Neurocomputing 139 (2014) 84–96

Contents lists available at ScienceDirect

Neurocomputing

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

Autoencoder for words

Cheng-Yuan Liou [a,*], Wei-Chen Cheng [b], Jiun-Wei Liou [a], Daw-Ran Liou [c]

[a] Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, ROC
[b] Institute of Statistical Science, Academia Sinica, Taiwan, ROC
[c] Sibley School of Mechanical and Aerospace Engineering, Cornell University, United States

ARTICLE INFO

Article history:
Received 13 March 2013
Received in revised form
31 August 2013
Accepted 16 September 2013
Available online 8 April 2014

Keywords:
Minimum entropy coding
Writing style similarity
Elman network
Polysemous word
Semantic indexing

ABSTRACT

This paper presents a training method that encodes each word into a different vector in semantic space and its relation to low entropy coding. Elman network is employed in the method to process word sequences from literary works. The trained codes possess reduced entropy and are used in ranking, indexing, and categorizing literary works. A modification of the method to train the multi-vector for each polysemous word is also presented where each vector represents a different meaning of its word. These multiple vectors can accommodate several different meanings of their word. This method is applied to the stylish analyses of two Chinese novels, Dream of the Red Chamber and Romance of the Three Kingdoms.

1. Introduction

In this work, we study an auto-encoder for the words in literary works. This encoder is accomplished by a devised training method that operates on Elman network [1,2]. Elman network is a super-Turing machine with powerful computation capability [3]. It can be trained to accomplish difficult tasks. We expect that it is powerful enough to capture the diversity of the language and accomplish the encoding task.

The output of the network is a renewed code vector after each training pass. Each dimension of the code may be looked at as a labeled attribute for the word that provides a partial meaning to its word. Manually constructed attributes for a large number of words have been developed for the purpose of indexing of documents, such as the associative method in [4]. It is costly and often brings contradictions to label words with multiple attributes manually by a team of linguists.

The stylish analyses based on statistics on the frequency of occurrence (or co-occurrence) of certain word (words) or phrase (phrases) have been developed with varying degrees of success [5–7]. These analyses must be manually operated repeatedly with consistent and confirmed results. Such statistical methods cannot be applied to document retrieval and semantic indexing. The trained codes obtained in this paper are useful in semantic indexing, see [8]. The non-negative matrix factorization [9] accomplished the stylish task by iteratively training the neural weights with improved non-negative values. The factorization is also based on statistics of words in a whole document. It cannot be used to pick a portion of a document.

Since Elman network is designed for semantic categorization of words [1,2], it cannot support the encoding task. This paper redesigned Elman network to encode the words in semantic space. The achieved codes can be used in ranking, indexing, and categorizing literary works. In the next section, we introduce the detailed design of the single-code encoder and Elman network mentioned above. Reduction of entropy is studied and many experimental results for the novel “The Adventures of Peter Pan” by James Matthew Barrie are recorded and illustrated in the section.

In the third section, we perform the more general approach to deal with the multi-code for the polysemous word. To achieve this goal, we associate with each word with a set of distinct codes such that these codes can accommodate several different meanings of their word. The trained codes are applied to and offer a new approach to the stylish analysis.

2. Single-code encoder by Elman network

In this section, we illustrate the detailed structure and training method for the single-code encoder.

2.1. Encoder and redesigned network

Elman network is a simple recurrent network that has a context layer, see Fig. 1. It was designed to find the hidden structure in sequential patterns [1]. Let L_o, L_h, L_c, and L_i be the number of neurons in the output layer, the hidden layer, the

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

http://dx.doi.org/10.1016/j.neucom.2013.09.055
0925-2312/

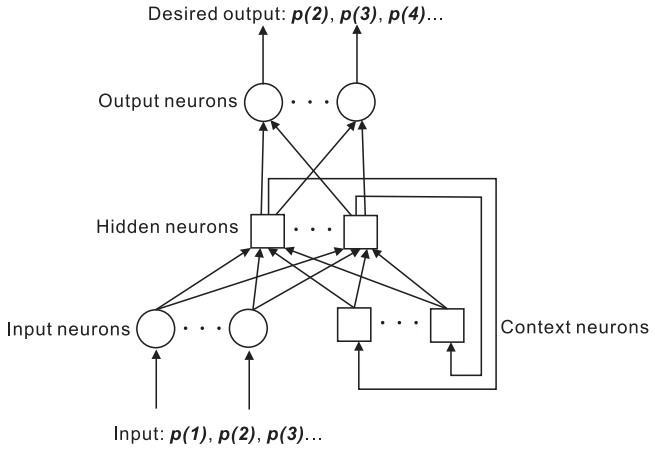

Fig. 1. Illustration of Elman network.

context layer, and the input layer, respectively. In the network, L_h is equal to L_c. During operation, at each training step t, the output of the hidden layer will be loaded to the context layer and together with the input layer to activate the hidden layer at time $t+1$.

Let the two weight matrices between layers be W_{oh} and W_{hic}, where W_{oh} is an L_h+1 by L_o matrix and W_{hic} is an L_i+L_c+1 by L_h matrix. Consider a sequence of input patterns, $\{p(t), t=1,2,3,\ldots\}$, the output vector of the hidden layer is denoted as $h(p(t))$ when $p(t)$ is fed to the input layer. $h(p(t))$ is an L_h by 1 column vector with L_h elements. Let $o(p(t))$ be the output vector of the output layer when $p(t)$ is fed to the input layer. $o(p(t))$ is an L_o by 1 column vector. The function of the hidden layer is $h(p(t))=\varphi(W_{hic}^T[in(p(t))])$, where $[in(p(t))]$ is an L_i+L_c+1 by 1 column vector and φ is a sigmoid activation function that operates on each element of a vector [10]. The input vector $[in(p(t))]$ has the form $[in(p(t))]=[p^T(t), h^T(p(t-1)), 1]^T$. The sigmoid function used in this paper is $\varphi(x)=[2/(1+\exp(-0.5x))]-1$. The function of the output layer is $o(p(t))=\varphi(W_{oh}^T[h^T(p(t)), 1]^T)$.

The back-propagation (BP) training algorithm [10] is commonly employed to train the weights, W_{oh} and W_{hic}, to reduce the difference between $o(p(t))$ and its desired output. The next input pattern $p(t+1)$ is served as the desired output [1]. For example, consider a sequence of word stream, $\{p(1), p(2), p(3), \ldots\}$, the input at time $t=1$ is $p(1)$, and its desired output at time $t=1$ is $p(2)$. The input at time $t=2$ is $p(2)$, and its desired output at time $t=2$ is $p(3)$. All the attempts are aimed at minimizing the error between the network's output and its desired output, $\|o(p(t))-p(t+1)\|^2$, to satisfy the prediction $o(p(t))\approx p(t+1)$.

Consider a corpus with N different words $\{c_n;\ n=1,2,\ldots,N\}$, each word c_n initially has a random lexical code vector with R dimensions, $c_n=[c_{n1}, c_{n2}, \ldots, c_{nR}]^T$. R is equal to L_i in this paper. The input word stream is encoded accordingly, $p(t)=c_i;\ c_i\in\{c_n; n=1,2,\ldots,N\}$. This stream is the same as that used in Elman's method. The redesigned network [8,11] minimizes the prediction error $\|o(p(t))-p(t+1)\|^2$ between the network's output $o(p(t))$ and its desired output $p(t+1)$ to satisfy the prediction $o(p(t))\approx p(t+1)$ by the BP algorithm. The weights W_{hic} and W_{oh} are updated after each word presented.

Set one epoch as when all T words in the corpus were presented, where T is the total number of words in the corpus, $T>N$. We renew the codes every k epochs. Let g be the number of epochs in the training process. The first renewal, $r=2$, is operated after the training of the first k epochs. We operate a "renewal" after each k epochs. We call each k epochs a "pass". After the training in each k epochs, a new raw code is calculated as follows:

$$c_n^{raw}=\frac{1}{freq_n}\sum_{\{t|p(t)=c_n^r\}} o(p(t-1)),\quad n=1\sim N, \tag{1}$$

where $freq_n$ is the number of times that the nth word c_n appears in the current training pass. Note that Elman averaged all the hidden output vectors for each word c_n, but we averaged all its prediction vectors instead.

All renewed codes are normalized before using them in the next pass by the equation

$$c_n^{r+1}=c_n^{nom}=\|c_n^{ave}\|^{-1}c_n^{ave}, \tag{2}$$

where the norm function is $\|c_n\|=(c_n^Tc_n)^{0.5}$. This equation sets the norm of each code vector as 1 and is able to prevent a diminished solution, $\{\|c_n\|\sim 0,\ n=1\sim N\}$, usually derived by the BP algorithm. c_n^{ave} is the nth column of the matrix $C_{R\times N}^{ave}$,

$$C_{R\times N}^{ave}=C_{R\times N}^{raw}-\frac{1}{N}C_{R\times N}^{raw}\begin{bmatrix}1 & \cdots & 1\\ \vdots & 1 & \vdots\\ 1 & \cdots & 1\end{bmatrix}_{N\times N}. \tag{3}$$

In (3), $C_{R\times N}$ is a code matrix and has the form $[c_1, c_2, \ldots, c_N]$. Eq. (3) makes each row of the code matrix $C_{R\times N}^{ave}$ become zero-mean.

The initial codes, $c_n^{r=1}$, before the first renewal is randomly assigned under the restriction that different words have different codes and they are normalized. Then use the BP algorithm to reduce the prediction error $\|o(p(t))-p(t+1)\|^2$ in each training step.

