Neurocomputing ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

# Modeling word perception using the Elman network

Cheng-Yuan Liou *, Jau-Chi Huang, Wen-Chie Yang

*Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, ROC*

ARTICLE INFO

*Keywords:*
Word perception
Compositional representation
Authorship
Stylistic similarity
Categorization
Semantic search
Elman network
Linguistic analysis
Personalized code
Content addressable memory
Polysemous word

ABSTRACT

This paper presents an automatic acquisition process to acquire the semantic meaning for the words. This process obtains the representation vectors for stemmed words by iteratively improving the vectors, using a trained Elman network. Experiments performed on a corpus composed of Shakespeare's writings show its linguistic analysis and categorization abilities.



## 1. Introduction

The semantic meaning of a word or a word sequence is often nonquantifiable. A central problem in the analysis of such a sequence is determining how to effectively encode and extract its contents. Existing analyses are primarily based on certain statistical linguistic features [2,7,23–25]. The semantic search [26] constructs a mathematical model that analyzes semantic features and creates a semantic operation space. It sorts data according to the semantic meaning of the devolved requests. Nevertheless, there are difficulties in implementing the model. The task of constructing a prime semantic space is extremely expensive and complex, because experienced linguists are needed to analyze huge numbers of words. This paper presents an automatic encoding process to accomplish this task.

Both the frequencies and the temporal sequence of words carry semantic meaning. When one listens to a talk or reads an article, one should get information from both isolated words and their sequences. Complying with temporal information [18,16], our approach employs the Elman network [4,3,13], which works well with temporal sequences, as an encoding mechanism [15,17,12]. This network can extract and accommodate the rich syntax grammars associated with each word in sentence sequences [9].

The automatic encoding method will be presented in Section 2.1. The semantic search [26] and its notations will be reviewed in Section 2.2. A method for dealing with polysemous words will be discussed in the third section. Applications to literary works will be presented in these two sections.

## 2. Encoding method

Semantic meaning comes from a sequence of words. It is sequential and temporal. We employ the Elman network to extract the meaning from sentence sequences.

### 2.1. The Elman network

The network is a single recursive network that has a context layer as an inside self-referenced layer, see Fig. 1. During operation, both current input from the input layer and previous state of the hidden layer saved in the context layer activate the hidden layer. Note that there exists an energy function associated with the hidden layer, context layer, and input layer [16,12]. With successive training, the connection weights can load the temporal relations in the training word sequences.

The context layer carries the memory. The hidden layer activates the output layer and refreshes the context layer with the current state of the hidden layer. The back-propagation learning algorithm [21] is commonly employed to train the weights in order to reduce the difference between the output of the output layer and its desired output. Note that in this paper, the threshold value of every neuron in the network is set to zero. Let $L_o$, $L_h$, $L_c$, and $L_i$ be the number of neurons in the output layer, the hidden layer, the context layer, and the input layer, respectively. In the Elman network, $L_h$ is equal to $L_c$, that is, $L_h = L_c$. In this paper,

* Corresponding author.
*E-mail address:* cyliou@csie.ntu.edu.tw (C.-Y. Liou).

0925-2312/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2008.04.030

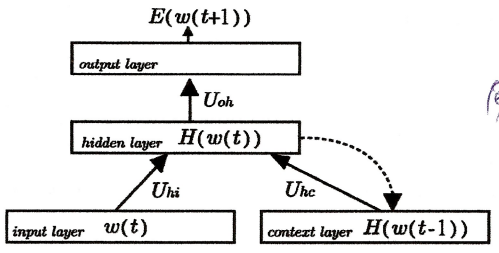


**Fig. 1.** The Elman network.

the number of neurons in the input layer is equal to that in the output layer and is also equal to the number of total features, that is, $R = L_o = L_i$.

Let $\{w_n, n = 1{-}N\}$ be the code set of different words in a corpus. The corpus, $D$, contains a collection of all given sentences. During training, a sentence is randomly selected from the corpus and fed to the network sequentially, word by word, starting from the first word of the sentence. Let $|D|$ be the total length of all the sentences in the corpus, $D$. $|D|$ is the total number of words in $D$. Usually, $|D|$ is several times the number of different words in the corpus, so $|D| > N$. Initially, $t = 0$, and all weights are set to small random numbers. Let $w(t)$ be the current word in a selected sentence at time $t$, i.e.,

$$w(t) \in D, \quad w(t) \in \{w_n, n = 1{-}N\}, \quad t = 1{-}T, \tag{1}$$

where $w(T)$ is the last word of a training epoch. In this paper, we set $T = 4|D|$ in one epoch. This means that in each epoch, we use all the sentences in the corpus to train the Elman network four times. Let the three weight matrices between layers be $U_{oh}, U_{hc}$, and $U_{hi}$, where $U_{oh}$ is an $L_h$ by $L_o$ matrix, $U_{hc}$ is an $L_c$ by $L_h$ matrix, and $U_{hi}$ is an $L_i$ by $L_h$ matrix, as shown in Fig. 1. The output vector of the hidden layer is denoted as $H(w(t))$ when $w(t)$ is fed to the input layer. $H(w(t))$ is an $L_h$ by 1 column vector with $L_h$ elements. Let $E(w(t+1))$ be the output vector of the output layer when $w(t)$ is fed to the input layer. $E(w(t+1))$ is an $L_o$ by 1 column vector.

The function of the network is

$$H(w(t)) = \varphi(U_{hi}w(t) + U_{hc}H(w(t-1)), \tag{2}$$

where $\varphi$ is a sigmoid activation function that operates on each element of a vector [21]. We use the sigmoid function $\varphi(x) = 1.7159 * \tanh(x * 2/3)$ for all neurons in the network. This function gives a value roughly between $+1.7159$ and $-1.7159$. In Elman's experiment, the first step is to update the weights, $U_{hi}$, $U_{hc}$ and $U_{oh}$, through training. The second step is to encode words with a tree structure. All the attempts are aimed at minimizing the error between the network outputs and the desired outputs to satisfy the prediction

