Systems*, 14(11):1071–1087, Nov. 1999.

[14] T. Yang. Computational verb systems: A new paradigm for artificial intelligence. *Information Sciences—An International Journal*, 124(1-4):103–123, 2000.

[15] T. Yang. Computational verb systems: Verb predictions and their applications. *International Journal of Intelligent Systems*, 15(11):1087–1102, Nov. 2000.

[16] T. Yang. Computational verb systems: Verb sets. *International Journal of General Systems*, 20(6):941–964, 2000.

[17] T. Yang. *Advances in Computational Verb Systems*. Nova Science Publishers, Inc., Huntington, NY, May 2001. ISBN 1-56072-971-6.

[18] T. Yang. Computational verb systems: Computing with perceptions of dynamics. *Information Sciences*, 134(1-4):167–248, Jun. 2001.

[19] T. Yang. Computational verb systems: The paradox of the liar. *International Journal of Intelligent Systems*, 16(9):1053–1067, Sept. 2001.

[20] T. Yang. Computational verb systems: Verb numbers. *International Journal of Intelligent Systems*, 16(5):655–678, May 2001.

[21] T. Yang. *Impulsive Control Theory*, volume 272 of *Lecture Notes in Control and Information Sciences*. Spinger-Verlag, Berlin, Aug. 2001. ISBN 354042296X.

[22] T. Yang. *Computational Verb Theory: From Engineering, Dynamic Systems to Physical Linguistics*, volume 2 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Research Institute, Tucson, AZ, Oct. 2002. ISBN:0-9721212-1-8.

[23] T. Yang. Computational verb systems: Verbs and dynamic systems. *International Journal of Computational Cognition*, 1(3):1–50, Sept. 2003.

[24] T. Yang. *Fuzzy Dynamic Systems and Computational Verbs Represented by Fuzzy Mathematics*, volume 3 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Press, Tucson, AZ, Sept. 2003. ISBN:0-9721212-2-6.

[25] T. Yang. *Physical Linguistics: Measurable Linguistics and Duality Between Universe and Cognition*, volume 5 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Press, Tucson, AZ, Dec. 2004.

[26] T. Yang. Simulating human cognition using computational verb theory. *Journal of Shanghai University(Natural Sciences)*, 10(s):133–142, Oct. 2004.

[27] T. Yang. Architectures of computational verb controllers: Towards a new paradigm of intelligent control. *International Journal of Computational Cognition*, 3(2):74–101, June 2005 [available online at $\underline{http://www.YangSky.com/ijcc/ijcc32.htm}$, $\underline{http://www.YangSky.us/ijcc/ijcc32.htm}$].

[28] T. Yang. Applications of computational verbs to the design of P-controllers. *International Journal of Computational Cognition*, 3(2):52–60, June 2005 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc32.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc32.htm}$].

[29] T. Yang. Applications of computational verbs to digital image processing. *International Journal of Computational Cognition*, 3(3):31–40, September 2005 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc33.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc33.htm}$].

[30] T. Yang. Bridging the Universe and the Cognition. *International Journal of Computational Cognition*, 3(4):1–15, December 2005 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc34.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc34.htm}$].

[31] T. Yang. Applications of computational verbs to effective and realtime image understanding. *International Journal of Computational Cognition*, 4(1):49–67, March 2006 [available online at $\underline{http://www.YangSky.com/ijcc/ijcc41.htm}$, $\underline{http://www.YangSky.us/ijcc/ijcc41.htm}$].

[32] T. Yang. Applications of computational verbs to feeling retrieval from texts. *International Journal of Computational Cognition*, 4(3):28–45, September 2006 [available online at $\underline{http://www.YangSky.com/ijcc/ijcc43.htm}$, $\underline{http://www.YangSky.us/ijcc/ijcc43.htm}$].

[33] T. Yang. Applications of computational verbs to cognitive models of stock markets. *International Journal of Computational Cognition*, 4(2):1–13, June 2006 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc42.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc42.htm}$].

[34] T. Yang. Applications of computational verbs to the study of the effects of Russell's annual index reconstitution on stock markets. *International Journal of Computational Cognition*, 4(3):1–8, September 2006 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc43.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc43.htm}$].

[35] T. Yang. Distances and similarities of saturated computational verbs. *International Journal of Computational Cognition*, 4(4):62–77, December 2006 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc44.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc44.htm}$].

[36] T. Yang. Accurate video flame-detecting system based on computational verb theory. AS *Installer*, (42):154–157, August 2007. (in Chinese).

[37] T. Yang. Applications of computational verb theory to the design of accurate video flame-detecting systems. *International Journal of Computational Cognition*, 5(3):25–42, September 2007 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc53.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc53.htm}$].

[38] T. Yang. Learning computational verb rules. *International Journal of Computational Cognition*, 5(3):43–56, September 2007 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc53.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc53.htm}$].

[39] Jian Zhang and Minrui Fei. Determination of verb similarity in computational verb theory. *International Journal of Computational Cognition*, 3(3):74–77, September 2005 [available online at $\underline{http://www.YangSky.us/ijcc/ijcc33.htm}$, $\underline{http://www.YangSky.com/ijcc/ijcc33.htm}$].