Note that the training step is operated after each word presented and the renewing step is operated after each k epochs. Iterations on these two steps constitute the training process. After the training process, we expect that each element of the code vector c_n contains a well learned attribute of the word c_n.

2.2. Properties of the encoder

Data description: We use the novel "The Adventures of Peter Pan by James Matthew Barrie" as the corpus to illustrate the encoder properties. We concatenated all sentences in it to form the training word stream. Note that the stream Elman used is formed with randomly sampled sentences from a manually constructed sentence pool. To reduce the amount of vocabularies, we apply several grammar rules to separate the suffix from the word such as "-s", "-ing", and "-ed". For example, "playing" becomes "play + ING"; "lights becomes "light + NVs"; "children's" becomes "children+ 's"; "turned" becomes "turn + Ved". After preprocessing, there are 3805 different root words including the mark that is added to represent the end of a sentence. The total number of sentences is 3101 and the total number of words is $T=54{,}999$.

Experiment setting: The network has $L_i=$ 15 input neurons, $L_o=15$ output neurons, $L_h=30$ hidden layer neurons, and $L_c=30$ context layer neurons. The numbers of neurons, L_i and $L_o=15$, are

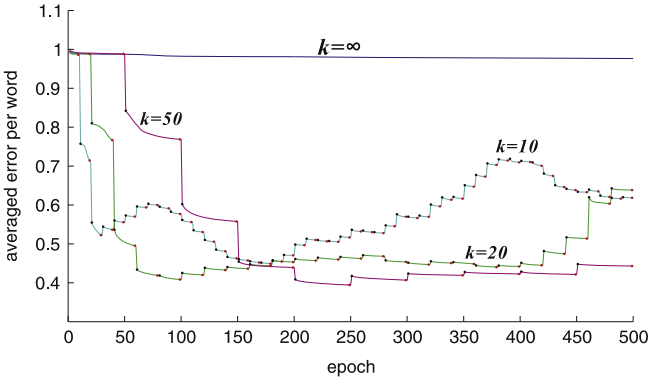

Fig. 2. Error curves under different re-encoding conditions, $k=\infty$, $k=10$, $k=20$, and $k=50$. The curve from red point to black point shows how error is reduced by re-encoding. The curve from black point to red point shows how error is reduced by BP weight adjustment. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

determined empirically. The number of hidden neurons $L_c = 30$ is close to the value in [10]. The initial values of the weights W_{hic} and W_{oh} are randomly assigned in the range $[-1, 1]$ and the initial values of the neurons in the context layer are set to zero. The initial code $c_n^{g=1}$ (or $c_n^{r=1}$) is randomly assigned under the restriction that different words have different codes and then is normalized through (2) and (3). Then use the BP algorithm to reduce the prediction error which is $\|o(p(t)) - p(t+1)\|^2$. The learning rate is fixed to 0.01. W_{hic} and W_{oh} are updated after each word presented. We set one epoch, $g=1$, as when all 54,999 words were presented, and we renew the codes every k epochs.

Experiment results: Fig. 2 recorded the error curves under different numbers of renewing epochs, $k = \infty$ (codes would not be renewed), 10, 20, and 50. This figure shows that the errors decrease drastically when we use the renewed codes. The performance of the BP algorithm on weight adjusting is far less than that of code renewing. To our knowledge, it is not possible to reach such low errors without the renewed codes using any existing algorithms. This figure also shows that intensive re-encoding may cause severe fluctuations.

Fig. 3 displays the codes during training under $k=20$ renewing condition. We use the principal component analysis (PCA) to map

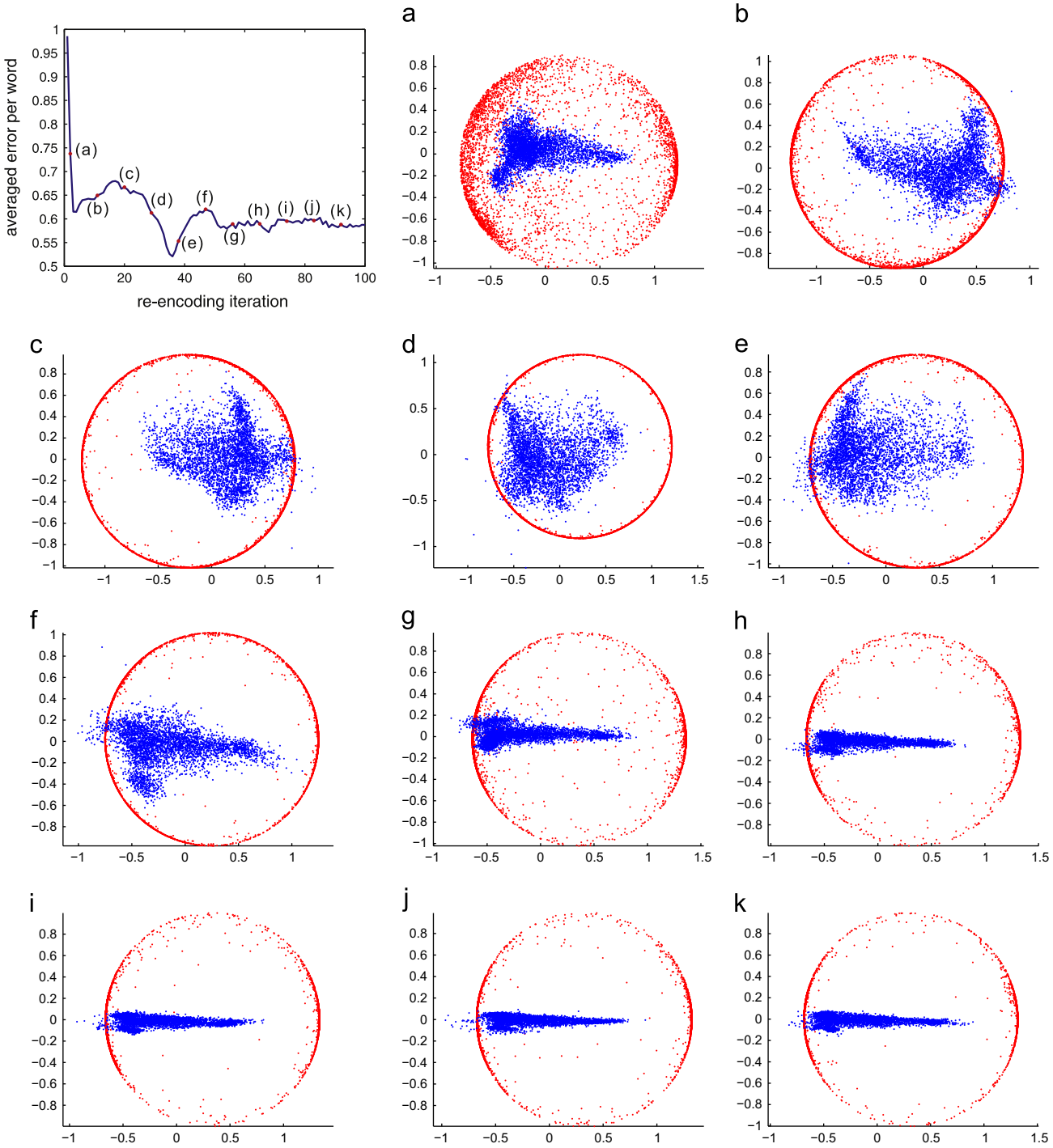

Fig. 3. Using PCA to display the code distributions on 2*D* space after different training epochs. We re-encode the words every 20 epochs. The blue points display the raw codes c_n^{raw}. The red points display the renewed codes c_n^{nom} after normalization. (a) $c_n^{g=20}$; (b) $c_n^{g=200}$; (c) $c_n^{g=380}$; (d) $c_n^{g=560}$; (e) $c_n^{g=740}$; (f) $c_n^{g=920}$; (g) $c_n^{g=1100}$; (h) $c_n^{g=1280}$; (i) $c_n^{g=1460}$; (j) $c_n^{g=1640}$; (k) $c_n^{g=1820}$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

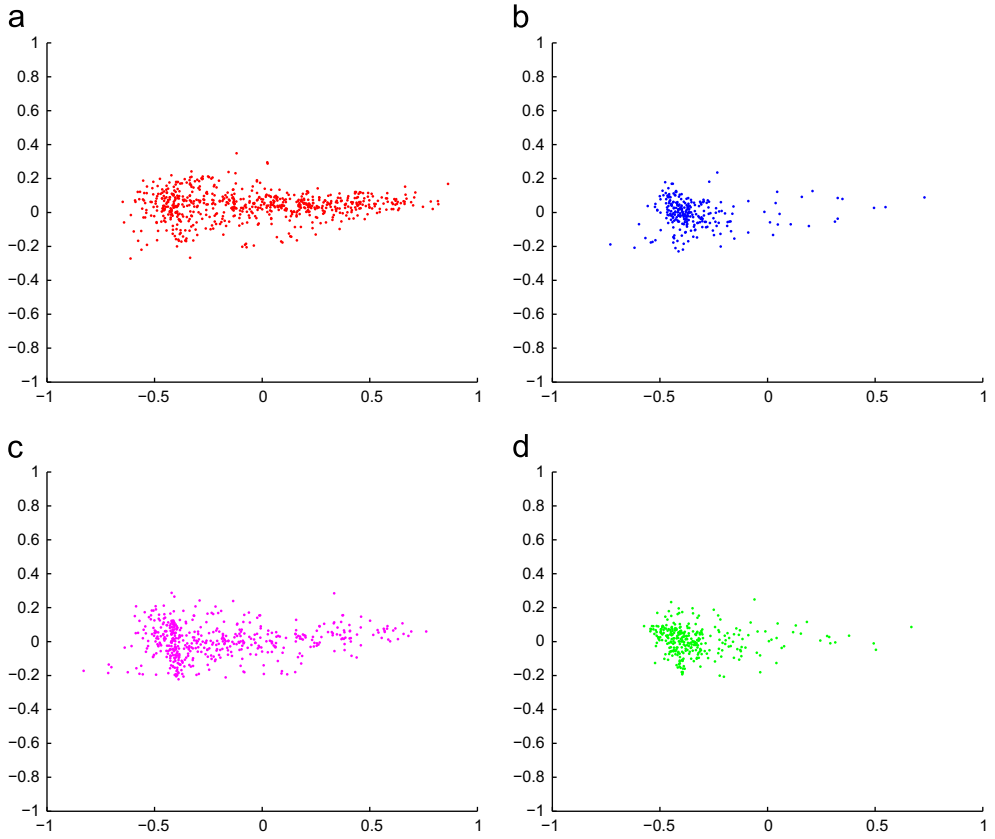

Fig. 4. We use PCA to analyze the code distribution on 2*D* space before normalization at the minimum epoch: (a) *noun*; (b) *verb*; (c) *adjective*; (d) *adverb*.