$$w(t+1) \approx E(w(t+1)) = \varphi(U_{oh}H(w(t))). \tag{3}$$

From a trained network, Elman uses a measure to locate the relationships among words. Before training, he prepares a list of words without inflections or rules. We will follow his preparation on words. All words are coded with given lexical codes. The available semantic combination is a fixed syntax (*Noun + Verb + Noun*). Elman generates sentences and temporal word sequences with this syntax grammar and collects all the sentences in a training corpus, $D$, for training a network [3]. The network has equal numbers of neuron units in its four layers. This network is trained sequentially by using the generated sentences. Elman defines the desired outputs as the sufficient words. For example, when the first word 'man' in a generated sentence 'men sleep' is used as the input, the sufficient word 'sleep' is its desired output. The network is trained to predict the following word. This training process continues until the variation of weights cannot be reduced. After training, Elman inputs the generated sentences again and collects all the output vectors of the hidden layer corresponding to each individual word in a separate set, $s_n^E = \{H(w(t)) | w(t) = w_n\}$. Then he obtains new code, $w_n^E$, for the $n$th word by averaging all vectors in set $s_n^E$:

$$w_n^E = \frac{1}{|s_n^E|} \sum_{\substack{w(t)=w_n \\ w(t)\in D}} H(w(t)), \quad n = 1{-}N, \tag{4}$$

where $|s_n^E|$ is the total number of vectors inside set $s_n^E$. Then, he constructs a tree for the words based on their new codes, $w_n^E$, to explore the relationships among the words.

Note that there exist extra temporal relations in the generated sentences with the simple fixed syntax *Noun + Verb + Noun*. For example, when $w(t)$ is a noun, $w(t+2)$ is most likely a noun, and when $w(t)$ is a verb, $w(t+3)$ is most likely a verb. These extra relations are additive to resolve the dichotomous classification between the verb and noun. A compound sentence may not possess such extra relations, and may not have additive resolutions.

### 2.2. The semantic search

The semantic search constructs a semantic model and a semantic measure. A manually designed semantic code set is used in the model. It assumes that the encoding task will be assigned to linguistics experts. It is hypothesized in advance that one can build a raw semantic matrix, $W$, as

$$W_{R\times N} \equiv [w_1 \ w_2 \ \dots \ w_N]_{R\times N}, \tag{5}$$

where $w_n$, $n = 1{-}N$, denotes the code of the $n$th stemmed word and $N$ denotes the total number of different words. A code of a word is a column vector with $R$ features as its elements:

$$w_n \equiv [w_{1n}, w_{2n}, \dots, w_{Rn}]^T. \tag{6}$$

To manage abstract features, one may use the orthogonal space configured by the characteristic decomposition of the matrix, $WW^T$:

$$W_{R\times N}W_{R\times N}^T = F_{R\times R}^T \begin{bmatrix} \lambda_1 & 0 & \cdot & 0 \\ 0 & \lambda_2 & 0 & \cdot \\ \cdot & 0 & \cdot & 0 \\ 0 & \cdot & 0 & \lambda_R \end{bmatrix}_{R\times R} F_{R\times R}, \tag{7}$$

where

$$F_{R\times R} \equiv [f_1, f_2, \dots, f_R]_{R\times R}, \quad \|f_r\| = 1,$$

and

$$\lambda_r \geqslant \lambda_{r+1}, \quad r = 1{-}R. \tag{8}$$

Since $WW^T$ is a symmetric matrix, all its eigenvalues are real and nonnegative numbers. Each eigenvalue $\lambda_i$ equals the variance of the $N$ projections of the codes on the $i$th eigenvector, $f_i$, that is, $\lambda_i = \sum_{n=1}^{N}((w_n \cdot f_i))^2$.

#### 2.2.1. Multi-dimensional scaling (MDS) space

We select a set of $R^s$ eigenvectors, $\{f_r, r = 1{-}R^s\}$, from the $R$ eigenvectors to build a reduced feature space:

$$F_{R\times R^s}^s \equiv [f_1, f_2, \dots, f_{R^s}]_{R\times R^s}. \tag{9}$$

This selection is based on the distribution of the projections of the codes on each eigenvector. An ideal distribution is an even distribution with large variance. We select those eigenvectors, $\{f_r, r = 1{-}R^s\}$, that have large eigenvalues. The MDS space is

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030

$$MDS \equiv span(F^S). \tag{10}$$

These selected features are independent and significant. The new code of each word in this space is

$$w_n^s = F^{s^T} w_n \tag{11}$$

or

$$W_{R\times N}^s = F^{s^T} W_{R\times N}. \tag{12}$$

### 2.2.2. Representative vector of a whole document

A document, denoted as $D$, usually contains more than one word. A representative vector should contain the semantic meaning of the whole document. Two such measures are defined. They are the peak-preferred measure,

$$v_D^a = [w_1^a, w_2^a, \ldots, w_R^a]^T,$$

where

$$w_r^a = \max_{w_n^s \in D} \|w_{rn}^s\|, \quad r = 1-R,$$

and the average-preferred measure,

$$v_D^b = \sum_{w_n^s \in D} w_n^s = [w_1^b, w_2^b, \ldots, w_R^b]^T,$$

where

$$w_r^b = \sum_{w_n^s \in D} w_{rn}^s, \quad r = 1-R. \tag{13}$$

The magnitude is normalized as follows:

$$v_D = \|v_D^b\|^{-1} v_D^b. \tag{14}$$

The normalized measure, $v_D$, is used here to represent the whole document. A representative vector, $v_Q$, for a whole query can be obtained similarly by using Eqs. (13), (14).

### 2.2.3. Relation comparison

The relation score is defined as follows:

$$RS_Q(D) = \frac{\langle v_D, v_Q \rangle}{\|v_D\| \times \|v_Q\|} = \langle v_D, v_Q \rangle. \tag{15}$$

## 2.3. Iterative re-encoding

Since Elman's encoding method for the sentences generated with simple fixed syntax, *Noun* + *Verb* + *Noun*, cannot be applied appropriately to more complex sentences, we modified his method. In our approach, each word initially has a random lexical code, $w_n^{j=0} = [w_{n1}, w_{n2}, \ldots, w_{nR}]^T$. After the $j$th training epoch, a new raw code is calculated as follows:

$$w_n^{raw} = \frac{1}{|s_n|} \sum_{\substack{w(t)=w_n \\ w(t)\in D}} \varphi(U_{oh} H(w(t-1))), \quad n = 1-N, \tag{16}$$

where $|s_n|$ is the total number of words in a set, $s_n$. This set contains all the predictions for the word, $w_n$, based on all its precedent words, $s_n = \{\varphi(U_{oh}H(w(t-1)))|w(t) = w_n,$ and $w(t) \in D\}$. This equation has a form slightly different from that in (4). Namely, we directly average all the prediction vectors for a specific word. The hidden layer may have a flexible number of neurons in our modified method. Note that there exist other promising methods to obtain a set of hidden representation codes from the set $s_n$, such as the self-organizing map [10], the multi-layer perceptron [21] and the SIR method [17]. After each epoch, all the codes are normalized with the following two equations:

$$W_{R\times N}^{ave} = W_{R\times N}^{raw} - \frac{1}{N} W_{R\times N}^{raw} \begin{bmatrix} 1 & \ldots & 1 \\ \vdots & 1 & \vdots \\ 1 & \ldots & 1 \end{bmatrix}_{N\times N},$$

$$w_n^j = w_n^{nom} = \|w_n^{ave}\|^{-1} w_n^{ave}, \tag{17}$$

where

$$\|w_n\| = (w_n^T w_n)^{0.5}, \quad n = 1-N. \tag{18}$$

This normalization can prevent a diminished solution, $\{\|w_n\| \sim 0, n = 1-N\}$, derived by the back-propagation algorithm.

In summary, the process starts with a set of random lexical codes for all of the stemmed words in a specific corpus. In each epoch, we use all the sentences in the corpus to train [16–18,12–14,21] an Elman network four times. We then compute the new code, $w_n^j$, for each word using Eqs. (16)–(18). The training phase is stopped (finished) at the $j$th epoch when there is no significant code difference between two successive epochs. We expect that such iterative encoding can extract certain salient features, in addition to word frequencies, in the sentence sequence which contains the writing style of the author or work. This writing behavior is unlikely to be consciously manipulated by the author and may serve as a robust stylistic signature. The trained code after the $j$th epoch, $w_n = w_n^j = [w_{1n}, w_{2n}, \ldots, w_{Rn}]^T$, which is a vector with $R$ features, is used in the semantic matrix $W_{R\times N}$ in (5) and the average-preferred measure (13). The normalization step (14) and the relation score (15) are then calculated based on this measure.

### 2.3.1. Example of literature categorization

In this experiment, we test the method's ability to classify 36 plays written by William Shakespeare (WS).

The words were prepared according to Elman's approach. We removed the functional words, such as articles, conjunctions, be-verbs, and even some words like 'take,' 'get,' 'you,' 'I,' etc. because they cause noises across different semantic categories. We then stemmed [6,20] each word as deep as possible to expose clean relations among words. Note that the degree of stemming is a much discussed lexical issue [3]. For example, it is not clear whether to stem the structure: '-ness,' '-able,' '-tion'. A trained code set was generated using a training corpus that contained the 36 works. We considered each play as the query input and computed the relation score between this query and one other play. Fig. 2 shows the relation tree of the 36 plays.

This tree was constructed by applying the methods in [5,8,22] to 630 scores of pairs of two plays. We also include the genre of each play in the right column of the figure, where 'h' denotes 'history,' 't' denotes 'tragedy,' 'c' denotes 'comedy,' and 'r' denotes 'romance.' The categorization result is very consistent with the genre [25]. In this example, we set $D_i = 1, \ldots, 36$, $Q_i = 1, \ldots, 36$, $N = 10,000$ (words with high frequencies of occurrence), $L_h = L_c = 200$, and $L_o = L_i = R^S = R = 64$ (features). The numbers in the figure indicate the publication years of the plays.

We provide a semantic search tool using the corpus of Shakespeare's comedies and tragedies at http://red.csie.ntu.edu.tw/literature/SAS.htm. Two search results are listed in Table 1. In this search, we set $D_i = 1, \ldots, 7777$ (the 7777 longest conversations in the 23 tragedies and comedies), $N = 10,000$, $L_o = L_i = R = 100$, $L_h = L_c = 200$, and $R^S = 64$. Each query indexed one conversation.

## 3. Coding for polysemous words

Following the studies ascribed in the previous section, we further developed a code structure for polysemous words that

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030

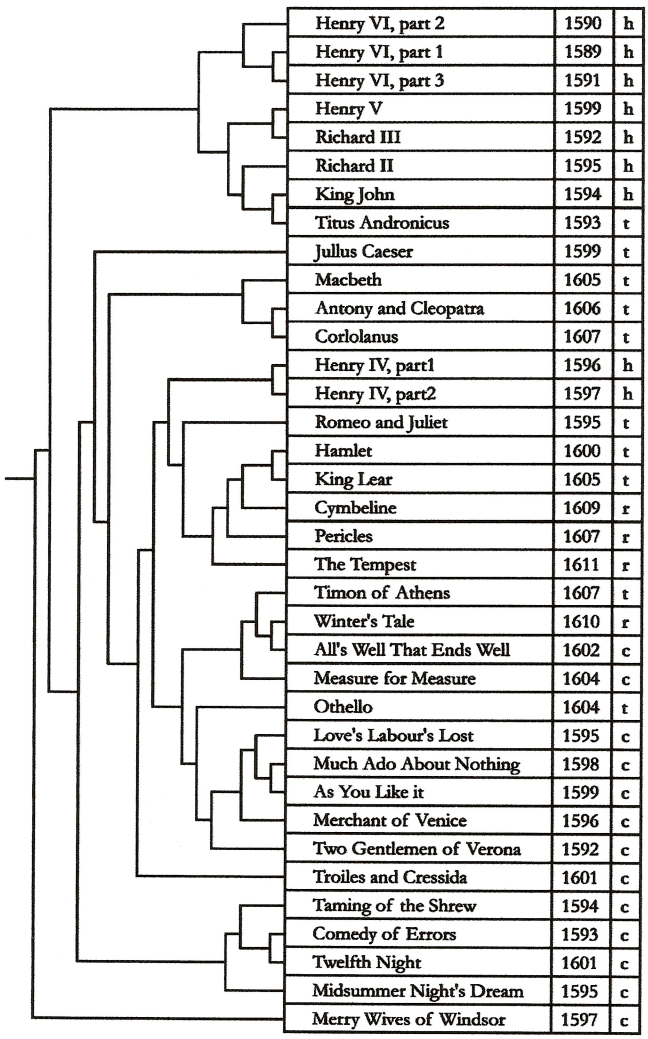


**Fig. 2.** Categorization of Shakespeare's plays.

**Table 1**
Search results by semantic associative search

| Query | Search result |
|---|---|
| She loves kiss | **BENVOLIO:** Tut, you saw her fair, none else being by herself poised with herself in either eye; but in that crystal scales let there be weigh'd. Your lady's love against some other maid that I will show you shining at this feast, and she shall scant show well that now shows best. – Romeo and Juliet |
| Armies die in blood | **MARCUS AND RONICUS:** Which of your hands hath not defended Rome, and rear'd aloft the bloody battle-axe, writing destruction on the enemy's castle? O, none of both but are of high desert my hand hath been but idle; let it serve. To ransom my two nephews from their death; then have I kept it to a worthy end. – Titus Andronicus |

resolves various degrees of multiple meanings of a single word. Based on the structure, we define a new measure.