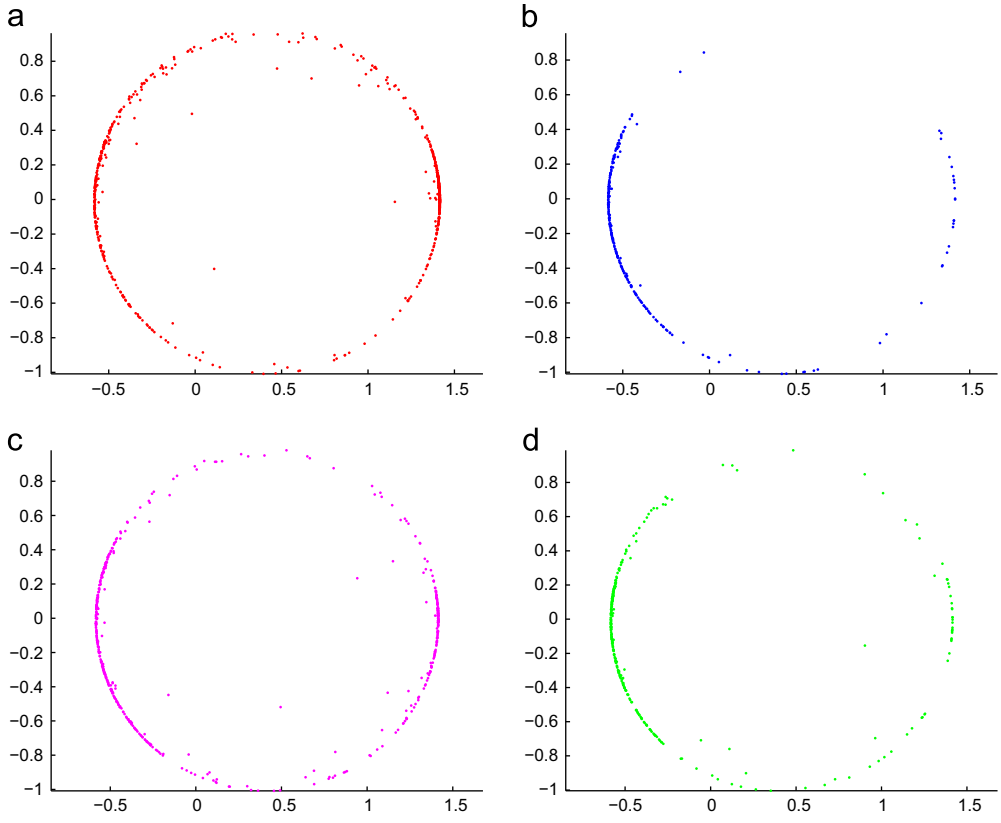

Fig. 5. We use PCA to analyze the code distribution on 2*D* space after normalization: (a) *noun*; (b) *verb*; (c) *adjective*; (d) *adverb*.

all output vectors, $o(p(t))$, from the code space ($R=15$ dimension) to a 2D space to show their distributions after different training epochs. This 2D space is constructed with the two eigenvectors of the largest two eigenvalues of $C_{R\times N}C_{R\times N}^{T}$. We see that normalization will force the codes distributed on a circle and they will not merge to a single point. However, this may change the interior structure of codes and cause that the error curve continually fluctuates.

2.3. Distributed representation of word

Elman averaged the hidden layer outputs of each word as its internal representation of that word and then fitted those representations of all words into a hierarchical clustering tree. The tree shows that there are several categories of words. These categories possess both syntactic relationships, such as noun vs. verb, and semantic relationships, such as human vs. animal.

In the redesigned method, we average the predictions of each word as the representation of that word instead and expect that the representation codes preserve syntactic and semantic relationships as well [11].

We analyze the renewed codes $c_n^{g=m+1}$, where the mth epoch achieves the minimum error, see the figure plotted on the top left corner of Fig. 3. The tree that Elman used to analyze the internal representations of words is restricted to display limited relationships of words among a small number of artificial vocabularies. We use PCA to map them from the code space ($R=15$) to a 2D space to show the distributions directly, see Figs. 4 and 5. We see verbs and adverbs are densely crowd to the left part, and nouns and adjectives have no clustering effect. The software that we use to judge the part of speech (nouns, verbs, adverbs and adjectives) is a lexical database, WordNet.

As for semantic relationships, we plot the pair of codes which have similar semantic meanings on the 2D space in Fig. 6. For comparison, we also plot the pair of codes which are randomly selected in Fig. 7. We see that the words with similar semantic meanings tend to cluster together. The distances in all 15 dimensions among the eight words are list in Tables 1 and 2. The distances in the lower triangle matrix are those recorded at the epoch 20. Those in the upper triangle are at epoch 1000. The initial codes are randomly assigned. The mean distance at epoch 1000 among the 8 words in Table 1 is 0.94 and the mean distance at epoch 1000 in Table 2 is 1.12. This shows that related words tend to have small

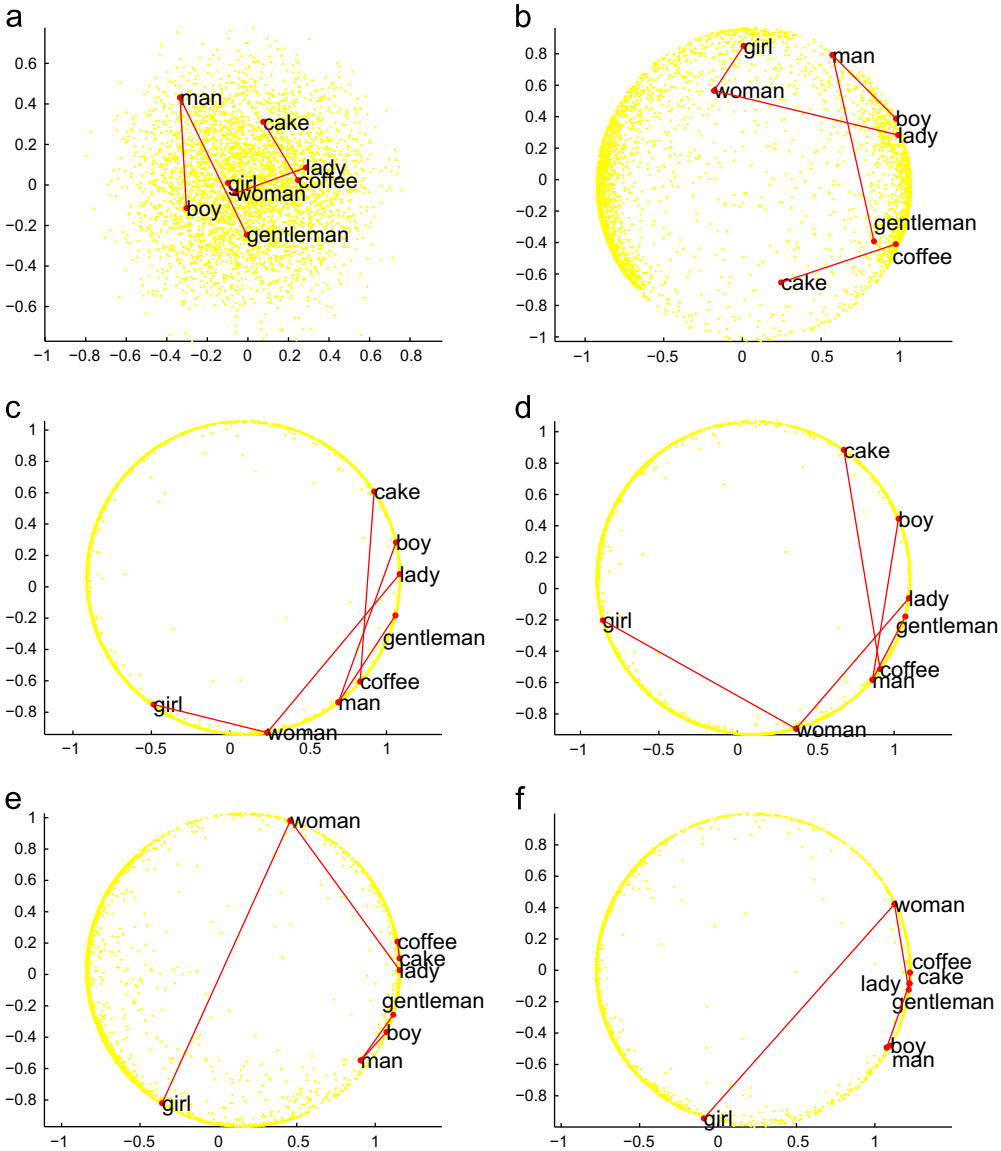

Fig. 6. The code relationship between semantic similar words. The pair of codes which have similar meanings are connected by a red line. The training error decreased from (a) to (f). (f) has the minimum training error. (a) $c_n^{g=0}$; (b) $c_n^{g=60}$; (c) $c_n^{g=600}$; (d) $c_n^{g=660}$; (e) $c_n^{g=6540}$; (f) $c_n^{g=6640}$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