To illustrate, the meaning of 'bank' in the sentence 'I keep money in a bank' differs from that in 'I fish on a river bank.' Furthermore, a single code of the word 'the' will cause difficult in resolving its following word, for example the two sentences, 'the dog is happy' and 'the tears well up.' It is more complicated about the meanings of 'will' in 'I will myself to say no' and 'I will say no', it is both polysemous and functional.

In Section 2, we treat the code of a word as a vector. This design is reasonable and expressive if most of the meanings of a word are close to the vector's position. This means a single position represents an isolated meaning. However, if a position is a compromise between two remote distributions of two separated meanings, then this position will carry no meaning. There is no semantic meaning of this word around this position. For example, the word 'lead' has two main meanings. If we encode the word by averaging all its meanings, the new code will be meaningless.

It may be practicable to code the word 'lead' into two vectors, one for each distribution of its semantics. The problem is that how far of two distributions should we code a word as two vectors. The meanings of 'lead' in 'lead is a metal' and in 'he leads us' can be separated clearly. But, the meanings of 'respect' in 'I respect my teacher' and in 'they differ in some respects' are very difficult to separate. Moreover, the meanings of 'eye' in 'she winks her blue eyes' and in 'the typhoon eye is advancing' are also very difficult to separate. No matter where the threshold is set, there is always a loss of information, because the degrees of the multi-semantics of a word constitute a continuous spectrum. Hence, we must solve the problem in another way, by designing a code structure that incorporates more than one semantic meaning and has the sense as those manually designed features in [26].

### 3.1. The idea

In Section 2, we collected a word's information by averaging all the predictions with (16) from all its precedent words after each epoch. When we study the variance structure of these predictions, we find detailed multi-semantic information. Let the matrix $V$ contains all the prediction vectors of a word. Each row of $V$ records a prediction vector and $V$ is an $R$ by $|S_n|$ matrix. The eigenvalues of $V^{\mathrm{T}}V$ are the variances of the predictions projected on the eigenvectors. By displaying the eigenvalues in descendant values, see Fig. 3, we observe the semantic contents of a word. Fig. 3 shows the spectrum of the largest 15 eigenvalues for 5000 words. In this figure, all the eigenvalues of a word are divided by the maximal one. This figure was obtained by using the first $5000 = N$ high occurrence words in the corpus $D = 36$ plays by Shakespeare. We set $L_o = L_i = R = R_S = 64$ and $L_h = L_c = 200$. We used re-encoding (16)–(18) to obtain this figure. We collected the predictions for each word after the $j$th = 4th epoch. These collected predictions were then used in $V^{\mathrm{T}}V$ to obtain the eigenvalues. Words that have sharp shapes, such as 'miss,' 'rag,' 'lead,' 'bank,' and 'scream,' carry concentrated semantic meanings. A word with a concave or convex shape carries wide meanings. A word with a straight slope shape, such as 'we' in this figure, has an even semantic distribution. Functional words or meaningless verbs like 'take,' 'get,' 'go' etc., may have such even distributions. When there are two or more concentrated semantic distributions, a single eigenvector cannot represent the semantic meanings of the word. The summation of all the eigenvalues for a word carries the same meaning as that obtained with (4). Fig. 3 also shows that the meaning of each word can be approximately expanded in a 13-D (or less) dimensional manifold embedded in a 64-D space. This result is comparable to that of the Chinese character (word) where each Chinese character has roughly nine meanings on average. This result has also been confirmed by inspecting the number of hidden features by factorizing the object matrix, $V$, using the NMF method into two matrices [11]. The number of hidden features is consistent with the result in this figure. Note that we will skip the

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030

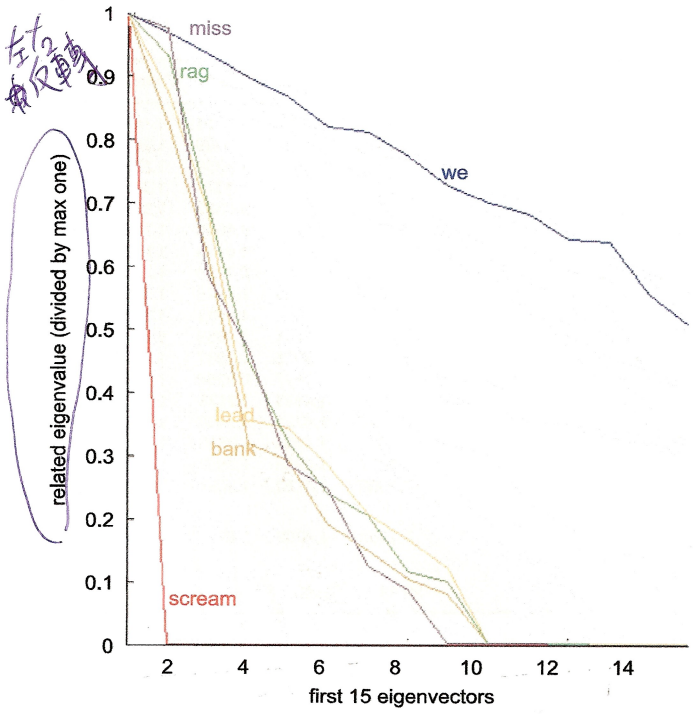


**Fig. 3.** Distributions of eigenvalues for each word. Six distributions are marked with their words. They are 'we,' 'miss,' 'rag,' 'lead,' 'bank,' and 'scream.'

discussion on designing multiple codes for the multiple meanings of a single word where each code corresponds to a single meaning.