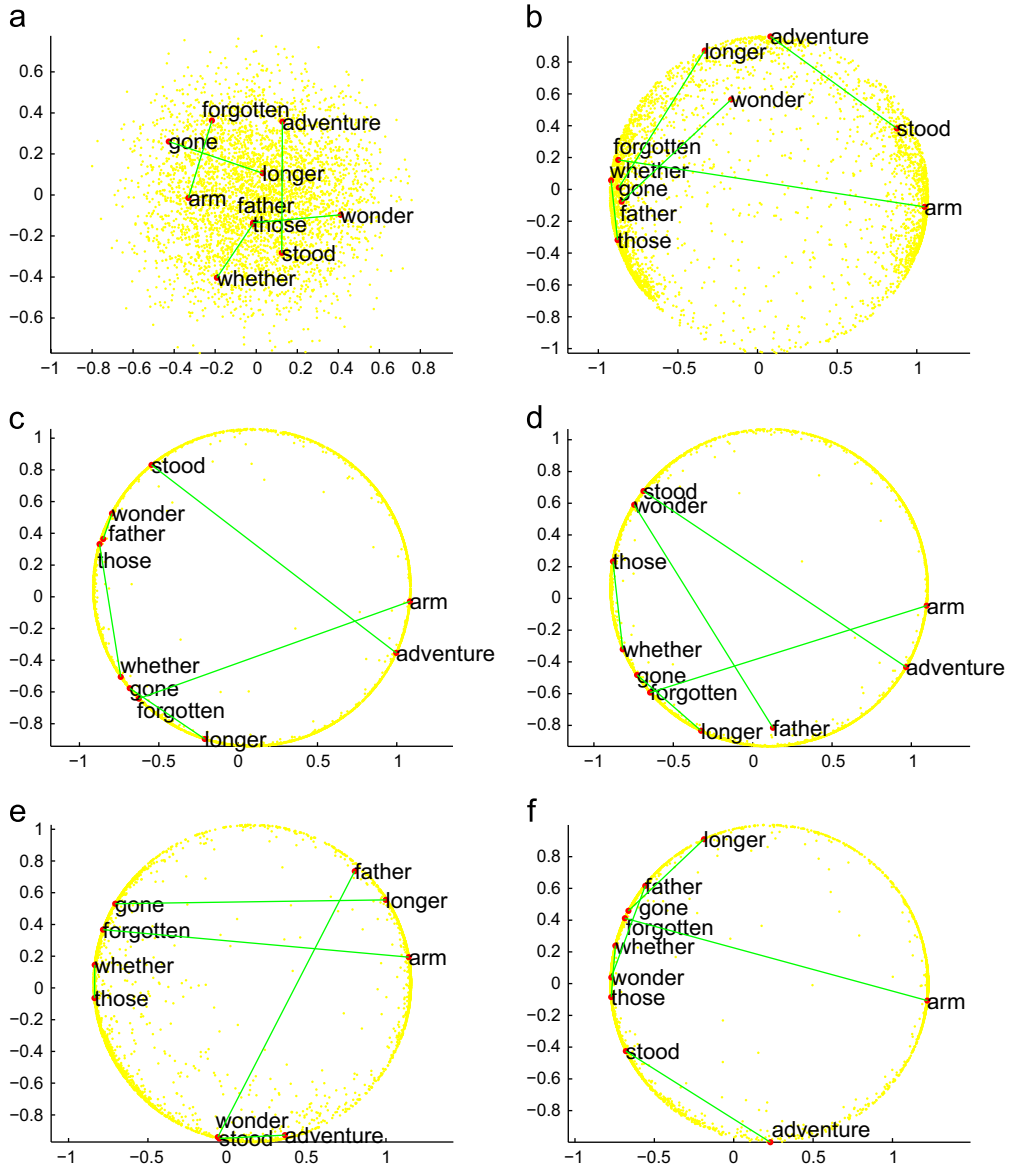

Fig. 7. Some randomly selected word pairs for comparison. The pair of codes are connected by a green line. (a) $c_n^{g=0}$; (b) $c_n^{g=60}$; (c) $c_n^{g=600}$; (d) $c_n^{g=660}$; (e) $c_n^{g=6540}$; (f) $c_n^{g=6640}$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

Table 1
The distances among related words. The distances in the lower triangle matrix are those recorded at the epoch 20. Those in the upper triangle are at epoch 1000.

	boy	man	gentleman	woman	girl	lady	cake	coffee
boy	0	0.127	0.50	1.95	1.99	0.91	0.44	0.20
man	1.43	0	0.59	1.96	1.98	0.99	0.54	0.31
gentleman	1.56	1.20	0	1.77	1.96	0.47	0.09	0.29
woman	1.61	1.41	1.43	0	0.56	1.52	1.80	1.89
girl	1.33	1.40	1.09	1.43	0	1.81	1.97	1.99
lady	1.62	1.51	1.56	1.05	1.51	0	0.55	0.74
cake	1.49	1.32	1.40	1.37	1.57	1.42	0	0.24
coffee	1.65	1.55	1.52	1.08	1.74	1.20	1.25	0

distances. There exists an exception, gentleman and cake in Table 1. This may be from insufficient training. Note that Fig. 6 shows that the code relationships between similar words, “girl”, “woman”, and “lady” at $g=60$ in Fig. 6(b), are much closer than those of $g=6640$ in Fig. 6(f). This is because there exist local distortions of similar words when the global error is reduced toward convergence.

In [2], Elman encoded each word as randomly assigned distributed representation to prove that the localist representations were not necessary to the connectionist models. However, the randomly assigned distributed representation of each word is arbitrary and is far from the real world composition of perceptual features (or attributes) that human infants use. Through the

Table 2
The distances among unrelated words. The distances in the lower triangle matrix are those recorded at the epoch 20. Those in the upper triangle are at epoch 1000.

	forgotten	arm	gone	longer	adventure	stood	whether	bottle
forgotten	0	2.00	0.16	1.83	0.18	1.96	0.42	2.00
arm	1.33	0	1.99	0.48	1.99	0.39	1.96	0.08
gone	1.25	1.37	0	1.90	0.33	1.92	0.56	1.99
longer	1.46	1.53	1.41	0	1.95	0.48	1.96	0.51
adventure	1.63	1.63	1.32	1.09	0	1.99	0.29	1.99
stood	1.55	1.43	1.60	1.33	1.35	0	1.98	0.33
whether	1.57	1.51	1.24	1.67	1.28	1.19	0	1.96
bottle	1.36	1.62	1.39	1.47	1.49	1.33	1.33	0