### 3.2. Multi-meaning re-encoding

To save the rich semantic contents of a word that have a continuous spectrum, we have developed a code structure with the following constraints:

$$\prod_{r=1}^{R} w_{rn} = 1, \quad w_{rn} \geqslant 0,$$

$$n = 1\text{–}N, \quad r = 1\text{–}R. \tag{19}$$

These constraints are used in each training epoch in place of normalization (17), (18). Instead of a vector position, this new code has a form with a unit volume and all positive features. In order to satisfy this constraint, we remove those prediction vectors with negative features and save those predictions with nonnegative features to estimate the updated code in each epoch. Note that all positive features can also be obtained by using a sigmoid function with non-negative outputs in the output layer. We normalize every saved output, $\varphi(U_{oh}H(w(t-1)))$, to satisfy the constraints in (19). Normalization, $\prod_{r=1}^{R} w_{rn} = 1$, can be accomplished by applying a logarithm function to the vector elements.

Note that a new code obtained by combining two other codes will keep this constraint. The operation is

$$w_{rn}^{\text{new}} = w_{rn}^{\text{old1}} \times w_{rn}^{\text{old2}}, \quad r = 1\text{–}R. \tag{20}$$

This operation (20) strengthens the consistent features of two old codes and neutralizes the discordant features. It benefits computation. For functional words, because discordance neutralization happens in every feature, their codes will have a form similar to that of a unit hypercube with each feature equal to 1.

We normalize each saved vector, $\varphi(U_{oh}H(w(t-1)))$, after the training in each epoch in order to satisfy the constraint in (19). Let $\varphi_{rn}^{\text{nom}}(w(t-1))$ be the $r$th element of the normalized vector. Instead of using Eqs. (16)–(18), we calculate the updated code vector after each epoch as follows:

$$w_{rn}^{j} = \prod_{\substack{w(t)=w_n \\ w(t)\in D}} \varphi_{rn}^{\text{nom}}(w(t-1)),$$

$$r = 1\text{–}R, \quad n = 1\text{–}N. \tag{21}$$

The constraint for the updated code, $\prod_{r=1}^{R} w_{rn}^{j} = 1$, is naturally satisfied.

Different from the measure in (13), we write a new measure '*RSM*' (richness in semantic meaning) as the product of all features larger than 1:

$$RSM \equiv \prod_{r=1}^{R} \max(w_{nr}, 1). \tag{22}$$

Table 2 shows the adjectives which have the largest *RSM* values in the corpus of Shakespeare's 36 plays. We set $N = 10,000$, $L_o = L_i = R = R_S = 64$, and $L_h = L_c = 200$ to obtain the results shown in this table.

An *RSM* value may also reveal the richness of semantic meaning of a whole play. Table 3 shows the values for Shakespeare's plays [1,19,25]. We used the same trained codes as those shown in Table 2. A single vector was obtained by combing all of the word vectors (all $|D|$ words in a play $D$) using (20), that is, $w_r^D = \prod_{w_i \in D}^{|D|} w_{ri}$, $r = 1\text{–}R$. We divided the *RSM* value by the total number of words in each play and applied a natural logarithm function to it: $RSM^D \equiv \ln((1/|D|)\prod_{r=1}^{R} \max(w_r^D, 1))$. This value, $RSM^D$, shows the average meaning of a word in a whole play. From this table, the play 'Winter's Tale' has the largest $RSM^D$ value. The plays published after the year 1604 have relatively large $RSM^D$ values. Note that when we do not divide the $RSM^D$ value by the total number of words, $|D|$, this value will roughly proportion to $|D|$. So, Hamlet would have the highest value because it contains the largest number of words.

### 3.3. Example of literature similarity

Since trained codes can be used to reveal the writing style of an author, they may serve as personalized codes. We present an example based on a comparison of the writing styles of different authors. Four literary corpuses of works by four writers were prepared individually. These four writers are WS, Mark Twain (MT), Stephen Leacock (SL), and John Fletcher (JF). There are 36

**Table 2**
The largest '*RSM*' adjectives

| Adjective | RSM |
|---|---|
| Matchless | 1.00441 |
| Tyrannous | 1.004039 |
| Ungentle | 1.003496 |
| Pent | 1.003416 |
| Arrant | 1.003406 |
| Regent | 1.002543 |
| Specular | 1.002293 |
| Breezy | 1.002263 |
| Insolent | 1.002082 |
| Thersitical | 1.001912 |
| Perplexed | 1.001762 |
| Piteous | 1.001732 |
| Queer | 1.001712 |
| Tough | 1.001671 |
| Traversable | 1.001621 |
| Moderate | 1.001321 |

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030

**Table 3**
'$RSM^D$' values of the dramas by Shakespeare

| Title | Genre | $RSM^D$ |
|---|---|---|
| Winter's Tale | r | 2.2987 |
| Cymbeline | r | 2.2986 |
| The Tempest | r | 2.2986 |
| Pericles | r | 2.2971 |
| Timon of Athens | t | 2.2962 |
| Coriolanus | t | 2.2958 |
| Henry VI, part 1 | h | 2.295 |
| Antony and Cleopatra | t | 2.2944 |
| All's Well That Ends Well | c | 2.294 |
| King Lear | t | 2.2934 |
| Othello | t | 2.2932 |
| Measure for Measure | c | 2.292 |
| Macbeth | t | 2.2919 |
| Henry VI, part 3 | h | 2.2916 |
| Henry VI, part 2 | h | 2.2913 |
| Twelfth Night | c | 2.2901 |
| As You Like It | c | 2.2893 |
| Merry Wives of Windsor | c | 2.2892 |
| Julius Caeser | t | 2.2892 |
| Henry IV, part 1 | h | 2.2891 |
| Much Ado About Nothing | c | 2.2888 |
| Henry V | h | 2.2883 |
| Midsummer Night's Dream | c | 2.288 |
| Hamlet | t | 2.2872 |
| Troiles and Cressida | c | 2.2869 |
| Henry IV, part 2 | h | 2.2857 |
| King John | h | 2.2839 |
| Richard II | h | 2.2837 |
| Taming of the Shrew | c | 2.2836 |
| Merchant of Venice | c | 2.2835 |
| Richard III | h | 2.2832 |
| Romeo and Juliet | t | 2.2825 |
| Love's Labour's Lost | c | 2.2825 |
| Titus Andronicus | t | 2.2818 |
| Comedy of Errors | c | 2.2806 |
| Two Gentlemen of Verona | c | 2.2805 |

plays in the WS corpus, 16 works in the MT corpus, four works in the JF corpus, and four works in the SL corpus. All of the works are listed along with their years of publication under the 'Year' in a separate row in Tables 4–8. We obtained a set of codes for each writer by training an individual Elman network using his corpus. This set of codes was then used in a specific work to test its stylistic similarity. The $RSM^D$ value was calculated for that specific work. The results are listed in Tables 4–8.