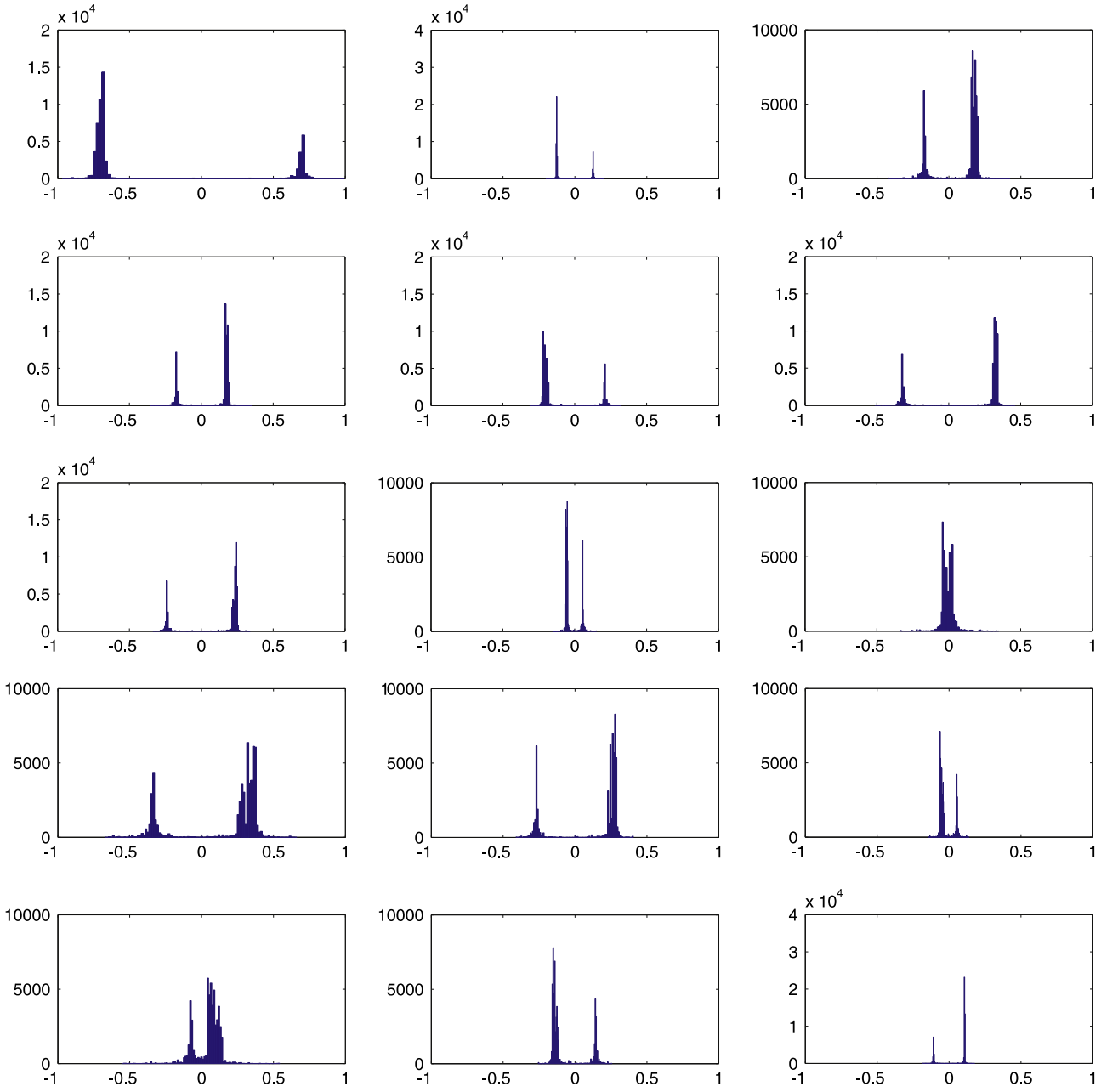

Fig. 8. Histograms of each dimension from $c_n^{g=1900}$. The number of bins is 100. The element values in each dimension of the codes are marked on the horizontal axis. The vertical axis marks the number of codes in a bin.

redesigned method, the representation can possess the form-based and function-based similarity in certain degree. In the next subsection, we use the minimum entropy coding [12] to further reveal the content of the representation codes.

2.4. Minimum entropy coding

To use the redundancy of sensory messages efficiently, Barlow developed the principle of minimum entropy coding [12,13].

The idea is to seek a set of symbols to represent sensory messages such that the occurrence of each symbol is as nearly as possible independent of each other. He used 7-bit ASCII code as an example to make the principle clear. The entropy calculated from A_l, which is the probability of the character l in an ordinary text, is $E(A) = -\sum_l A_l \log A_l$, and for a code a, the entropy expression for the bits takes the form:

$$e(A,a) = -\sum_i p_i \log p_i - \sum_i q_i \log q_i \tag{9}$$

where $p_i = 1 - q_i$ is the probability of the ith bit taking the value 1 in the encoded form of an ordinary text. $e(A,a)$ is always greater than $E(A)$ because the capacity of the bits of the code is greater than that of the states they represent, except in the case of a factorial code, $e(A,a) = E(A)$, where the states are represented by a product of independent bits. The goal is to find the code with minimum $e(A,a)$. Barlow suggested that the minimum entropy coding is the one which the representation of sensory information is based on. Moreover, a code selected according to this principle stores a wealth of knowledge about the statistical structure of the normal environment. However, he did not provide a constructive method other than searching through all codes and calculating the summed entropy for each to find the minimum entropy code.

In Fig. 8, we plot the histograms of each dimension for all words in the epoch, $c_n^{g=1900}$, to display the distribution of coding values in each dimension. The initial codes distribute $c_n^{r=1}$, evenly between -0.5 and 0.5. After re-encoding, the codes centralize to one or two values. Although the codes we obtained from our re-encoding method are not binary, we still can use quantized scales to roughly estimate the entropy. The entropy is calculated by $-\sum p_i \log p_i$, where $i = 1 \backsim$ *total number of bins*, p_i is the number of elements in each bin divided by the total number of elements, and $p_i = 0$ is ignored. Note that after quantization, any two words cannot have the same code. We increase the number of bins to 100,000 to meet this restriction. The entropies of the codes for the first 100 re-encoding renewals are displayed in Fig. 9. We see that the entropy decreases during the re-encoding process, and it implies that the network extracts the knowledge (redundancy) from sequential words and stores it in the codes during learning.

In [12] Barlow wrote, "a representation whose elements are independent makes it possible to form associations with logical functions of the elements, not just with the elements themselves." These logical functions play the key role in composition semantics. Through the re-encoding method, we obtain decreased entropy coding over re-encoding renewals. Both form-based similarity and function-based similarity can be loaded to the code. This decreasing is consistent with Barlow's idea [13] and provides independent representations of words. To our knowledge, this is the only constructive method to reach such coding.

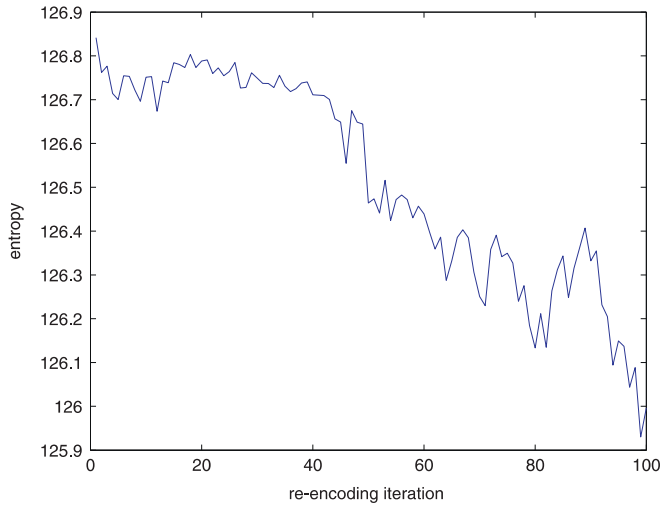

Fig. 9. The entropy decreases during the re-encoding process. Each number along the horizontal axis denotes the pass number.

3. Multi-code re-encoding

To facilitate various operations in semantic space, this paper further presents a method to encode each polysemous word with multiple codes by using the redesigned network. Since a polysemous word has more than one meaning, such as Chinese characters, it should occupy more than one position in the semantic space. The averaged code vector (1) cannot represent a polysemous word precisely. Although a code structure for accumulating the richness of semantic meanings of the polysemous word has been developed in [8], it cannot be used in many word space models, such as the word-based co-occurrence model [14] and the syntax-based semantic space model [15]. The popular cosine and Euclidean distance cannot be applied to this structure. This paper proposes an automatic method to assign more than one code for a polysemous word. This may benefit the research of text mining, ranking, indexing and categorization.

Consider a corpus with N different words, $\{c_n, n = 1 \sim N\}$. Instead of assigning one lexical code vector to each word c_n, we prepare a meaning pool matrix $(M_n)_{R\times B}$ for each of them. In this matrix, one word c_n can have at most B different codes. During the training process, the specific code vector used in the input sequence at time $t+1$, $p(t+1) = c_j$, is the sth column of the pool M_j^r which receives the minimum prediction error from the input $p(t)$, see Fig. 10. The sth meaning among all codes in the pool M_j^r is the best matched one of the prediction of its precedent word $p(t)$. Denote the sth column vector of M_j^r as $M_j^r(s)$. Note that we start from $p(0) = c_n$, where $n =$ "end of sentence". Set the selected input as

$$p(t+1) = M_j^r(s), \tag{10}$$

where the sth column vector minimizes $\{\|o(p(t)) - M_j^r(l)\|^2$ for all $l = 1 \sim B\}$.

One word may have a set of different code vectors when it appears in different locations of the sequence. All code vectors are fixed in each pass until we apply the re-encoding method to renew them at the end of the pass. The renewed vectors will be used in the next pass.

After every k training epochs, a new raw code for the sth meaning, $c_n(s)$, of the word c_n is calculated as follows:

$$c_n^{raw}(s) = M_n^r(s) = \frac{1}{freq_{n,s}} \sum_{\{t|p(t) = M_n^r(s)\}} o(p(t-1)), \tag{11}$$

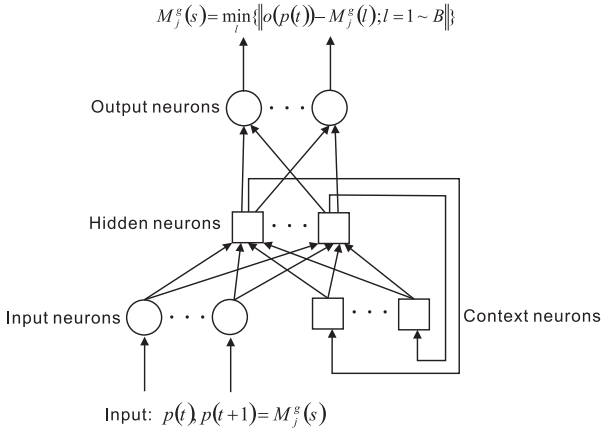

Fig. 10. Illustration of Elman network for multi-code.

where $freq_{n,s}$ is the number of times the code vector $c_n(s)$ appears in the sequence of the current pass. All raw codes are further normalized before using them in the next pass by the equation $c_n^{r+1}(s) = \|c_n^{ave}(s)\|^{-1} c_n^{ave}(s)$, where $\|c_n(s)\| = (c_n^T(s)c_n(s))^{0.5}$ and

$$C_{R\times S}^{ave} = C_{R\times S}^{raw} - \frac{1}{S} C_{R\times S}^{raw} \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & 1 & \vdots \\ 1 & \cdots & 1 \end{bmatrix}_{S\times S}. \quad (12)$$

In the normalization, $S=kT$ is the total number of different code vectors presented in the input sequence of the current pass.

Note that only the code vectors (or, all those matched vectors s) presented in the input sequence during the current pass will be updated and the normalization is only applied to them as well. In the next pass, all B vectors in each pool will be the candidates for the next input, $p(t+1)$, according to the minimization. The re-encoded vectors in the pool M_n after the minimum pass are the accomplished vectors for word meanings. They will be used to represent the meanings of the polysemous words. The minimum pass is the pass that reaches the minimum prediction error during all the training process. The last training pass may not be the minimum training pass. According to BP, the last epoch is, usually, the minimum epoch of the minimum pass.

As for the computation cost of Elman network, the number of operations for training a word, including forward and backward pass, is $O((L_i+L_c+1)\times L_h+(L_h+1)\times L_o)$. In each epoch, the network reads through the whole corpus that contains T words, the complexity is the computational complexity of training multiplies T. So, the computational complexity of a training epoch is $O(T)$ when the training algorithm is given and fixed.

In the method, we need to sort a list of all N distinct words. The sorting complexity is $O(N \log N)$. After the construction of the list, we assign B vectors to each distinct word and this assignment would not increase the computational complexity. During the training phase, the number of search operations in each word pool is $O(B\times R)$. This pool will not increase the computational complexity. This is because B and R are assigned at the beginning of the method. The computational complexity for each epoch is also $O(T)$.

The computational complexity for renewing the codes is $O(N\times R)$, the operation of summing all elements and of dividing by the square root of the sum in each column. The computation for the zero-mean would not increase the complexity.

3.1. Experiment on real corpus

We use the novel "A Tale of Two Cities" by Charles Dickens as our corpus. We apply the same grammar rules as before to preprocessing the corpus and reduce the amount of vocabularies. After preprocessing, there are 7231 different root words including "ING", "NVs", "'s", "Ved", and the mark that is added to represent the end of a sentence. We concatenated all sentences in the novel to form the training word stream. The length of the whole sequence is $T=157{,}482$.

From experiences, the dimension of each code vector is set to $R=15$. There are $L_i=15$ input neurons and $L_o=15$ output neurons. Set $L_h=L_c=30$. The initial weights and context neurons are assigned in the same way as before. To simplify the training, for those words with too high or too low frequency in the corpus, we set one code vector c_n to represent them, that is $B=1$. We prepare the code vector pools M_n for the words with frequency >50 and ≤ 200. There are 252 such words and 252 pool matrices M_n with $B>1$. The initial codes for all words $M_n^{r=1}$ are randomly assigned under the restriction that there are no repeated codes and they are normalized. Use the BP algorithm to reduce the prediction error, $\|o(p(t))-p(t+1)\|^2$. The learning rate is fixed to 0.01. W_{hic} and W_{oh} are updated after each input word presented. We set one epoch, $g=1$, as when all 157,482 words are presented, and we renew the codes after each pass, $k=100$ epochs.

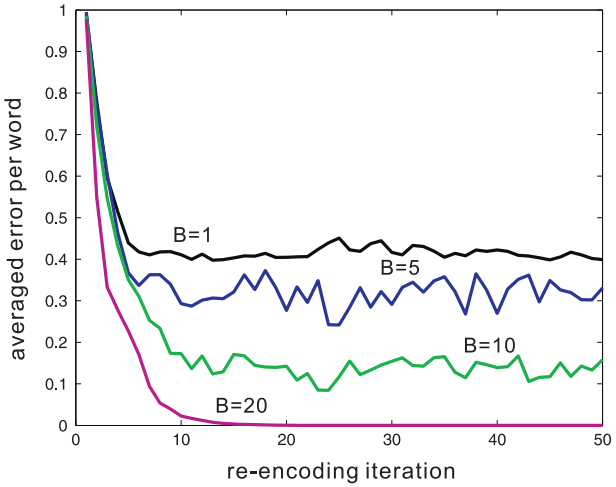

Fig. 11. Training errors of different pool sizes. The number in the horizontal axis is the number of renewals.

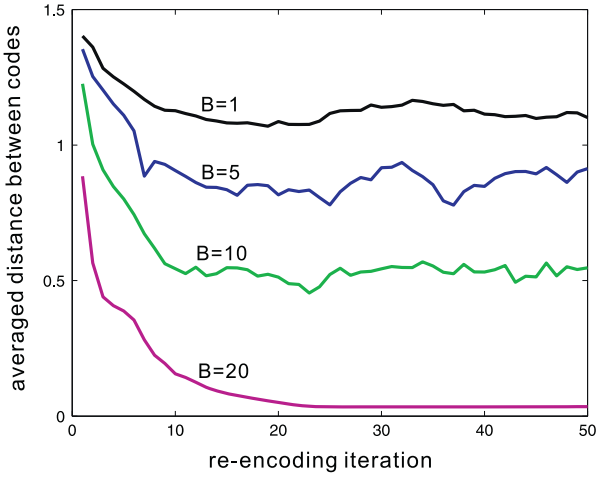

Fig. 12. Averaged distances between codes of different size pools. The number in the horizontal axis is the number of renewals.

We compare the experiment results for different sizes, B, of pools. Fig. 11 shows that the converged errors decrease when we enlarge the code pools. A portion of error reduction in Fig. 11 is caused by the decrease of difference between code vectors, see Fig. 12. This is because there are too many vectors available in the pool and it is easy to find a similar one in the pool. Note that similar vectors will give low error in the BP training process. Fig. 12 plots the averaged difference between all code vectors. The reduction in difference, Fig. 12, is coherent with that in Fig. 11. Fig. 13 plotted the averaged number of different code vectors which are assigned to one word. The number in the horizontal axis is the number of renewals.

3.2. Experiment on Chinese literatures

Each Chinese character is a word. Two greatest classical novels in Chinese literatures are analyzed. The first novel is *Dream of the Red Chamber*, which is composed by Cao Xueqin in the 18th century. The second is *Romance of the Three Kingdoms*, which is written by Luo Guanzhong in the 14th century. *Red Chamber* has more than 841 thousand characters and uses 5069 different Chinese characters, including punctuation marks. *Three Kingdoms*

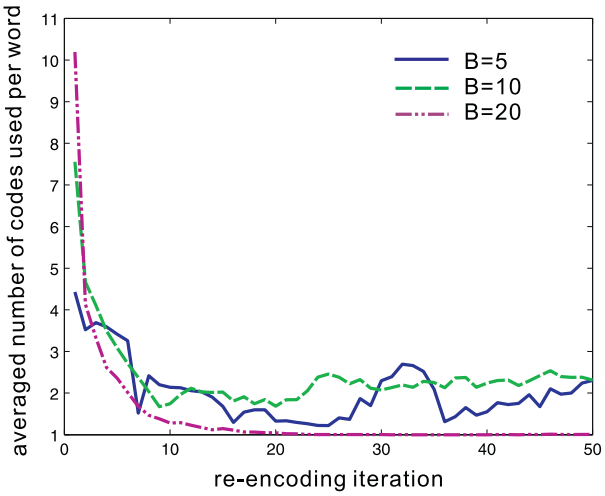

Fig. 13. Averaged number of different code vectors which are assigned to one word. The number in the horizontal axis is the number of renewals.

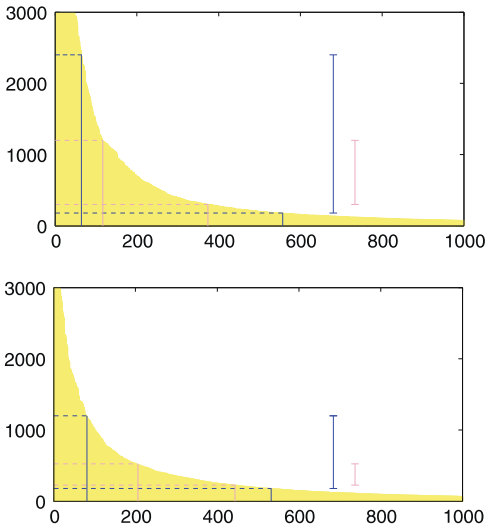

Fig. 14. Appearing frequencies of words in both novels are plotted. The word with a frequency larger than 3000 is skipped. The words with $B>1$ are marked by two colored lines. The red lines mark those words used in the first training stage and the green lines mark the words used in the second training stage. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

has more than 570 thousand characters and uses 5071 Chinese characters. Each kind of punctuation mark is treated as a single Chinese character. There are 27 kinds of such marks. In our training, each of the names, which consists of several Chinese characters, has a single matrix M_n.

A separate network is trained for each novel. All sentences and characters in each novel are concatenated together, sentence after sentence, into one single character sequence. This sequence will be used as the network input. In each epoch, the network starts from the first character of the novel and reads the characters one after one until the end character of the novel.

From experiences, we set $R=15$ for each code. The number of dimensions in the input layer is 15 as well as the output layer, $L_i=L_o=15$. Both context layer and hidden layer have 15 neurons. The initial weights and the context neurons are assigned as before. The initial codes for all characters in the first pass, $M_n^{r=1}$, are randomly assigned under the restriction that there are no repeated codes and they are normalized. Use the BP algorithm to reduce the prediction error, $\|o(p(t))-p(t+1)\|^2$. The learning rate is fixed to 0.01. The weights W_{hic} and W_{oh} are updated after each input character presented. We renew the codes after every $k=100$ epochs.

The polysemous meanings are much more severe in Chinese characters than that of "A Tale". To ease the problem, for the character with too high or too low frequency in the corpus, we set one single code vector c_n to represent it, that is $B=1$. We prepare the code vector pools M_n for the rest characters whose appearing frequency within the range $\{\geq 300$ and $\leq 1200\}$ in *Red Chamber*. There are 246 such characters and 246 pool matrices M_n with $B>1$. As for the novel *Three Kingdoms*, there are 258 character pools with $B>1$ whose appearing frequency within the range $\{\geq 225$ and $\leq 525\}$. The appearing frequency of each distinct word is plotted in Fig. 14. The words with multiple codes, $B>1$, are marked by red lines in the figure. The horizontal axis of this histogram is labeled with the word numbers for the 20,000 words with high frequencies. The vertical axis is labeled with their appearing frequencies. The maximal frequency of the word in *Red Chamber* is 60,317 and that of *Three Kingdoms* is 44,373. The frequency distributions of word are roughly the same for the two novels.