The four sets of trained codes are listed in columns 3–6 under 'Code WS,' 'Code JF,' 'Code MT,' and 'Code SL.' These trained codes were then used in a specific work to compute its $RSM^D$ value. In the five tables, the first column lists the titles of the specific works, and the second column lists their publication years. The third to sixth columns list the calculated $RSM^D$ values.

The high and increasing values after 1604 in the WS column of Tables 4 and 5 show why people name the years around 1604 'the peak' of Shakespeare. To our knowledge, this value is the only one that can reveal the peak. The $RSM^D$ value of Mark Twain's fiction is small when we use the codes trained by Shakespeare's corpus, which implies that the connections of words and senses of sentences between the two masters are different. The largest $RSM^D$ of a writer can usually be reached by using his or her codes.

The last column lists the average percentages of the four $RSM^D$ values in each row. For example, the 'Grade' of the play 'Othello' in Table 5 is 0.7892, which is equal to $(2.2932/2.2987)+(1.4826/1.8504)+(1.1933/1.7878)+(1.2732/1.8436)=0.7892=78.92(100\%)$. The denominator 2.2987 (1.8504, 1.7878, 1.8436) is the largest $RSM^D$ value among all the $RSM^D$ values obtained using the WS code (JF code, MT code, SL code). This means that 'Winter's Tale' ('Philaster,' 'A Horse's Tale,' 'Arcadian

**Table 4**
'$RSM^D$' values of William Shakespeare's plays

| Title | Year | Code WS | Code JF | Code MT | Code SL | Grade |
|---|---|---|---|---|---|---|
| Henry VI, part 1 | 1589 | 2.2950 | 1.3379 | 1.1235 | 1.1424 | 74.24 |
| Henry VI, part 2 | 1590 | 2.2913 | 1.4015 | 1.2489 | 1.2625 | 78.44 |
| Henry VI, part 3 | 1591 | 2.2916 | 1.4393 | 1.3238 | 1.2750 | 80.17 |
| Richard III | 1592 | 2.2832 | 1.4639 | 1.3378 | 1.3084 | 81.06 |
| Two Gentlemen of Verona | 1592 | 2.2805 | 1.5195 | 1.0887 | 1.3018 | 78.21 |
| Comedy of Errors | 1593 | 2.2806 | 1.6138 | 0.8666 | 1.2728 | 75.98 |
| Titus Andronicus | 1593 | 2.2818 | 1.4603 | 1.0045 | 1.1484 | 74.16 |
| Taming of the Shrew | 1594 | 2.2836 | 1.4448 | 0.7780 | 1.2535 | 72.23 |
| King John | 1594 | 2.2839 | 1.4601 | 1.3679 | 1.3551 | 82.07 |
| Love's Labour's Lost | 1595 | 2.2825 | 1.4962 | 1.1593 | 1.2910 | 78.76 |
| Richard II | 1595 | 2.2837 | 1.4791 | 1.3059 | 1.3137 | 80.90 |
| Romeo and Juliet | 1595 | 2.2825 | 1.5085 | 1.3037 | 1.2458 | 80.33 |
| Midsummer Night's Dream | 1595 | 2.2880 | 1.3977 | 0.9564 | 1.1792 | 73.13 |
| Merchant of Venice | 1596 | 2.2835 | 1.4813 | 1.2138 | 1.3739 | 80.45 |
| Henry IV, part 1 | 1596 | 2.2891 | 1.4723 | 1.2357 | 1.2910 | 79.57 |
| Merry Wives of Windsor | 1597 | 2.2892 | 1.4904 | 1.2515 | 1.2705 | 79.76 |
| Henry IV, part 2 | 1597 | 2.2857 | 1.4954 | 1.2951 | 1.3690 | 81.74 |
| Much Ado About Nothing | 1598 | 2.2888 | 1.5120 | 1.2034 | 1.3851 | 80.93 |
| Average $RSM^D$ | | 2.2858 | 1.4708 | 1.1702 | 1.2799 | |
| Percentage | | 99.44 | 79.48 | 65.46 | 69.43 | 78.5 |

**Table 5**
'$RSM^D$' values of William Shakespeare's works

| Title | Year | Code WS | Code JF | Code MT | Code SL | Grade |
|---|---|---|---|---|---|---|
| Henry V | 1599 | 2.2883 | 1.4409 | 1.1987 | 1.3040 | 78.80 |
| Julius Caeser | 1599 | 2.2892 | 1.4500 | 0.9923 | 1.2715 | 75.61 |
| As You Like It | 1599 | 2.2893 | 1.5115 | 1.2497 | 1.4021 | 81.81 |
| Hamlet | 1600 | 2.2872 | 1.4710 | 1.3317 | 1.3843 | 82.14 |
| Twelfth Night | 1601 | 2.2901 | 1.4949 | 1.2007 | 1.3469 | 80.16 |
| Troiles and Cressida | 1601 | 2.2869 | 1.4068 | 0.7687 | 1.1462 | 70.17 |
| All's Well That Ends Well | 1602 | 2.2940 | 1.5647 | 1.4159 | 1.4103 | 85.01 |
| Measure for Measure | 1604 | 2.2920 | 1.5114 | 1.2466 | 1.3190 | 80.66 |
| Othello | 1604 | 2.2932 | 1.4826 | 1.1933 | 1.2732 | 78.92 |
| King Lear | 1605 | 2.2934 | 1.5260 | 1.3456 | 1.3121 | 82.17 |
| Macbeth | 1605 | 2.2919 | 1.4771 | 1.1931 | 1.2977 | 79.16 |
| Antony and Cleopatra | 1606 | 2.2944 | 1.4473 | 1.1919 | 1.3102 | 78.94 |
| Coriolanus | 1607 | 2.2958 | 1.5119 | 1.2147 | 1.3306 | 80.43 |
| Timon of Athens | 1607 | 2.2962 | 1.5284 | 1.0982 | 1.3295 | 79.01 |
| Pericles | 1607 | 2.2971 | 1.5141 | 1.3184 | 1.4117 | 83.02 |
| Cymbeline | 1609 | 2.2986 | 1.5178 | 1.2779 | 1.3663 | 81.90 |
| Winter's Tale | 1610 | 2.2987 | 1.5310 | 1.2886 | 1.3973 | 82.65 |
| The Tempest | 1611 | 2.2981 | 1.4847 | 1.2412 | 1.2822 | 79.80 |
| Average $RSM^D$ | | 2.2930 | 1.4929 | 1.2093 | 1.3275 | |
| Percentage | | 99.75 | 80.68 | 67.64 | 72.00 | 80.0 |

**Table 6**
'$RSM^D$' values of John Fletcher's works

| Title | Year | Code WS | Code JF | Code MT | Code SL | Grade |
|---|---|---|---|---|---|---|
| Philaster | 1609 | 2.1800 | 1.8504 | 1.4028 | 1.4884 | 88.51 |
| The Maid's Tragedy | 1610 | 2.2068 | 1.8450 | 1.4337 | 1.4465 | 88.59 |
| A King and No King | 1611 | 2.0340 | 1.8409 | 1.0689 | 1.3870 | 80.75 |
| The Scornful Lady | 1615 | 2.0709 | 1.8288 | 1.1829 | 1.4653 | 83.64 |
| Average $RSM^D$ | | 2.1229 | 1.8413 | 1.2721 | 1.4468 | |
| Percentage | | 92.35 | 99.51 | 71.15 | 78.48 | 85.4 |

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030

**Table 7**
'$RSM^D$' values of Mark Twain's works

| Title | Year | Code WS | Code JF | Code MT | Code SL | Grade |
|---|---|---|---|---|---|---|
| Innocents Abroad | 1869 | 1.6052 | 1.1636 | 1.6691 | 1.3857 | 75.31 |
| Roughing It | 1872 | 1.5382 | 1.1897 | 1.6976 | 1.4357 | 76.01 |
| The Gilded Age | 1873 | 1.4102 | 1.1639 | 1.7026 | 1.4506 | 74.54 |
| The Adventures of Tom Sawyer | 1876 | 1.5299 | 1.3450 | 1.7301 | 1.4472 | 78.63 |
| A Tramp Abroad | 1880 | 1.5355 | 1.1839 | 1.7162 | 1.4467 | 76.31 |
| The Prince and the Pauper | 1881 | 1.8501 | 1.3453 | 1.7155 | 1.4027 | 81.31 |
| Life on the Mississippi | 1883 | 1.5555 | 1.1945 | 1.7319 | 1.4949 | 77.55 |
| The Adventures of Huckleberry Finn | 1885 | 1.4423 | 1.2508 | 1.7573 | 1.4659 | 77.04 |
| A Connecticut Yankee in King Arthur's Court | 1889 | 1.7397 | 1.3036 | 1.7504 | 1.4848 | 81.14 |
| The American Claimant | 1892 | 1.4552 | 1.2501 | 1.7620 | 1.4867 | 77.52 |
| The Tragedy of Pudd'n'head Wilson | 1894 | 1.3068 | 1.1822 | 1.7635 | 1.3002 | 72.48 |
| The Mysterious Stranger | 1900 | 1.7558 | 1.3743 | 1.7748 | 1.5067 | 82.91 |
| A Double Barrelled Detective Story | 1902 | 1.3951 | 1.2433 | 1.7714 | 1.4561 | 76.49 |
| A Dog's Tale | 1904 | 1.7127 | 1.3912 | 1.7783 | 1.6225 | 84.29 |
| What Is Man? | 1906 | 1.5695 | 1.2644 | 1.7763 | 1.5450 | 79.94 |
| A Horse's Tale | 1907 | 1.6603 | 1.3180 | 1.7878 | 1.4704 | 80.80 |
| Average $RSM^D$ | | 1.5664 | 1.2602 | 1.7428 | 1.4626 | |
| Percentage | | 68.14 | 68.11 | 97.48 | 79.33 | 78.3 |

**Table 8**
'$RSM^D$' values of Stephen Leacock's works

| Title | Year | Code WS | Code JF | Code MT | Code SL | Grade |
|---|---|---|---|---|---|---|
| Sunshine Sketches of a Little Town | 1912 | 1.1234 | 1.1873 | 0.9568 | 1.8309 | 66.47 |
| Arcadian Adventures With the Idle Rich | 1914 | 1.0495 | 1.0316 | 0.7132 | 1.8436 | 60.32 |
| Frenzied Fiction | 1918 | 1.4986 | 1.2661 | 1.5312 | 1.8429 | 79.81 |
| Nonsense Novels | 1920 | 1.4270 | 1.2003 | 1.2693 | 1.8295 | 74.30 |
| Average $RSM^D$ | | 1.2746 | 1.1713 | 1.1176 | 1.8367 | |
| Percentage | | 55.45 | 63.3 | 62.51 | 99.63 | 70.2 |

Adventures with the Idle Rich') has the largest $RSM^D$ among all of the WS works (JF works, MT works, SL works). A work that possesses shared semantics can be revealed by a large 'Grade' value, such as 'The Maid's Tragedy' in Table 6.

The second row above the bottom in each of the five tables lists the average of all of the $RSM^D$ values above it in the same table. The bottom row lists the percentage of the averaged $RSM^D$ value. For example, the bottom value in the third column in Table 6 is $0.9235 = \frac{2.1229}{2.2987}$. This value was obtained for the John Fletcher's four works by using the Shakespeare code set. This value is the highest one across any two authors. The corresponding value, obtained by switching the roles of these two authors, is 0.8068, as shown in Table 5. The value 0.8068 is the second highest one. This evidence of shared styles between the two masters agrees with the view expressed in [1].

## 4. Summary

In summary, we modified Elman's encoding method to construct compositional representations, $w_n^J$, for words. This construction is both partial and tempered by qualifications of the whole corpus. It extracts temporal contents from the whole corpus epoch after epoch and may support for productive behavior or support inferencing. The representations can serve as a major role in 'quasi-regular' domains and serve to the abstract structure which underlies the surface strings of sentences or sequences of words. They have, roughly, a functional compositionality without being syntactically compositional (see 'Sentence Generating Function using Neural Network' in Website http://red.csie.ntu.edu.tw/english/Usage/index.php). The experimental results on categorizing literary works are consistent with the genre [25].