We compare the experiment results using different pool sizes, $B=1$, 5, 10, and 20. The results for *Red Chamber* are recorded in Fig. 15. This figure recorded the averaged prediction error for each word during the first 40 training passes. The four color vertical lines mark the minimum passes for the four pool sizes. The

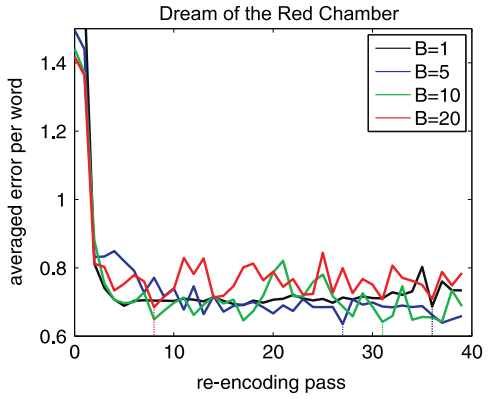

Fig. 15. Training errors using different pool sizes. Color vertical lines mark their minimum passes. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

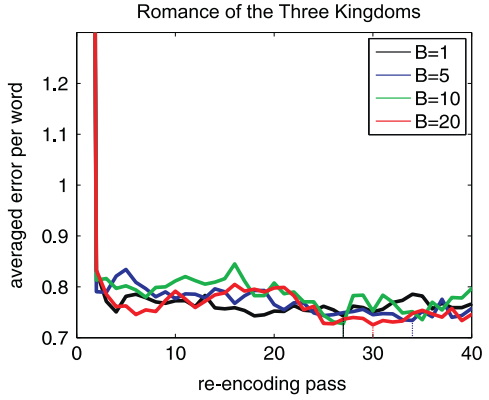

Fig. 16. Training errors using different pool sizes. Color vertical lines mark their minimum passes. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

training errors decrease roughly when we enlarge the pool size. The averaged prediction errors for *Three Kingdoms* are plotted in Fig. 16.

Table 3
Characters have multiple codes. The total number of meanings of a character is labeled next to its character in the bracket (). (For interpretation of the references to color in this table, the reader is referred to the web version of this paper.)

紅樓夢	三國演義
張 (2) 、真 (4) 、輕 (2) 、 花 (4) 、分 (2) 、長 (3) 、 把 (3) 、思 (2) 、答 (2) 、 紅 (2) 、經 (3) 、方 (5) 、	車 (2) 、發 (2) 、差 (2) 、 合 (2) 、陣 (2) 、投 (2) 、 成 (3) 、禮 (3) 、飛 (2) 、 老 (2) 、騎 (2) 、勢 (2) 、

Table 4
Sentences in *Red Chamber* contain the same character with two different meanings, $s=2$ and $s=5$.

紅樓夢

Sentences	Meaning
林黛玉便知是鳳姐來了，連忙立起身說道：「我打後院子裡去罷，回來再來。」寶玉一把(2)拉住說道：「這可奇了，好好的怎麼怕起她來？」	將
如今誰承望姑娘人大心大，不把(2)我放在眼裏，倒把外四路的什麼寶姐姐、鳳姐姐的放在心坎兒上，倒把我三日不理四日不見的。	將
晴雯笑著，便倚在床上說道：「我也乏了，明兒再撕罷。」寶玉笑道：「古人云，『千金難買一笑』，幾把(5)扇子能值幾何？」	隻
寶玉便笑求她：「好姐姐，你把(2)那湯拿了來我嘗嘗。」玉釧兒道：「我從不會餵人東西，等她們來了再吃。」	將
今年春天，老爺不知在哪個地方看見了幾把(5)舊扇子，回家來，看家裏所有收著的這些好扇子都不中用了，立刻叫人各處搜求。	隻

Table 5
Sentences in *Three Kingdoms* contain the same character with two different meanings, $s=2$ and $s=4$.

三國演義

Sentences	Meaning
郤在山上大吹大擂飲酒，並不下山。張飛令軍士大罵，郤只不出。飛只得還(2)營。次日，雷同又去山下搦戰。	返
忠聽了，白鬚倒豎而言曰：「某雖老，兩臂尚開三石之弓，渾身還(4)有千斤之力；豈不足敵張郤匹夫耶？」孔明曰：「將軍年近七十，如何不老？」	尚
孔明曰：「王上既允所請，便可築臺擇吉，恭行大禮。」即時送漢中王還(2)宮，一面令博士許慈、諫議郎孟光掌禮，築臺於成都武擔之南。	返
孔明囑馬岱曰：「匠作人等，不許放出；外人不許放入。吾還(4)不時自來點視。捉司馬懿之計，只在此舉。切不可走漏消息。」	尚
燕王涕泣而去。遂封曹爽為大將軍，總攝朝政。叡病漸危，急令使持節詔司馬懿還(2)朝。懿受命逕到許昌，入見魏主。	返

We study the accomplished codes in $B=5$ after $r=27$ re-encoding renewals. The left part of Table 3 lists the 12 characters in *Red Chamber* that have multiple meanings. The red underlines mark the characters that have the highest number of meanings. The total number of meanings of a character is labeled next to its character. The right portion of this table lists 12 characters in the novel *Three Kingdoms* that have multiple meanings obtained after the minimum pass, $r=35$.

One multi-meaning character is shown in Table 4. This character has three meanings. Two meanings of them in different contexts are “a grip” or “a bundle”. Five samples are listed in this table. The sth meaning of the character in M, $1\le s\le B=5$, is labeled next to its character. A multi-meaning character in *Three Kingdoms* is shown in Table 5. This character has two meanings, “return” and “still”. Five samples are listed in this table.

Red Chamber has 15 names that have multiple codes and *Three Kingdoms* has 6 names that have multiple codes. Several samples are listed in Table 6. The different codes for the same name imply different roles or characteristics in different contexts.

Note that a portion of error reduction for large size B in Figs. 15 and 16 is also caused by the decrease of difference between code vectors by using BP.

Table 6
Samples of two names having multiple codes.

紅樓夢	三國演義
賈政 (4) 也撐不住笑了。因說道：「那怕再念三十本《詩經》，也都是掩耳偷鈴，哄人而已。你去請學裏師老爺安，就說我說的：什麼《詩經》、古文，一概不用虛應故事，只是先把《四書》一齊講明背熟，是最要緊的。」....李貴忙答應「是」，見賈政 (5) 無話，方退了出去。此時，寶玉獨站在院外，避貓鼠兒似的，屏聲靜候。....當下，賈家眾人齊來弔問，榮國府賈赦贈銀二十兩，賈政 (4) 亦是二十兩，寧國府賈珍亦有二十兩，別者族中貧富不一，或三兩或五兩，不可勝數。....水溶十分謙遜，因問賈政 (5) 道：「哪一位是銜玉而誕者？幾次要見一見，都為雜冗所阻。想今日是來的，何不請來一會？」	卓怒，引軍同李儒出迎。兩陣對圓，只見呂布 (3) 頂束髮金冠，披百花戰袍，擐唐猊鎧甲，繫獅蠻寶帶，縱馬挺戟，隨丁建陽出到陣前。....董卓未及回言，呂布 (4) 飛馬直殺過來。董卓慌走，建陽率軍掩殺。....卓曰：「吾觀呂布 (3) 非常人也。吾若得此人，何慮天下哉？」....肅曰：「某聞主公有名馬一匹，號曰『赤兔』，日行千里。須得此馬，再用金珠，以利結其心。某更進說詞，呂布 (4) 必反丁原，來投主公矣。」卓問李儒曰：「此言可乎？」儒曰：「主公欲取天下，何惜一馬？」卓欣然與之，更與黃金一千兩、明珠數十顆、玉帶一條。李肅齎了禮物，投呂布寨來。伏路軍人圍住。

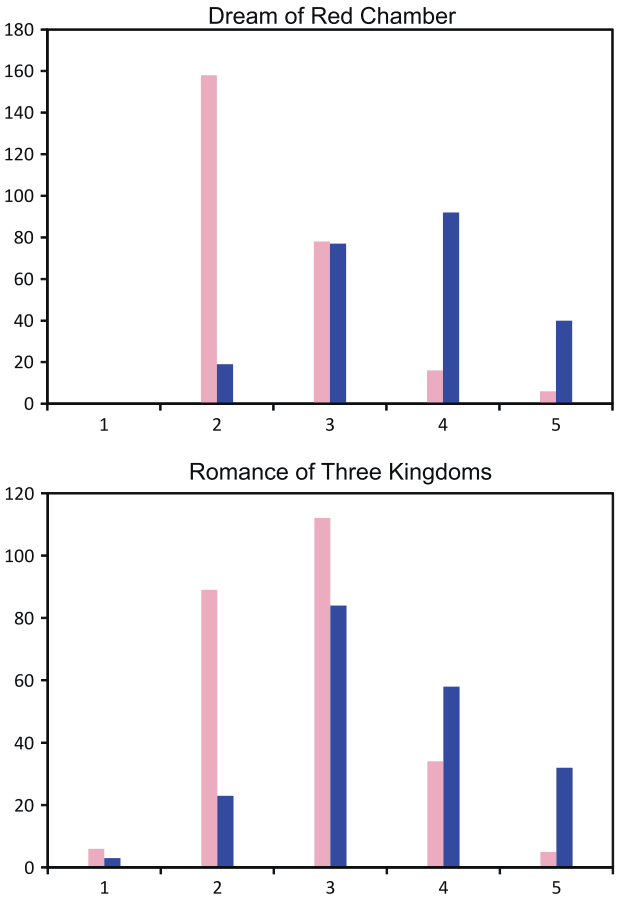

Fig. 17. The histogram of the Chinese characters that have different meanings estimated from the two novels. The pink lines are those words in the range $\{\ge 300$ and $\le 1200\}$ for *Red Chamber* and $\{\ge 225$ and $\le 525\}$ for *Three Kingdoms*. The blue lines are words in the enlarged ranges in the successive stage. The enlarged range for *Red Chamber* is $\{\ge 180$ and $\le 299\}$ and $\{\ge 1201$ and $\le 2400\}$. The enlarged range for *Three Kingdoms* is $\{\ge 180$ and $\le 224\}$ and $\{\ge 526$ and $\le 1200\}$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