We further developed a richness measure, *RSM*, for the multi-semantic contents of a word that have a continuous spectrum and applied it to resolve 'the peak' of Shakespeare 1604. The results, largest value 0.9235 in Table 6 and largest value 0.8068 in Table 5, show the shared styles between John Fletcher and Shakespeare. These results agree with the view expressed in [1]. The *RSM* can be used to reveal and discriminate the writing styles of authors and works. Both the representations, $w_n^J$, and measure, *RSM*, can facilitate other research, such as studies on personalized codes, linguistic analysis, authorship identity, categorization, etc.

## Acknowledgment

This work was supported by National Science Council.

## References

[1] H. Bloom, Shakespeare: The Invention of Human, Riverhead Books, New York, 1998.
[2] J. Burrows, Questions of authorship: attribution and beyond, A lecture delivered on the occasion of the Roberto Busa Award, ACH-ALLC 2001, New York, Comput. Humanit. 37 (2003) 5–32.

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030

[3] J.L. Elman, Generalization, simple recurrent networks, and the emergence of structure, in: Proceedings of the 20th Annual Conference of the Cognitive Science Society, Mahway, NJ, 1998.
[4] J.L. Elman, E.A. Bates, M.H. Johnson, A. Karmiloff-Smith, D. Parisi, K. Plunkett, Rethink Innateness, MIT Press, Cambridge, MA, 1996.
[5] J. Felsenstein, PHYLIP (Phylogeny Inference Package) version 3.5c [Program], Department of Genetics, University of Washington, Seattle, 1993.
[6] W.B. Frakes, Stemming algorithms, in: W.B. Frakes, B.-Y. Ricardo (Eds.), Information Retrieval: Data Structures and Algorithms, Prentice-Hall, Englewood Cliffs, NJ, 1992, pp. 131–160.
[7] D.I. Holmes, The evolution of stylometry in humanities scholarship, Lit. Linguist. Comput. 13 (1998) 111–117.
[8] D.A. Huffman, A method for the construction of minimum-redundancy codes, Proc. Inst. Radio Eng. 40 (1952) 1098–1102.
[9] M.I. Jordan, Serial order: a parallel distributed processing approach, Cognitive Science Institute Technical Report, 1986, file#:8604.
[10] T. Kohonen, Clustering, taxonomy, and topological maps of patterns, in: Proceedings of the Sixth International Conference on Pattern Recognition (ICPR), 1982, pp. 114–128.
[11] D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (1999) 788–791.
[12] C.-Y. Liou, Backbone structure of hairy memory, in: The 16th International Conference on Artificial Neural Networks, ICANN, Lecture Notes in Computer Science, Springer, Berlin, vol. 4131, Part I, 2006, pp. 688–697.
[13] C.-Y. Liou, W.-C. Cheng, Resolving hidden representations, in: International Conference on Neural Information Processing (ICONIP), Lecture Notes in Computer Science, Kitakyushu, Japan, Springer, Berlin, 2007.
[14] C.-Y. Liou, J.-C. Huang, Y.-T. Kuo, Geometrical perspective on learning behavior, J. Inf. Sci. Eng. 21 (2005) 721–732.
[15] C.-Y. Liou, S.-L. Lin, The other variant Boltzmann machine, in: International Joint Conference on Neural Networks, IJCNN, Washington, DC, USA, 1989, pp. 449–454.
[16] C.-Y. Liou, S.-L. Lin, Finite memory loading in hairy neurons, Natural Comput. 5 (2006) 15–42.
[17] C.-Y. Liou, W.-J. Yu, Ambiguous binary representation in multilayer neural network, in: Proceedings of the International Conference on Neural Networks, vol. 1, ICNN, Perth, Australia, 1995, pp. 379–384.
[18] C.-Y. Liou, S.-K. Yuan, Error tolerant associative memory, Biol. Cybern. 81 (1999) 331–342.
[19] T. McEnery, M. Oakes, Authorship identification and computational stylometry, in: Handbook of Natural Language Processing, Marcel Dekker, New York, 2000, pp. 545–562.
[20] M.F. Porter, An algorithm for suffix stripping, Program 14 (1980) 130–137.
[21] D.E. Rumelhart, J.L. McClelland (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations, MIT Press, Cambridge, MA, 1986.
[22] N. Saitou, M. Nei, The neighbor-joining method: a new method for reconstructing phylogenetic trees, Mol. Biol. Evol. 4 (1987) 406–425.
[23] F.J. Tweedie, R.H. Baayen, How variable may a constant be measures of lexical richness in perspective?, Comput. Humanit. 32 (1998) 323–352.
[24] C.B. William, Mendenhall's studies of word-length distribution in the works of Shakespeare and Bacon, Biometrika 62 (1975) 207–212.
[25] A.C.-C. Yang, C.-K. Peng, H.-W. Yien, A.L. Goldberger, Information categorization approach to literary authorship disputes, Physica-A Stat. Mech. Appl. 329 (2003) 473–483.
[26] N. Yoshida, Y. Kiyoki, T. Kitagawa, An associative search method based on symbolic filtering and semantic ordering for database systems, in: Proceedings of 7th IFIP 2.6 Working Conference on Database Semantics (DS-7) in Leysin, Switzerland, 1997, pp. 215–237.

**Cheng-Yuan Liou** was born in Taiwan in 1951. He received the Ph.D. from Massachusetts Institute of Technology in 1985. He is a Professor in the Department of Computer Science and Information Engineering, National Taiwan University. His research interests include brain theory and neural networks.

**Jau-Chi Huang** received the B.S. and M.S. degrees from Department of Computer Science and Information Engineering, National Taiwan University in 2000 and 2002. She is a Ph.D. candidate in the same department. Her research interests are neural networks, artificial intelligence, and semantic search engines. Q1

**Wen-Chi Yang** received the B.S. and M.S. degrees from National Taiwan University in Taipei, Taiwan. His research interests are semantic search engines, neural networks, and artificial intelligence.

Please cite this article as: C.-Y. Liou, et al., Modeling word perception using the Elman..., Neurocomputing (2008), doi:10.1016/j.neucom.2008.04.030