The results on the experiment, $B=5$, are further analyzed. We calculate the statistics for the trained codes. The histogram in Fig. 17 shows the number of characters that have a different number of meanings. There are many characters that have five meanings in the *Red Chamber*. *Three Kingdoms* uses less number of meanings for a character. This suggests that the romantic fiction *Red Chamber* uses much diverse meanings. The historical novel *Three Kingdoms* tends to use accurate meaning to keep integrity and consistency.

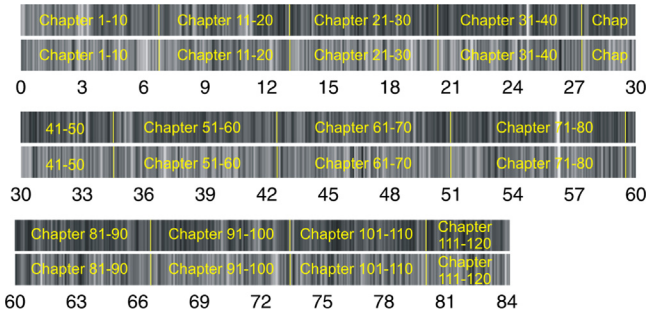

Fig. 18. The prediction errors along the novel Red Dream. Each number along the horizontal line indicates the number of thousands of Chinese characters. The novel is segmented every 10 chapters. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

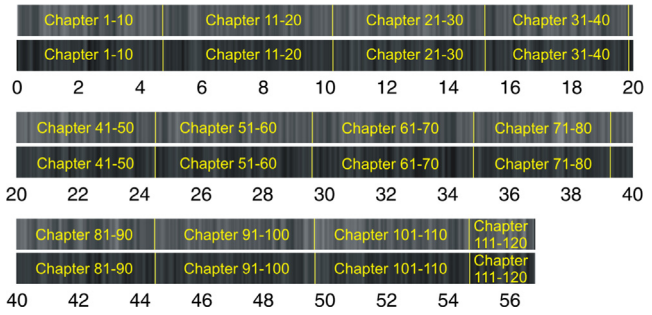

Fig. 19. The prediction errors along the novel Three Kingdoms. Each number on the horizontal line indicates the number of thousands of Chinese characters. The novel is segmented every 10 chapters.

Figs. 18 and 19 record the prediction errors of all characters in the experiment, $B=5$. The errors are averaged by a Gaussian filter to exhibit the error distribution along words. The standard deviation of the filter is 500 characters wide and its window size is 1001 characters. Figs. 18 and 19 showed that the word sequence of *Red Chamber* is less predictable. This means that there are shifts in writing tstyles in different places. This novel has been received a lot of discussion on its authorship. *Three Kingdoms* is much more accurate.

Note that the word ranges for *Red Chamber*, $\{\geq 300$ and $\leq 1200\}$, can be enlarged in successive training stages to include more words with $B>1$. Fig. 14 plots the words, marked by green lines, that will be trained in the next stage. Those 246 trained word codes are fixed in the next stage. New pools in the enlarged range will be re-encoded and the network will be retrained using the trained weights in the previous stage as its initial weights. This successive enlargement strategy can reduce the amount of computations. The distinct number of accomplished meanings is plotted in Fig. 17 with blue lines for the words in the next stage.

4. Conclusions

Through the re-encoding method, we obtain a set of new codes with reduced entropy over renewals. Both form-based similarity and function-based similarity can be loaded into the code [2]. This reduction is consistent with Barlow's idea and provides independent representations of words, see Fig. 9. To our knowledge, this is the only constructive method to reach such coding.

Judging from the renewed codes in Fig. 2, the decreased error by the BP algorithm on weight adjusting is far less than that of code renewing. To our knowledge, it is not possible to reach such low errors without the renewed codes using any existing algorithms.

The trained codes depend directly on the concatenated words in the novel work. This relates the codes and the network to their novel and brings these codes into play. The trained network and codes should be saved with their novel for further semantic uses.

We showed the more general approach to deal with the multi-code for the polysemous word. To achieve this goal, we associate with each word with a set of distinct codes and provide the multi-code re-encoding method to assign different meanings to each word automatically. The resolution of the polysemous word impacts the accuracy of semantic search, machine translation, etc. and has been described as an AI-complete problem. To our knowledge, this method is the only one that automatically resolves multiple meanings for the polysemous word in semantic space.

We showed that the method can be applied to the stylish analyses of both Chinese and English literary works.

Acknowledgments

This work was supported by National Science Council under Project NSC 100-2221-E-002-234-MY3.

References

[1] J.L. Elman, Finding structure in time, Cogn. Sci. 14 (1990) 179–211.
[2] J.L. Elman, Generalization, simple recurrent networks, and the emergence of structure, in: M.A. Gernsbacher, S.J. Derry (Eds.), Proceedings of the Twentieth Annual Conference of the Cognitive Science Society, Lawrence Erlbaum Associates, Mahwah, NJ, 1998.
[3] H. Siegelmann, Computation beyond the Turing limit, Science 268 (5210) (1995) 545–548.
[4] 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), Leysin, Switzerland, 1997, pp. 215–237.
[5] C.B. William, Mendenhall's studies of word-length distribution in the works of Shakespeare and Bacon, Biometrika 62 (1975) 207–212.
[6] Java Graphical Authorship Attribution Program ⟨http://evllabs.com/jgaap/w/index.php/Main_Page⟩.
[7] The Signature Stylometric System by Peter Millican ⟨http://www.philocomp.net/humanities/signature⟩.
[8] C.-Y. Liou, J.-C. Huang, W.-C. Yang, Modeling word perception using the Elman network, Neurocomputing 71 (2008) 3150–3157.
[9] D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (1999) 788–791.
[10] D.E. Rumelhart, Parallel distributed processing: explorations, in: J.L. McClelland (Ed.), The Microstructure of Cognition, MIT Press, Cambridge, MA, 1986.
[11] J.-C. Huang, W.-C. Cheng, C.-Y. Liou, Distributed representation of word, in: N.T. Nguyen, C.-G. Kim, A. Janiak (Eds.), ACIIDS 2011, Lecture Notes in Artificial Intelligence, vol. 6591, Springer, Heidelberg, 2011, pp. 169–176.
[12] H.B. Barlow, T.P. Kaushal, G.J. Mitchison, Finding minimum entropy codes, Neural Comput. 1 (1989) 412–423.
[13] H.B. Barlow, Unsupervised learning, Neural Comput. 1 (3) (1989) 295–311.
[14] W. Lowe, Towards a theory of semantic space, in: Proceedings of the 23rd Annual Conference of the Cognitive Science Society, Edinburgh, UK, 2001, pp. 576–581.
[15] G. Grefenstette, Explorations in Automatic Thesaurus Discovery, Kluwer Academic Publishers, Dordrecht, 1994.

Cheng-Yuan Liou was born in Taiwan 1951. He received the B.S. degree in physics from National Central University in 1974, the M.S. degree in oceanography from National Taiwan University in 1978, and the Ph.D. degree in ocean engineering from Massachusetts Institute of Technology in 1985. From 1986 to 1987 he was a visiting associate professor in Institute of Applied Mechanics, National Taiwan University. In 1988 he joined the faculty at the same university, where he is currently a professor in the Department of Computer Science and Information Engineering. His interests include neural networks, artificial intelligence, information science, visualization, semantic indexing, and brain theory. In the year 1993, he reported the discovery that the EEG signals generated from different conscious mental tasks can be discriminated between tasks. His early article in 1978 reported the discovery of two ocean tidal currents surrounding Taiwan and article in 1980 reported the highest ocean wave ever measured, 12.47 meters, near coast of Taiwan.

Wei-Chen Cheng is currently working at National Taiwan University. He was a postdoctoral research fellow in Information Systems Technology and Design at Singapore University of Technology and Design in 2014. He was a postdoctoral fellow in Institute of Statistical Science at Academia Sinica from 2010 to 2013. He received his PhD from the Department of Computer Science and Information Engineering at the National Taiwan University. He received his MSc in the same department. His research interests include neural network, artificial intelligence, language modeling, and information visualization.

Jiun-Wei Liou was born in 1981 in Taiwan. He received his PhD from Department of Computer Science and Information Engineering at the National Taiwan University in 2013.

Daw-Ran Liou was born in Taiwan 1989. He received the B.S. degree from the Department of Mechanical Engineering, National Taiwan University in 2011 and the M.S. degree from the Sibley School of Mechanical and Aerospace Engineering, Cornell University in 2013. His research interests include robotics, vision systems, and artificial intelligence.